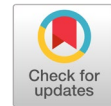


Tumor-Net: convolutional neural network modeling for classifying brain tumors from MRI images



Abu Kowshir Bitto ^{a,1,*}, Md. Hasan Imam Bijoy ^{b,2}, Sabina Yesmin ^{a,3}, Imran Mahmud ^{a,4}, Md. Jueal Mia ^{b,5}, Khalid Been Md. Badruzzaman Biplob ^{a,6}

^a Department of Software Engineering, Daffodil International University, Daffodil Smart City, Dhaka, 1207, Bangladesh

^b Department of Computer Science and Engineering, Daffodil International University, Daffodil Smart City, Dhaka, 1207, Bangladesh

¹ abu.kowshir777@gmail.com; ² hasan15-11743@diu.edu.bd; ³ sabina35-2406@diu.edu.bd; ⁴ imranmahmud@daffodilvarsity.edu.bd; ⁵ mjueal02@gmail.com; ⁶ khalid@daffodilvarsity.edu.bd

* corresponding author

ARTICLE INFO

Article history

Received July 18, 2022

Revised January 30, 2023

Accepted February 20, 2023

Available online April 7, 2023

Keywords

Brain tumor

MRI images

VGG16

VGG19

ResNet50

ABSTRACT

Abnormal brain tissue or cell growth is known as a brain tumor. One of the body's most intricate organs is the brain, where billions of cells work together. As a head tumor grows, the brain suffers damage due to its increasingly dense core. Magnetic resonance imaging, or MRI, is a type of medical imaging that enables radiologists to view the inside of body structures without the need for surgery. The image-based medical diagnosis expert system is crucial for a brain tumor patient. In this study, we combined two Magnetic Resonance Imaging (MRI)-based image datasets from Figshare and Kaggle to identify brain tumor MRI using a variety of convolutional neural network designs. To achieve competitive performance, we employ several data preprocessing techniques, such as resizing and enhancing contrast. The image augmentation techniques (E.g., rotated, width shifted, height shifted, shear shifted, and horizontally flipped) are used to increase data size, and five pre-trained models employed, including VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3. The model with the highest accuracy, ResNet-50, performs at 96.76 percent. The model with the highest precision overall is Inception V3, with a precision score of 98.83 percent. ResNet-50 performs at 96.96% for F1-Score. The prominent accuracy of the implemented model, i.e., ResNet-50, compared with several earlier studies to validate the consequence of this introspection. The outcome of this study can be used in the medical diagnosis of brain tumors with an MRI-based expert system.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

The unnatural growth of brain tissue or cells is referred to as a brain tumor [1]- [4]. When malignancy grows in the head, the weighted interior of the brain expands, causing damage to the brain. An intracranial neoplasm, often a brain tumor, is a disorder where unexpected cells grow in the human brain. Two varieties of brain tumors exist malignant (cancerous) and benign (noncancerous). Tumors that cause cancer might be primary, metastases, or secondary tumors. When the DNA of normal brain cells has a flaw, it results in brain tumors. As we know, cells in the body constantly split and die, only to be replaced by another cell. Modern cells are formed in several circumstances, but the old cells are eliminated. These cells coagulate as a result, and they have the potential to form tumors. Brain tumors are frequently passed down through the generations. Gliomas are the most prevalent and powerful [5], [6]. Glioma detection at an early stage is critical for achieving the best treatment results [7]. Computed Tomography (CT), Attractive Resonance Imaging (MRI), etc. provide essential details on brain tumors'

form, size, location, and digestive system. While these modalities are used in combination to provide the most detailed information about brain tumors, MRI is considered the standard strategy due to its significant delicate tissue differentiation and broad accessibility [8], [9].

Many papers, publications, and research projects focus on detecting and categorizing brain tumors from MRI images. The study of Pereira *et al.* [2] described a CNN-based approach for segmenting brain tumors in MRI images. In order to accommodate deeper architectures, the CNN is constructed using small 3x3 kernels over convolutional layers. Brain Tumor Division identified a few strategies for producing a parametric / non-parametric classification of the fundamental architecture data in this article. These models frequently include probability work compared to perceptions and a prior model. Preprocessing, classification by CNN, and post-processing are the three key parts of the approach. The BRATS 2013 and 2015 databases were used to test the suggested technique. Because brain tumors vary greatly in their spatial placement and basic composition; researchers have looked into the use of data increase to deal with this inconstancy. They looked at expanding our prepared data collection by pivoting patches and inspecting underrepresented HGG classes in LGG. The authors of a research by Deepak *et al.* [10] classified the three most common forms of brain tumors—gliomas, meningiomas, and pituitary tumors—into three different categories. Deep transfer learning and a Google Neural Network that had been trained to extract characteristics from brain MRI data were used in their suggested classification strategy. The authors extracted features using a pre-trained VGG-16 and fine-tuned AlexNet models. Support vector machines were then used to classify the features (SVM). Knowledge transfer was the foundation of the 3D CNN architecture used in the learning process. The accuracy metrics specified in their study were used by the authors to compare their transfer learning-based strategy with hand-crafted feature engineering.

Talo *et al.* [11] proposed a procedure for mechanically categorizing functional and dysfunctional brain MR pictures using deep learning. MR imaging is a well-known non-invasive technique for measuring neuronal movement in the brain of a human. MR pictures offer a huge unrealized for providing important data about numerous brain disorders' mentality, analysis, genetic characteristics, hemodynamics, and chemistry. ResNet34, Convolutional neural networks (CNNs) are the foundation of a deep learning model, is also used. The PSO SVM classifier with a basis function radial kernel was used with the Res-Net34 model, and it successfully identified brain anomalies. The experiments in this paper were carried out using the Harvard Medical School MR dataset, and numerous sophisticated deep learning approaches for hyperparameter optimization were used. Ahuja *et al.* [12] proposed employing the super pixel approach to detect and segment brain tumors via transfer learning. For starters, MRI cuts are divided into three categories: particular, typical, LGG, and HGG. Utilizing VGG-19 at epoch-6, the proposed methodology consistently produced validation data precision of 99.82 percent and 96.32 percent. The LGG and HGG MRI brain tumor pictures now used for segmenting the tumor. The super-pixel approach is used to divide tumors. The tumor segmentation yields a 0.932 normal dice file. The technique should be tested on a real-time patient database in the future. Supporting advancement in preprocessing procedures in division organizations is essential to raise the average dice list's quality. Khan *et al.* [13] describe a totally automatic profound learning method for multifunctional brain tumor classification that includes differentiating enhancement. The work's quality was graded in three stages. To begin, differentiation extending using the edge-based interface; he was used to prolong the image contrast of the tumor region in the preprocessing step. Furthermore, the use of general underlined the option of robust deep learning. Finally, the ELM classification was updated to categorize reported tumors into the appropriate group. This research extracted features derived from two various CNN models using transfer learning, and the synthesis was done. The goal of combining two CNN models was always to create a more modern include variable with more data. The test was run on the BraTs datasets, and the results showed an increase in precision (98.16 percent, 97.26 percent, and 93.40 percent, respectively, for the BraTs2015, BraTs2017, and BraTs 2018 datasets).

Kaur *et al.* [14] evaluated various pre-trained deep convolutional neural network (DCNN) models with TLCs for MR brain image prediction. The authors achieved high recognition accuracy by employing pre-trained DCNN models and exchange learning. Among the models tested, AlexNet

performed the best, with 100%, 94%, and 95.92% classification rates for all datasets. Their findings suggest that the use of pre-trained DCNN models and exchange learning can significantly improve the accuracy of MR brain image classification. They arose due to existing conventional and deep learning algorithms based on the categorization of brain tasks. In contrast, the author describes next work that will concentrate on running models on frameworks with GPU-enabled capacity, which is anticipated to reduce computational overhead and investigate various fine-tuning techniques. Khan *et al.* [13] describe an automatic multimodal classification technique for bra based on deep learning for brain tumor type categorization. Cancerous and noncancerous brain tumors exist [15]. It has the potential to injure the brain, which might be fatal. This study used a direct distinguish upgrade technique modified with the help of histogram equalization. The extracted features from two different CNN models were done via exchange learning, and the integration was done. The goal of combining two CNN architectures was to provide more data to an underused highlight vector.

According to the current state-of-art of categorization scheme for identifying brain tumors, there are more than 120 different types of brain tumors that vary in their origin, range, size, and features. This raises the chance of tumors, which can be caused by a genetic disorder called neurofibromatosis, furthermore exposure to chemicals like vinyl chloride, Epstein-Barr virus, and ionized radiation. Our study article's main objective is to determine how to employ transfer learning based on deep learning to identify the tumor from MRI pictures. The method of leveraging the information gained through a planned demonstration to acquire a new set of facts is known as transfer learning [16]. The transfer learning method is called inductive trade learning when labeled data is present inside the source and target areas for a classification problem [17]. Brain magnetic reverberation imaging (MRI) is one of the most trustworthy imaging modalities that analysts rely on for diagnosing brain malignancies and predicting tumor growth, both in the detection and therapy stages [18]. To contribute on previous studies, this study developed numerous computational techniques for brain tumor detection and classification utilizing brain MRI images since it became possible to channel and stack meaningful images to the computer. For the category and location of tumors, a CNN-based multi-task classification is built. We used Kaggle and Figshare datasets for this study. A total of 7138 brain MRI scans have been divided into four categories: pituitary, meningioma, glioma, and no tumor, where the CNN pre-trained model will be used.

The remainder of the paper is provided below. The proposed methodology for detecting brain tumors is described in Section 2, the results are shown and discussed in Section 3, along with a comparison of the methods used in Section 3, and the study is concluded in Section 4.

2. Method

The main goal of our study is to develop a deep learning-based transfer learning classifiers (TLCs) model that can distinguish tumors from MRI images. We must go through numerous phases to attain our aim, including dataset collecting, data preprocessing, model creation, etc. In Fig. 1, the functioning procedure is presented.

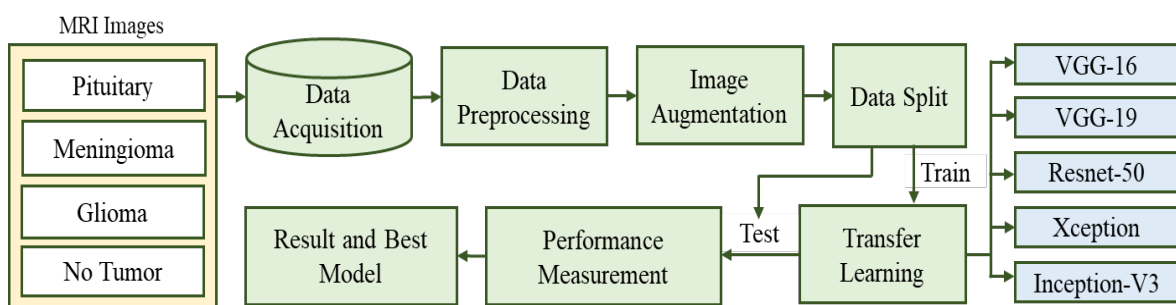


Fig. 1. Working procedure diagram to classify the brain tumor disease from MRI images

2.1. Dataset Description

We used Figshare and Kaggle to obtain brain tumor data from MRI images [19]. This dataset has 7138 brains MRI pictures, divided into four categories: pituitary, meningioma, glioma, and no tumor. Unusual growths that form in the pituitary gland are known as pituitary tumors. The size of a pea, this gland is an organ. It is situated near the base of the brain, behind the nose. Some of these tumors cause the pituitary gland to produce excessive hormones regulating vital bodily processes. One kind of tumor that develops close to the brain is a meningioma. While most of these tumors are benign (around 90%), some develop into a malignancy. Cell growth, known as a glioma, begins in the brain or spinal cord. Glial cells, which are healthy brain cells, resemble the cells in gliomas. Glial cells support nerve cells' functionality by surrounding them. A tumor is a mass of cells that develops when a glioma expands. Colors have been applied to the photographs taken, and example data has been presented in Fig. 2.

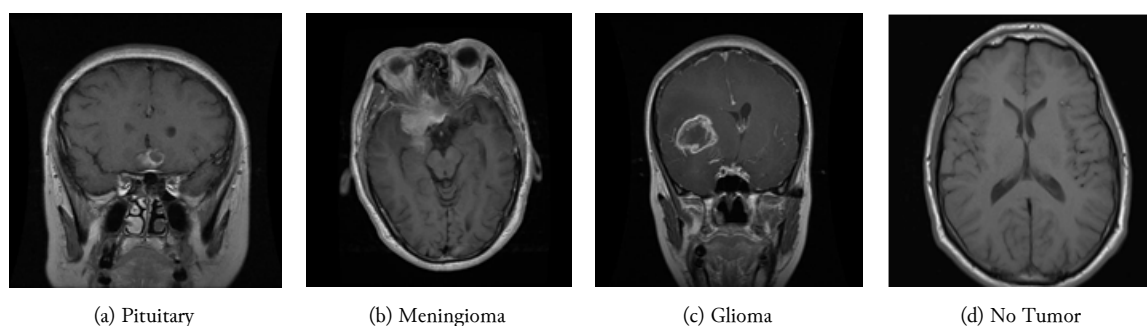


Fig. 2. Sample dataset for (a) Pituitary, (b) Meningioma, (c) Glioma and (d) No Tumor

2.2. Data Processing

Methods for data preparation use geometrical modifications. Scale, or image normalization, is used to map the data between an input image and an output image with a set of aligned image pairs for model development. The translation is used to map the data between an input image and an output image. Rotation is also used to correct the direction for the used accurate dimension of the image. The data resolution was lowered during the whole planning process. The picture pixels are 220x220 for VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3. The quality of each photograph is the same high standard. The images were rotated, sheared, moved in width and height, and horizontally flipped based on the changes to the images.

2.3. Model Implementation

This study used the Convolutional Neural Network (CNN) based transfer algorithm for the brain tumor dataset. Transfer learning models relevant theory given below.

Transfer supervised machine-learning approach in which a show made for an errand is used as the project focusing on a significant task [20], [21]. Given the enormous computation, it is a common method in DL to use pre-trained algorithms as the preliminary step on computer vision and normal language-generating assignments. In computer vision, neural systems ordinarily point to identify edges within the first layer, shapes within the center layer, and task-specific highlights within the last-mentioned layers. The early and central layers are utilized in transfer learning, and the areas of the last-mentioned layer were retrained. It makes use of the named information from the errand it was prepared on.

VGG-16 (Fig. 3) is highly appealing because of its architecture, consisting of 16 convolutional layers [22]. There are many filters but only 3x3 convolutions, which makes it very close to AlexNet. On four GPUs, it might be taught for two to three weeks. However, it is currently the most widely used technique in the neighborhood for extracting information from pictures. Open-source feature extractors have used the VGG weight setting as a starting point in a range of other applications and problems. However, managing VGG might be challenging due to its 138 million parameters [23]. Gaining VGG is possible

through transfer learning. The model has already been trained on a dataset, the parameters have been tweaked for greater accuracy, and the parameter values can be applied.

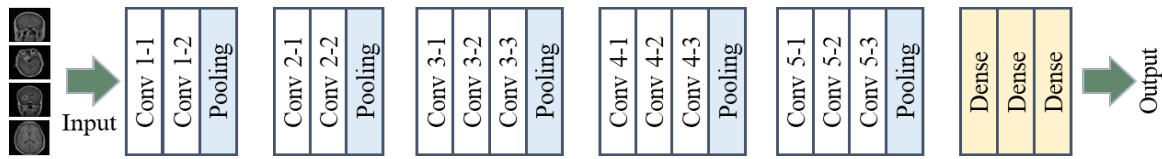


Fig. 3. VGG-16 block/architecture diagram

The VGG-19 architecture [24] consists of five convolutional blocks, which are implemented by three fully linked layers (Fig. 4). Following each convolution, an enhanced direct unit (ReLU) is developed, and the spatial dimension is then minimized at intervals using the max-pooling technique. To guarantee each physical measurement of the layer of neurons from the preceding layer is isolated, max-pooling layers use 2x2 sections with a walk of 2 and no cushioning. Currently employing the ultimate 1,000 fully softmax layer are two wholly connected layers with 4,096 ReLU enacted units. Extraction layers, which can be thought of as a subset of them, are included in convolutional components. The bottleneck characteristics are produced by the actuation maps produced by these layers.

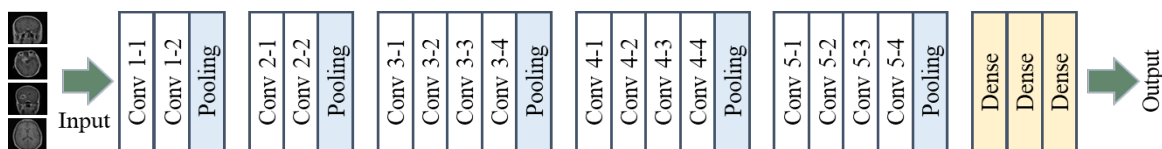


Fig. 4. VGG-19 block/architecture diagram

ResNet-50 (Fig. 5) is a CNN architecture, which is a variant of the ResNet model and is also known as a "Residual Neural Network" [25], [26]. ResNet50 is capable of processing up to 50 neural network layers. The ResNet50 architecture includes one MaxPool layer, one Normal Pool layer, and 48 Convolution layers. The model is known for its high computational performance, with 3.8×10^9 floating-point operations.



Fig. 5. ResNet-50 block/architecture diagram

Xception uses a 71-layer CNN-based architecture (Fig. 6). By loading a pre-trained network that has been trained on more than a million photos from the ImageNet database, the Xception model uses transfer learning. The Inception architecture has been changed to become Xception [27], where depth-wise Separable Convolutions have taken the role of the fundamental Inception modules. This change provides for better performance and more efficient calculation.

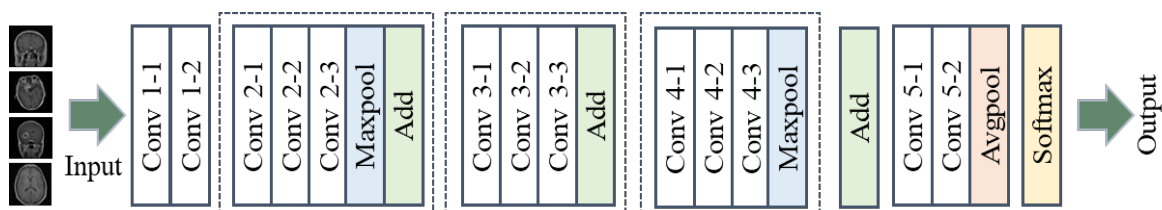


Fig. 6. Xception block/architecture diagram

The Inception Net (Fig. 7) is a CNN-based architecture that introduced a new standard for CNN classifiers by improving performance and accuracy while maintaining computational efficiency [28]. Inception Modules were developed to address over-fitting and computational expense by reducing dimensionality with stacked 1×1 convolutions, which enabled more efficient computation and deeper networks. The Inception Module consists of multiple layers, including 1×1 Convolutional Layer, 3×3 Convolutional Layer, and 5×5 Convolutional Layer. The output filter banks of each layer are combined into a single output sequence, which serves as input for the next phase. This approach helps to improve accuracy and computational efficiency in CNN classifiers.

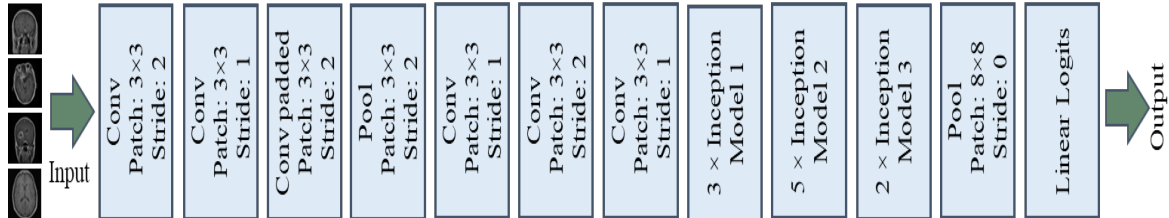


Fig. 7. Inception-V3 block/architecture diagram

2.4. Performance Calculation

After fitting/training the models, we utilized test data to estimate their performance. The metrics that were calculated for performance evaluation are listed below. We identified the model that could predict the outcome best using these parameters. Many percentage performance metrics [29] have been generated using Equations (1) to (7) based on the confusion matrix (CM) provided by the model.

$$Accuracy = \frac{True\ Positive + True\ Negative}{Total\ Number\ of\ Sentiment} \times 100\% \quad (1)$$

$$True\ Positive\ Rate\ (TPR)\ or\ Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \times 100\% \quad (2)$$

$$True\ Negative\ Rate\ (TNR) = \frac{True\ Negative}{False\ Positive + True\ Negative} \times 100\% \quad (3)$$

$$False\ Positive\ Rate\ (FPR) = \frac{False\ Positive}{False\ Positive + True\ Negative} \times 100\% \quad (4)$$

$$False\ Negative\ Rate\ (FNR) = \frac{False\ Negative}{False\ Negative + True\ Positive} \times 100\% \quad (5)$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \times 100\% \quad (6)$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \quad (7)$$

3. Results and Discussion

We used a deep learning-based five transfer learning classifiers (TLCs) model to predict and classify the tumor from MRI images. A split of 80:20 was used to divide the 5712 tumor training photos and 1404 validation images. The test platform is run on an Intel Core i5 computer with 16 GB of RAM. The resolutions of each input image were scaled to 220×220, 220×220, 220×220, and 220×220 for the VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3 models, accordingly. These models were employed in our study to upscale images to 220×220-pixel resolution. Pre-trained VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3 model weights were used. For each model that is provided, Table 1 displays the four-class resulting confusion matrix from [30], [31] (TP, FN, FP, TN). According to the classification, the procedure can unquestionably offer precise and accurate outcomes.

We set up train, testing, and cross-validation for the transfer learning algorithm to acquire correct results on our dataset. We used Adam optimizer [32], [33] to calculate each parameter's learning ratio. After using the classification method, we constructed the confusion matrix for each machine learning

(ML) and deep learning (DL) model. The assessment for ML and a confusion matrix represents deep learning classification. It places great emphasis on metering accuracy, precision, and F1-Score. Each of the following rates is accurately calculated.

Table 1. Confusion matrices for applied five transfer learning algorithms

Model	Disease	TP	FN	FP	TN
VGG-16	Pituitary	818	48	23	515
	Meningioma	774	43	18	569
	Glioma	722	32	23	627
	No Tumor	671	53	8	672
VGG-19	Pituitary	840	55	14	495
	Meningioma	773	65	13	553
	Glioma	669	51	11	673
	No Tumor	868	22	10	504
Resnet-50	Pituitary	798	25	15	566
	Meningioma	760	12	19	613
	Glioma	744	36	26	598
	No Tumor	723	34	22	627
Xception	Pituitary	831	40	49	483
	Meningioma	854	104	24	422
	Glioma	715	127	89	474
	No Tumor	718	16	46	624
InceptionV3	Pituitary	751	22	11	620
	Meningioma	758	106	15	525
	Glioma	654	90	4	657
	No Tumor	794	44	6	560

We used 40 epochs with a batch size of 32 for VGG-16. We construct the confusion matrix and assess performance for each class when VGG-16 is completed. Table 2 shows the computed performance, while Fig. 8 shows the accuracy graph and loss.

Table 2. Performance appraisal tables by class for VGG-16

Model	Disease	Accuracy (%)	TPR (%)	FNR (%)	FPR (%)	TNR (%)	Precision (%)	F1 Score (%)
Vgg16	Pituitary	94.94	94.46	5.54	4.28	95.72	97.27	95.84
	Meningioma	95.66	94.74	5.26	3.07	96.93	97.73	96.21
	Glioma	96.08	95.76	4.24	3.54	96.46	96.91	96.33
	No Tumor	95.66	92.68	7.32	1.18	98.82	98.82	95.65

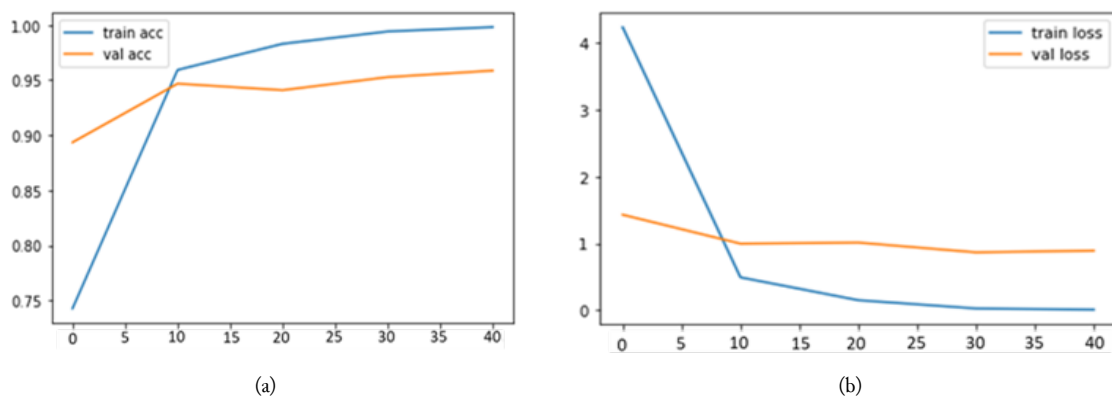


Fig. 8. Diagram for (a) VGG-16 accuracy and (b) VGG-16 loss on 40 epochs

We used 40 epochs and a 32-batch size for VGG-19. We build the confusion matrix from the model after VGG-19 is finished and assess the performance of each class. The computed performance is displayed in Table 3, while the accuracy graph and loss are displayed in Fig. 9.

Table 3. Performance appraisal tables by class for VGG-19

Model	Disease	Accuracy (%)	TPR (%)	FNR (%)	FPR (%)	TNR (%)	Precision (%)	F1 Score (%)
Vgg19	Pituitary	95.09	93.85	6.15	2.75	97.25	98.36	96.05
	Meningioma	94.44	92.24	7.76	2.30	97.70	98.35	95.20
	Glioma	95.58	92.92	7.08	1.61	98.39	98.38	95.57
	No Tumor	97.72	97.53	2.47	1.95	98.05	98.86	98.19

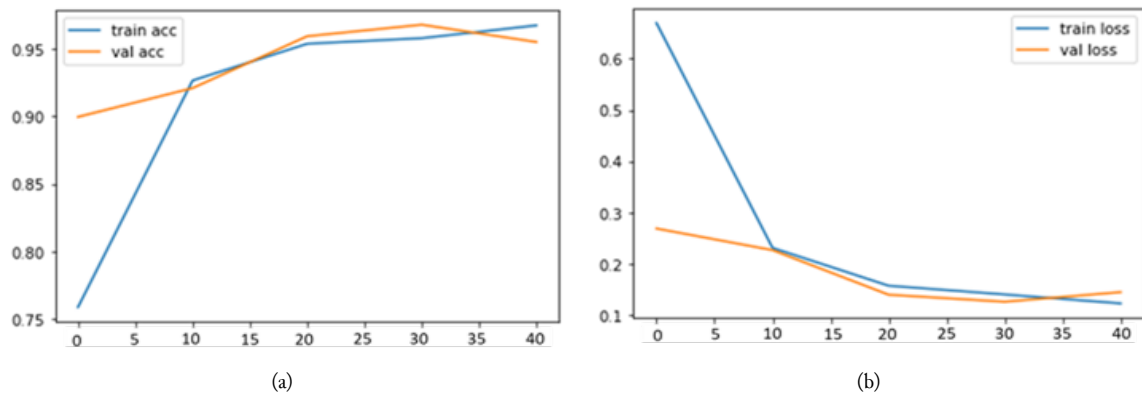


Fig. 9. Diagram for (a) VGG-19 accuracy and (b) VGG-19 loss on 40 epochs

Forty epochs and a batch size of 32 were used for ResNet-50. We build the confusion matrix from the model and assess the performance of each class after ResNet-50 is finished. Fig. 10 depicts the accuracy graph and loss, while Table 4 displays the computed performance.

Table 4. Performance appraisal tables by class for ResNet-50

Model	Disease	Accuracy (%)	TPR (%)	FNR (%)	FPR (%)	TNR (%)	Precision (%)	F1 Score (%)
ResNet-50	Pituitary	97.15	96.96	3.04	2.58	97.42	98.15	97.56
	Meningioma	97.79	98.45	1.55	3.01	96.99	97.56	98.00
	Glioma	95.58	95.38	4.62	4.17	95.83	96.62	96.00
	No Tumor	96.15	95.51	4.49	3.39	96.61	97.05	96.27

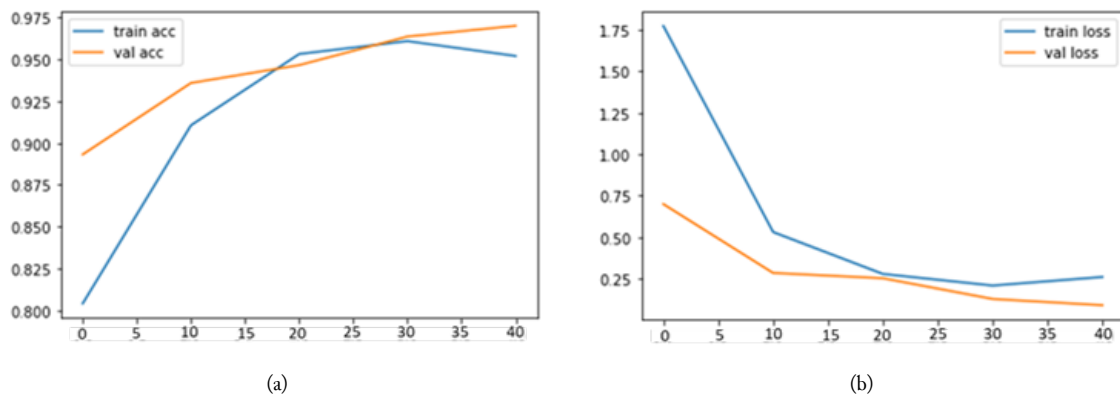


Fig. 10. Diagram for (a) ResNet-50 accuracy and (b) ResNet-50 loss on 40 epochs

For Xception, we utilized 40 epochs and a batch size of 32. When Xception is completed, we create the confusion matrix from the model and evaluate each class's performance. Table 5 shows the computed performance, while Fig. 11 shows the accuracy graph and loss.

Table 5. Performance appraisal tables by class for Xception

Model	Disease	Accuracy (%)	TPR (%)	FNR (%)	FPR (%)	TNR (%)	Precision (%)	F1 Score (%)
Xception	Pituitary	93.62	95.37	4.63	9.24	90.76	94.41	94.88
	Meningioma	90.88	89.11	10.89	5.30	94.70	97.31	93.03
	Glioma	84.64	84.91	15.09	15.76	84.24	88.97	86.89
	No Tumor	95.58	97.85	2.15	6.90	93.10	93.94	95.86

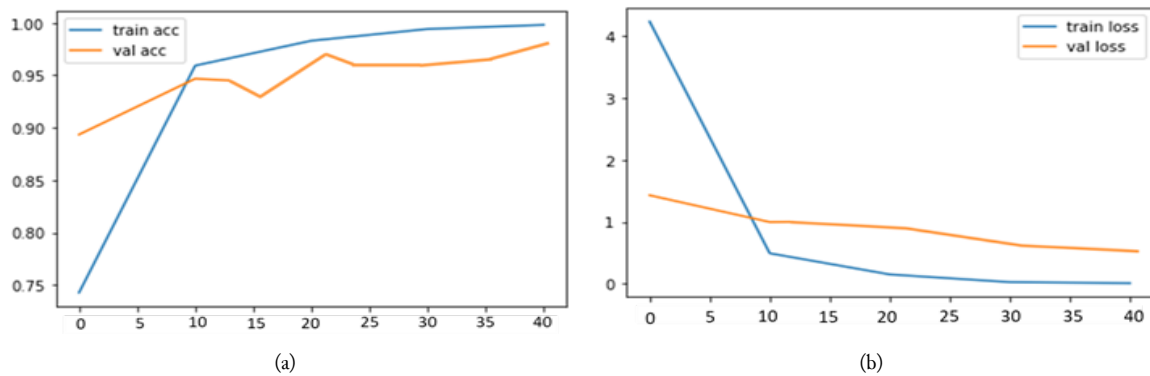


Fig. 11. Diagram for (a) Xception accuracy and (b) Xception loss on 40 epochs

For Inception-V3, we utilized 40 epochs and a batch size of 32. When Inception-V3 is completed, we create the confusion matrix from the model and evaluate each class's performance. Table 6 shows the computed performance, while Fig. 12 shows the accuracy graph and loss.

Table 6. Performance appraisal tables by class for Inception-V3

Model	Disease	Accuracy (%)	TPR (%)	FNR (%)	FPR (%)	TNR (%)	Precision (%)	F1 Score (%)
Inception-V3	Pituitary	97.69	97.20	2.80	1.72	98.28	98.58	97.88
	Meningioma	91.37	87.70	12.30	2.74	97.26	98.09	92.60
	Glioma	93.34	87.95	12.05	0.60	99.40	99.40	93.32
	No Tumor	96.42	94.71	5.29	1.04	98.96	99.26	96.93

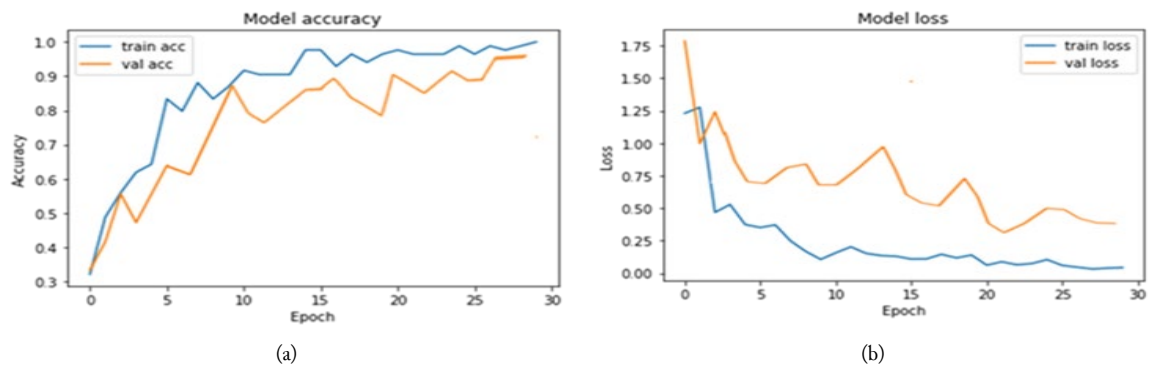


Fig. 12. Diagram for (a) Inception-V3 accuracy and (b) Inception-V3 loss on 40 epochs

In this study, the performance of the trained model was evaluated using a separate test dataset. The dataset used for training the model contained both real and augmented images. The VGG-16, VGG-19,

ResNet-50, Xception, and Inception-V3 architectures were employed to train the model. After training, the model's accuracy was assessed using the test images. The weights of the pre-trained VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3 models were experimented with to compare the model's performance with the other established transfer learning networks. The choice of the pre-trained network that best suited the dataset was evaluated. Table 7 lists the five unique models used in this study, and the comparison diagram for each section of the models is shown separately in Fig. 13.

Table 7. Performance appraisal tables by class for all models

Model	Accuracy	TPR	FNR	FPR	TNR	Precision	F1 Score
VGG-16	95.58	94.41	5.59	3.01	96.99	97.68	96.01
VGG-19	95.71	94.14	5.86	2.15	97.85	98.49	96.25
ResNet-50	96.67	96.58	3.42	3.29	96.71	97.35	96.96
Xception	91.18	91.81	8.19	9.30	90.70	93.66	92.67
Inception-V3	94.71	91.89	8.11	1.52	98.48	98.83	95.19

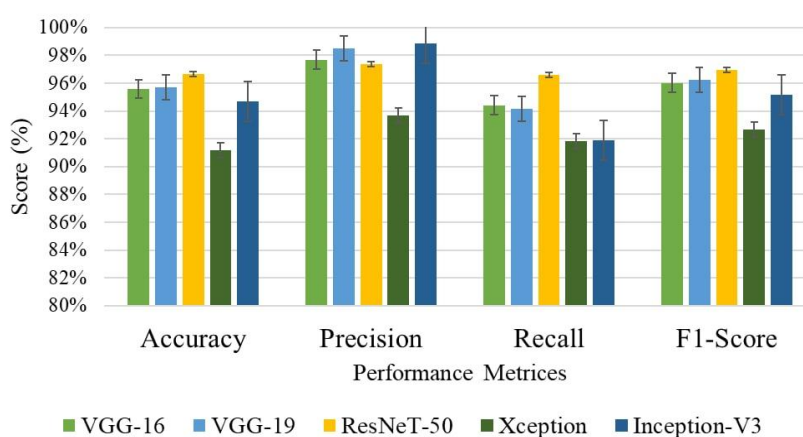


Fig. 13. Overall performance metrics for all applied TLCs

Table 8 compares our observations about detecting brain tumors with those of other writers. Previous studies have shown that raw CNN models and hybrid models can accurately detect particular types of brain tumors from MRI images with a detection rate of 84 to 92%. Comparing our results to those of other authors, we beat the previous study in identifying brain tumors with a 96.67% accuracy using 7138 Images with four groups using ResNet-50.

Table 8. Comparative analysis with previous studies

Research Work	Context	Best Method	Accuracy
Jun Cheng [34]	Brain tumor prediction via tumor region augmentation	BoW-SVM	91.28%
Ismael [35]	Brain tumor classification via statistical features	DWT-Gabor-NN	91.90%
Pashaei [36]	Brain tumor classification	CNN-ELM	93.68%
Abiwinanda [37]	Brain tumor classification	CNN	84.19%
Afshar [38]	Brain tumor classification via coarse tumor boundaries	CapsNet	90.89%
This work	Brain tumor classification via MRI images	ResNet-50	96.67%

4. Conclusion

The brain is one of the body's most intricate systems, with trillions of neurons interacting. The pressure inside the brain increases as a tumor develops in the head, harming the brain. A disease in which abnormal cells proliferate in the human brain is known as an intracranial neoplasm, also called a brain tumor. This work outlines CNN's classification attempts and deep feature extraction on brain tumor MRI recognition using data from Figshare and Kaggle. Here, five well-known deep CNN architectures—the VGG-16, VGG-19, ResNet-50, Xception, and Inception-V3—are used for deep feature extraction and transfer learning. The acquired dataset is precise in experimental work because

many sample photos are available. With a 96.76 percent identification rate for brain tumor MRI, ResNet50 has all models' highest accuracy. We want to increase detection accuracy in the future by utilizing a number of CNN architectural such as pertained model AlexNet, ZfNet model, and a hybrid model with adding a layer or dropping the layers. The limitation of this study recommends that inaccurate brain tumor detection from the MRI picture, which is significant scope, may be corrected in the following stages of this study by studying the non-detection image.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

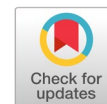
References

- [1] A. M. Rauschecker *et al.*, "Artificial Intelligence System Approaching Neuroradiologist-level Differential Diagnosis Accuracy at Brain MRI," *Radiology*, vol. 295, no. 3, pp. 626–637, Jun. 2020, doi: [10.1148/radiol.2020190283](https://doi.org/10.1148/radiol.2020190283).
- [2] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, "Brain Tumor Segmentation Using Convolutional Neural Networks in MRI Images," *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1240–1251, May 2016, doi: [10.1109/TMI.2016.2538465](https://doi.org/10.1109/TMI.2016.2538465).
- [3] R. R. Laddha and S. A. Ladhake, "A Review on Brain Tumor Detection Using Segmentation And Threshold Operations," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 1, pp. 607–611, 2014. [Online]. Available at : <https://citeseerx.ist.psu.edu/>
- [4] "Brain Tumors - Classifications, Symptoms, Diagnosis and Treatments," *American Association of Neurological Surgeons*. Accessed Apr. 05, 2022. [Online]. Available at: <https://www.aans.org/en/Patients/Neurosurgical-Conditions-and-Treatments/Brain-Tumors>.
- [5] R. Krishna and T. Menzies, "Bellwethers: A Baseline Method for Transfer Learning," *IEEE Trans. Softw. Eng.*, vol. 45, no. 11, pp. 1081–1105, Nov. 2019, doi: [10.1109/TSE.2018.2821670](https://doi.org/10.1109/TSE.2018.2821670).
- [6] D. Theckedath and R. R. Sedamkar, "Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks," *SN Comput. Sci.*, vol. 1, no. 2, p. 79, Mar. 2020, doi: [10.1007/s42979-020-0114-9](https://doi.org/10.1007/s42979-020-0114-9).
- [7] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, vol. 2017-Janua, pp. 1800–1807, doi: [10.1109/CVPR.2017.195](https://doi.org/10.1109/CVPR.2017.195).
- [8] A. Y. Saleh, C. K. Chin, V. Penshie, and H. R. H. Al-Absi, "Lung cancer medical images classification using hybrid CNN-SVM," *Int. J. Adv. Intell. Informatics*, vol. 7, no. 2, p. 151, Jul. 2021, doi: [10.26555/ijain.v7i2.317](https://doi.org/10.26555/ijain.v7i2.317).
- [9] M. Shatara *et al.*, "EPCT-07. Updated report on the pilot study of using MRI-guided laser heat ablation to induce disruption of the peritumoral blood brain barrier to enhance deliver and efficacy of treatment of pediatric brain tumors," *Neuro. Oncol.*, vol. 24, no. Supplement_1, pp. i37–i37, Jun. 2022, doi: [10.1093/neuonc/noac079.135](https://doi.org/10.1093/neuonc/noac079.135).
- [10] S. Deepak and P. M. Ameer, "Brain tumor classification using deep CNN features via transfer learning," in *Computers in Biology and Medicine*, Aug. 2019, vol. 111, p. 103345, doi: [10.1016/j.compbiomed.2019.103345](https://doi.org/10.1016/j.compbiomed.2019.103345).
- [11] M. Talo, U. B. Baloglu, Ö. Yıldırım, and U. Rajendra Acharya, "Application of deep transfer learning for automated brain abnormality classification using MR images," in *Cognitive Systems Research*, May 2019, vol. 54, pp. 176–188, doi: [10.1016/j.cogsys.2018.12.007](https://doi.org/10.1016/j.cogsys.2018.12.007).

- [12] S. Ahuja, B. K. Panigrahi, and T. Gandhi, "Transfer Learning Based Brain Tumor Detection and Segmentation using Superpixel Technique," in *2020 International Conference on Contemporary Computing and Applications (IC3A)*, Feb. 2020, pp. 244–249, doi: [10.1109/IC3A48958.2020.233306](https://doi.org/10.1109/IC3A48958.2020.233306).
- [13] M. A. Khan *et al.*, "Multimodal Brain Tumor Classification Using Deep Learning and Robust Feature Selection: A Machine Learning Application for Radiologists," *Diagnostics*, vol. 10, no. 8, p. 565, Aug. 2020, doi: [10.3390/diagnostics10080565](https://doi.org/10.3390/diagnostics10080565).
- [14] T. Kaur and T. K. Gandhi, "Deep convolutional neural networks with transfer learning for automated brain image classification," *Mach. Vis. Appl.*, vol. 31, no. 3, p. 20, Mar. 2020, doi: [10.1007/s00138-020-01069-2](https://doi.org/10.1007/s00138-020-01069-2).
- [15] C. Srinivas *et al.*, "Deep Transfer Learning Approaches in Performance Analysis of Brain Tumor Classification Using MRI Images," in *Journal of Healthcare Engineering*, Mar. 2022, vol. 2022, pp. 1–17, doi: [10.1155/2022/3264367](https://doi.org/10.1155/2022/3264367).
- [16] Xiaoling Xia, Cui Xu, and Bing Nan, "Inception-v3 for flower classification," in *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, Jun. 2017, pp. 783–787, doi: [10.1109/ICIVC.2017.7984661](https://doi.org/10.1109/ICIVC.2017.7984661).
- [17] A. K. Bitto and I. Mahmud, "Multi categorical of common eye disease detect using convolutional neural network: a transfer learning approach," *Bull. Electr. Eng. Informatics*, vol. 11, no. 4, pp. 2378–2387, Aug. 2022, doi: [10.11591/eei.v11i4.3834](https://doi.org/10.11591/eei.v11i4.3834).
- [18] A. Wadhwa, A. Bhardwaj, and V. Singh Verma, "A review on brain tumor segmentation of MRI images," in *Magnetic Resonance Imaging*, Sep. 2019, vol. 61, pp. 247–259, doi: [10.1016/j.mri.2019.05.043](https://doi.org/10.1016/j.mri.2019.05.043).
- [19] "Brain Tumor MRI Dataset," *Kaggle*. Accessed Apr. 24, 2022. [Online]. Available at: <https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset>.
- [20] J. Mia, H. I. Bijoy, S. Uddin, and D. M. Raza, "Real-Time Herb Leaves Localization and Classification Using YOLO," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Jul. 2021, pp. 1–7, doi: [10.1109/ICCCNT51525.2021.9579718](https://doi.org/10.1109/ICCCNT51525.2021.9579718).
- [21] A. W. Reza, J. F. Sorna, M. M. U. Rashed, and M. M. A. Shibly, "ModCOVNN: a convolutional neural network approach in COVID-19 prognosis," *Int. J. Adv. Intell. Informatics*, vol. 7, no. 2, p. 125, Apr. 2021, doi: [10.26555/ijain.v7i2.604](https://doi.org/10.26555/ijain.v7i2.604).
- [22] H. Ali Khan, W. Jue, M. Mushtaq, and M. Umer Mushtaq, "Brain tumor classification in MRI image using convolutional neural network," *Math. Biosci. Eng.*, vol. 17, no. 5, pp. 6203–6216, 2020, doi: [10.3934/mbe.2020328](https://doi.org/10.3934/mbe.2020328).
- [23] S. Hasan, G. Rabbi, R. Islam, H. Imam Bijoy, and A. Hakim, "Bangla Font Recognition using Transfer Learning Method," in *2022 International Conference on Inventive Computation Technologies (ICICT)*, Jul. 2022, pp. 57–62, doi: [10.1109/ICICT54344.2022.9850765](https://doi.org/10.1109/ICICT54344.2022.9850765).
- [24] V. Rajinikanth, A. N. Joseph Raj, K. P. Thanaraj, and G. R. Naik, "A Customized VGG19 Network with Concatenation of Deep and Handcrafted Features for Brain Tumor Detection," *Appl. Sci.*, vol. 10, no. 10, p. 3429, May 2020, doi: [10.3390/app10103429](https://doi.org/10.3390/app10103429).
- [25] A. Pramanik, M. H. I. Bijoy, and M. S. Rahman, "Detection of Potholes using Convolutional Neural Network Models: A Transfer Learning Approach," in *2021 IEEE International Conference on Robotics, Automation, Artificial-Intelligence and Internet-of-Things (RAAICON)*, Dec. 2021, pp. 73–78, doi: [10.1109/RAAICON54709.2021.9929623](https://doi.org/10.1109/RAAICON54709.2021.9929623).
- [26] P. Harish and S. Baskar, "MRI based detection and classification of brain tumor using enhanced faster R-CNN and Alex Net model," *Mater. Today Proc.*, Dec. 2020, doi: [10.1016/j.matpr.2020.11.495](https://doi.org/10.1016/j.matpr.2020.11.495).
- [27] D. Hirahara, "Preliminary assessment for the development of CAde system for brain tumor in MRI images utilizing transfer learning in Xception model," in *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, Oct. 2019, pp. 922–924, doi: [10.1109/GCCE46687.2019.9015529](https://doi.org/10.1109/GCCE46687.2019.9015529).
- [28] N. Noreen, S. Palaniappan, A. Qayyum, I. Ahmad, M. Imran, and M. Shoaib, "A Deep Learning Model Based on Concatenation Approach for the Diagnosis of Brain Tumor," in *IEEE Access*, 2020, vol. 8, pp. 55135–55144, doi: [10.1109/ACCESS.2020.2978629](https://doi.org/10.1109/ACCESS.2020.2978629).

- [29] M. P. Mahmud, M. A. Ali, S. Akter, and M. H. I. Bijoy, "Lychee Tree Disease Classification and Prediction using Transfer Learning," in *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Oct. 2022, pp. 1–7, doi: [10.1109/ICCCNT54827.2022.9984286](https://doi.org/10.1109/ICCCNT54827.2022.9984286).
- [30] M. J. Mia, S. K. Maria, S. S. Taki, and A. A. Biswas, "Cucumber disease recognition using machine learning and transfer learning," *Bull. Electr. Eng. Informatics*, vol. 10, no. 6, pp. 3432–3443, Dec. 2021, doi: [10.11591/eei.v10i6.3096](https://doi.org/10.11591/eei.v10i6.3096).
- [31] M. M. Fouad, E. M. Mostafa, and M. A. Elshafey, "Detection and localization enhancement for satellite images with small forgeries using modified GAN-based CNN structure," *Int. J. Adv. Intell. Informatics*, vol. 6, no. 3, p. 278, Nov. 2020, doi: [10.26555/ijain.v6i3.548](https://doi.org/10.26555/ijain.v6i3.548).
- [32] M. R. Mia, M. J. Mia, A. Majumder, S. Supriya, and M. T. Habib, "Computer Vision Based Local Fruit Recognition," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 1, pp. 2810–2820, Oct. 2019, doi: [10.35940/ijeat.A9789.109119](https://doi.org/10.35940/ijeat.A9789.109119).
- [33] J. S. Murugaiyan, M. Palaniappan, T. Durairaj, and V. Muthukumar, "Fish species recognition using transfer learning techniques," *Int. J. Adv. Intell. Informatics*, vol. 7, no. 2, p. 188, Jul. 2021, doi: [10.26555/ijain.v7i2.610](https://doi.org/10.26555/ijain.v7i2.610).
- [34] J. Cheng *et al.*, "Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition," *PLoS One*, vol. 10, no. 10, p. e0140381, Oct. 2015, doi: [10.1371/journal.pone.0140381](https://doi.org/10.1371/journal.pone.0140381).
- [35] M. R. Ismael and I. Abdel-Qader, "Brain Tumor Classification via Statistical Features and Back-Propagation Neural Network," in *2018 IEEE International Conference on Electro/Information Technology (EIT)*, May 2018, vol. 2018-May, pp. 0252–0257, doi: [10.1109/EIT.2018.8500308](https://doi.org/10.1109/EIT.2018.8500308).
- [36] A. Pashaei, H. Sajedi, and N. Jazayeri, "Brain Tumor Classification via Convolutional Neural Network and Extreme Learning Machines," in *2018 8th International Conference on Computer and Knowledge Engineering (ICCKE)*, Oct. 2018, pp. 314–319, doi: [10.1109/ICCKE.2018.8566571](https://doi.org/10.1109/ICCKE.2018.8566571).
- [37] N. Abiwinanda, M. Hanif, S. T. Hesaputra, A. Handayani, and T. R. Mengko, "Brain Tumor Classification Using Convolutional Neural Network," in *IFMBE Proceedings*, vol. 68, no. 1, Springer Verlag, 2019, pp. 183–189, doi: [10.1007/978-981-10-9035-6_33](https://doi.org/10.1007/978-981-10-9035-6_33).
- [38] P. Afshar, K. N. Plataniotis, and A. Mohammadi, "Capsule Networks for Brain Tumor Classification Based on MRI Images and Coarse Tumor Boundaries," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, vol. 2019-May, pp. 1368–1372, doi: [10.1109/ICASSP.2019.8683759](https://doi.org/10.1109/ICASSP.2019.8683759).

Image contrast enhancement for preserving entropy and image visual features



Bilal Bataineh ^{a,1,*}

^aInformation Systems Department, College of Computers and Information Systems, Umm Al-Qura University, Makkah, Saudi Arabia

¹ bmbataineh@uqu.edu.sa

* corresponding author

ARTICLE INFO

Article history

Received August 29, 2022

Revised January 21, 2023

Accepted February 7, 2023

Available online April 7, 2023

Keywords

Entropy

Histogram equalization

Image processing

Image enhancement

Contrast enhancement

ABSTRACT

Histogram equalization is essential for low-contrast enhancement in image processing. Several methods have been proposed; however, one of the most critical problems encountered by existing methods is their ability to preserve information in the enhanced image as the original. This research proposes an image enhancement method based on a histogram equalization approach that preserves the entropy and fine details similar to those of the original image. This is achieved through proposed probability density functions (PDFs) that preserve the small gray values of the usual PDF. The method consists of several steps. First, occurrences and clipped histograms are extracted according to the proposed thresholding. Then, they are equalized and used by a proposed transferring function to calculate the new pixel values in the enhanced image. The proposed method is compared with widely used methods such as Clahe, CS, HE, and GTSHE. Experiments using benchmark datasets and entropy, contrast, PSNR, and SSIM measurements are conducted to evaluate the performance. The results show that the proposed method is the only one that preserves the entropy of the enhanced image of the original image. In addition, it is efficient and reliable in enhancing image quality. This method preserves fine details and improves image quality, supporting computer vision and pattern recognition fields.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Many image-capturing conditions cause low-quality images, noise, blurring, as well as incorrect contrast and brightness levels [1]–[4]. Therefore, considerable research interest exists for improving the quality of images to their advantage. Image enhancement techniques seek to improve the quality of the visual appearance of images to increase their effectiveness for human vision, or for image processing, pattern recognition, and computer vision applications [2], [4]–[10].

Contrast enhancement is an essential branch of image enhancement. It aims to improve the quality of the visual appearance of a low-contrast image [2], [11]–[15]. Contrast enhancement may be categorized into genetic algorithms, fuzzy set algorithms, or histogram equalization (HE) algorithms [15]–[20]. One of the most widespread techniques for low contrast enhancement is the HE [5], [15]–[19]. Several works claimed that the most widespread technique for low contrast enhancement is HE because of its simplicity and easy implementation, as well as its efficiency with most kinds of images [21]–[23]. In general, all methods that adopted the HE principle enhance the poor contrast by remapping the cumulative distribution histogram of the pixels' density values to the full range according to their probabilities [21]–[23]. The methods in this category are important preprocessing stages in many computer vision applications [6], [20], [24]–[26].

HE methods can effectively overcome the problem of low contrast low brightness. However, several approaches have been proposed for a specific type of images, and these methods fail to generalize for wide types of different images [27]–[29]. Moreover, numerous problems occurred, such as over-enhancement, noises, poor brightness, and the loss of original information and fidelity state from enhanced images [27]–[29]. These techniques also overlook the important point of preserving the original information and details in enhanced images for computer vision applications. It's important to note show that the principle of HE allows the gray levels that have high probability density values to devour the associated gray values with a small probability. This situation causes the loss of some gray levels of the original information. By contrast, the local approach of HE methods generates extra gray values, a feature that leads to additional unwanted information in the enhanced images. The optimal methods are those that produce an optimum enhanced image with similar information content as those in the original images. State-of-the-art techniques are mainly concerned with enhancing the visual appearance rather than preserving the original information content [27]–[29].

In the literature, several HE methods were proposed to overcome the challenges of low contrast enhancement [14], [20], [30]–[34]. The method is the basic and the most common approach that uses a probability density function (pdf) and different cumulative distribution functions (cdf) to transform gray values into new distributions [35]. This approach is the foundation of many subsequently developed methods. However, it produces unwanted side effects with a wide type of images. The HE results improve by applying preprocessing steps on the original image and then applying HE [36]. This technique gives better results than the basic HE [35], but many problems still appear.

Many methods tried to improve the HE performance by using the local approach which based on dividing the histogram to many parts and equalizing each part individually. In Yoon *et al.* [37], the histogram is divided into sub-histograms according to brightness and then each individual sub-histogram is equalized by extending it to its range limits. This method overcomes the over-enhancement or under-enhancement issues. However, its results are not satisfactory with wide types of images. The dualistic sub-image histogram equalization (DSIHE) method [38] improves the contrast and keeps the image details. Initially, the median of the gray-values is used as the threshold to divide the histogram into two sub-histograms and remap each part independently. This approach preserves the information and brightness of the result as comparable to those of the original images. However, this method suffers from the over-enhancement problem and produces noises.

Two methods were presented by the same researchers [39], [40] called the minimum mean brightness error bi-histogram equalization method (MMBEBHE) and the recursive mean-separate histogram equalization (RMSHE), respectively. Both methods divide the image into two parts according to the minimum mean brightness error and then enhances each part individually. The results enhanced the contrast and maintained the optimal brightness of the original images. However, both methods lose information from the original images. The MMBEBHE method is also slow, and the RMSHE is complex to develop. An improvement of the RMSHE called the (recursive sub-image histogram equalization (RSIHE) [41] used the mean value of gray levels to divides the image into two sub-histograms. In addition to unwanted side effects, the RSIHE is time consuming.

The exposure-based sub-image histogram equalization (ESIHE) [42] aims to preserve the brightness after enhancement processes. In the ESIHE method, the histogram is divided into two parts by using a proposed threshold technique. Then, each part is separately enhanced by applying HE. This approach preserves the image brightness but suffers from over-enhancement and under-enhancement problems. The dynamic histogram specification (DHS) proposed [40] has no side effects but does not achieve satisfactory enhancement with many images.

The radiance indicator-based histogram equalization for retinal vessel enhancement method is proposed for retinal images [43]. The histogram is divided into sub-histograms according to multi-threshold levels. Then, each sub-histogram is equalized separately. The results of this method for medical images were satisfactory, but the technique failed to produce acceptable results with other types of images. For the proposed mean and variance-based sub-image histogram equalization (MVS IHE) [44],

the histogram is divided into four sub-histograms that are equalized individually. This approach has high accuracy, but the images lost some details in some image cases.

The approach of bi-histogram equalization using modified histogram bins [45] was proposed to enhance contrast for narrow range brightness images. However, it produces low-quality contrast encasement. The adaptive histogram equalization algorithm (AHEA) is proposed by using the entropy to organize the gaps between the gray levels to identify the histogram [37]. The AHEA method affected image enhancement but produced over-enhancement or under-enhancement in some images.

The exposure region-based multi-histogram equalization method was proposed to enhance images with uneven illumination [23]. First, histograms were divided by using a region-based thresholding method. Then, the entropy of the gray level was used to reshape the new sub-histograms. This method suitably enhanced images while maintaining their original appearance. However, its results are unsatisfactory for many brightness cases. The global two stages histogram equalization (GTSHE) [46] depends on several steps. In the beginning, the clipped histogram must equalize according to the occurrences of pixel values. Then, equalizing processes are conducted according to the available and missing pixel level occurrences. The results of this method show high performance in terms of information and details preservation. However, its accuracy needs to be better with many cases of images.

Kandhway *et al.* [20] proposed the method that aims to find the optimal contrast of the magnetic resonance imaging. In this method, the minimum, maximum, mean, and median of the histogram are used to clip its values. Then, several methods are applied to reset the levels in the new histograms. This approach achieves excellent performance for medical images. However, the technique is complex to apply and is dependent on the accuracy of the methods employed in its development. Moreover, experimenting with other exciting methods shows a better result than this technique. A set of filters for enhancement, a binary tree structure to remap the grayscale, and suppressing artifacts have been proposed in Xiong *et al.* [19]. The resulting approach is simple and easy to implement. However, it requires modifying a parameter to obtain the required results, and its outcome for a wide range of images is unsatisfactory. The krill herd (KH) [20] method aims to enhance medical images based on two proposed functions; the Plateau limit function to clip the histogram and fitness function to improved different characteristic information of the anatomical images. This method improves the visual appearance of images effectively. However, it is failed to generalize for other types of images, and it didn't preserve the original entropy.

Several methods have been proposed to overcome the problem of poor contrast. Only the DSIHE method [38] explored the entropy conservation in the processed images, but its results are not satisfactory. Moreover, many of these methods have been proposed for specific purposes such as [4], [6], [9], [12], [29]. These methods fail to generalize to a wide variety of different images as claimed [23]. The proposed methods suffer from over-enhancement, under-enhancement, noise, and uneven brightness problems, as well as loss of original information from the enhanced images. These methods also ignore the point of preserving the original information, entropy, and fine details in the enhanced images.

Accordingly, the motivation of this study is to propose an adaptive HE method for gray images that enhances the poor contrast and preserves the original information (entropy) effectively. This is achieved by using the equal probability of occurrences to preserve the gray values of small values in the probability density function (pdf). The proposed approach extracts the occurrences and histogram of the gray values in the original image. Next, the histogram is clipped according to suggested thresholding method that depends on the histogram and occurrence values. Then, the occurrences and clipped histogram are equalized and used by a proposed transferring function to calculate the new gray values for enhanced images. The performance of the proposed method is compared against popular and recent contrast enhancement methods including the adaptive histogram equalization (Clahe) [47], CS [29], HE [35][28], and GTSHE [46] methods. Experiments were conducted using different benchmark datasets. The entropy measurement was used to evaluate the amount of information in the image. In addition to the contrast ratio, the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) measurements were employed to evaluate the quality of the enhanced images. The method is simple to implement for any type of image.

The contributions of this work are summarized as follows:

- To propose an HE method that takes equal effects for each pixel value present instead of different effects based on the percentage of each pixel value in the enhancement process.
- To propose an enhancement method that preserves fine detail and entropy in images.
- To propose an image enhancement method that can improve the quality of visual detail better than existing methods.

The remainder of this paper is organized as follows: Section 2 explores the proposed method, Section 3 presents the experiments and results, and Section 4 provides a discussion of the results. Finally, Section 5 provides the conclusion of this work.

2. Method

In this section, the proposed method is explained in detail. The flowchart of the steps of proposed method is presented in Fig. 1, and the description is presented below.

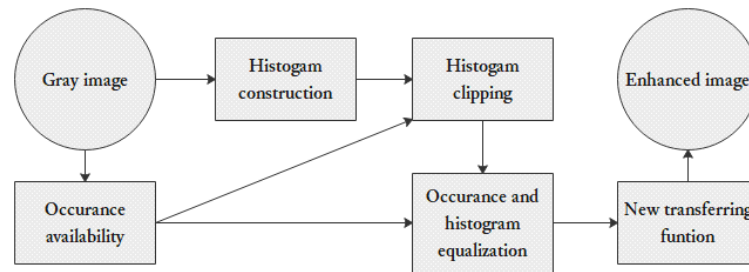


Fig. 1. The flowchart of the proposed method

2.1. Occurrence availability

In this step, the occurrence values are extracted from the input original image (l_o). Each value in the gray-scale range will be checked if it is existing in the original image or not, and the new gray values are detected. This aspect will play a major role in preserving the entropy and the image details in the subsequent enhancement steps. Thus, Equation (1) extracts the occurrences in the original image:

$$Occ(l) = \begin{cases} 1: & l \text{ is in } l_o \\ 0: & l \text{ is not in } l_o \end{cases} \text{ for each Pixel in } l_o \quad (1)$$

where the $Occ(l)$ is an occurrence array, and its size is in the maximum gray-level value (here, its size is 255). The variable l denotes each gray value in the image (0 to 255). Thus, each cell in the array Occ will be coded as 1 if the value is available in the original image and 0 if otherwise.

2.2. Histogram construction

Parallel to the occurrence availability extraction, the frequency of each gray value of the input image is extracted. That represented by constructing the histogram of the gray-values which normally range in (0 – 255). Thus, let His is the accumulator histogram of each gray level value in the image. The accumulator histogram is extracted according to Equation (2).

$$His(l) = His(l) + 1: \text{ for each } l \text{ in } l_o, \quad (2)$$

2.3. Histogram clipping

Usually, frequency variance occurs between gray values in images. This situation leads to huge variance between the accumulator values of the histogram, thereby causes many challenges in the contrast enhancement process. To overcome these challenges, the extreme affection of high values in the histogram is minimized to reduce their over affection and obtain a neutral appearance of the enhanced

images. Accordingly, the histogram itself must be clipped [35], [19] so as to reduce the large values in suitable form to other accumulator values in the histogram.

Thresholding is the most popular process in state-of-the-art methods to solve this problem [1]. Several techniques have been employed to calculate the clipping thresholds. In this work, a new thresholding method is proposed to calculate a more optimal clipping threshold. Equation (3) presents the proposed thresholding method which based on the histogram values and available occurrences of the image. The suggested approach is effective and consumes less time [19],[35],[36].

$$T = \frac{1}{\#Occ} \sum_{l=0}^L His(l), \quad (3)$$

$$H_c(l) = T : \text{for } His(l) > T, \quad (4)$$

where T is the clipping threshold value, $\#Occ$ is the number of non-zero occurrences in the image, and H_c is the clipped histogram. The clipped histogram has fewer values relative to the original image. The high histogram values were reduced, and the low histogram values are preserved.

2.4. Equalization process

Next, the clipped histogram and occurrence matrix are equalized to produce affected new values and show the new values of pixels in the enhanced images. For effective equalization, we proposed a new equalization process that considers the state of available occurrence of gray-level values in addition to the pixel values and their frequency, unlike previous methods that consider only the pixel values and the probability of their frequency.

Accordingly, the corresponding pdf of the clipping histogram Pdf_c was computed as follows:

$$Pdf_c(l) = \frac{H_c(l)}{\sum H_c(l)}. \quad (5)$$

To add the advantage of equal probability for each available value in the image (which will be used later to increase the probability of small frequent gray-value), the corresponding probability density of occurrences is used to preserve the small accumulate histogram values from merging by associated large accumulate. The corresponding pdf of occurrences in the original image Pdf_{occ} was computed as follows:

$$Pdf_{occ}(l) = \frac{occ(l)}{\#Occ} \quad (6)$$

Next, two cumulative distribution functions (cdf) of both the corresponding pdf of the clipping histogram Pdf_c and the occurrences of each gray-value in the original image Pdf_{occ} are calculated as follows:

$$Cdf_{H_c}(l) = \sum_{l=0}^{255} Pdf_c(l). \quad (7)$$

$$Cdf_{occ}(l) = \sum_{l=0}^{255} Pdf_{occ}(l). \quad (8)$$

The results of the each of previous equations are an array in size 255. Each array has a cumulative distribution function value ranging from 0 to 1.

2.5. Transferring function

In this method, a new transferring function is proposed to identify the optimal new gray values by using each $Cdf_{H_c}(l)$ and $Cdf_{occ}(l)$ to combine their advantages. The proposed cumulative distribution function for the available gray values improves their small values of a traditional pdf and preserves them from disappearing by the large ones. That aim is achieved by adding the probability of each available pixel value in $Cdf_{H_c}(l)$ and $Cdf_{occ}(l)$. The results generate a probability value from 0 to 2. To arrive the final transferred pixel value in the range of 0 to 255, we multiply each probability value in (127.5). The results include new pixel values of the enhanced image in the range of 0 to 255.

$$image_E = (Cdf_{occ}(l) + Cdf_{Hc}(l)) \times 127.5. \quad (9)$$

Finally, the gray values of the enhanced image pixels are integrated and presented. All the pixels values will be represented again in the enhanced images. The enhanced images will contain neither fewer nor additional occurrences. Algorithm 1 presents all the steps of the proposed method (Fig. 2).

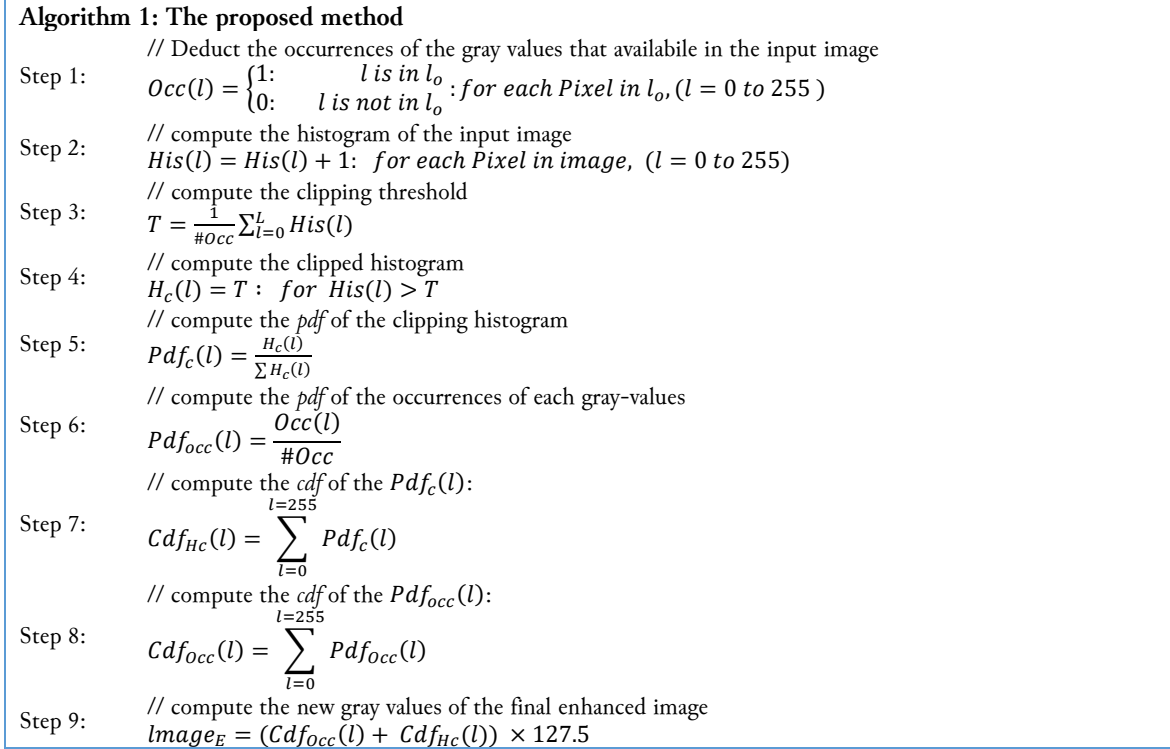


Fig. 2. Proposed algorithm

3. Results and Discussion

After the proposed method is developed, several experiments were conducted to evaluate its performance. The evaluation process according to a comparison of the performance of the proposed and popular benchmark histogram equalization methods. According to the state-of-the-art knowledge, the original HE method [35] is the basic principle of the histogram equalization technique and most of the next methods are improvements on it. The contrast stretching and Clahe methods [47] are popular and high-performance techniques that are prevalent in computer vision libraries. GTSHE [46], [36] is a recent method that shows a high performance relative to several state-of-the-art methods such as the MVSHE [15], AHEA [37], ESIHE [35] techniques. These methods outperformed many well-known approaches such as the HE [35], DSIHE [31], RMSHE [32], MMBEBHE [33], RSIHE [34] methods. Accordingly, the HE [35], contrast stretching, Clahe [47], and GTSHE [46], [36] techniques were selected to evaluate the performance of the proposed method.

To obtain meaningful results that show the real performance of each of the methods, experiments were conducted on several benchmark images. The used images include 46 aerial images, 39 miscellaneous images of the database of the University of Southern California, and 100 images from the CG-1050 dataset. In total, approximately 185 different images that cover many types of shapes, object, sizes, and contrast qualities were used in the evaluation experiments.

3.1. Visual experiments

Visual experiments were conducted for a clear assessment of the methods. Fig. 3 to Fig. 7 show some of the outputs visually. These experiments were conducted on selected images to demonstrate the ability of each method to improve the appearance by optimizing the contrast.

The visual results in Fig. 3 to Fig. 7 indicate that each method shows various forms of contrast enhancements. The first image (Fig. 3) suffers from the low contrast problem. The proposed method and GTSHE technique show balanced performance. HE and contrast stretching tends to present darkness that removes some details from the image. The Clahe did not show any enhancement.

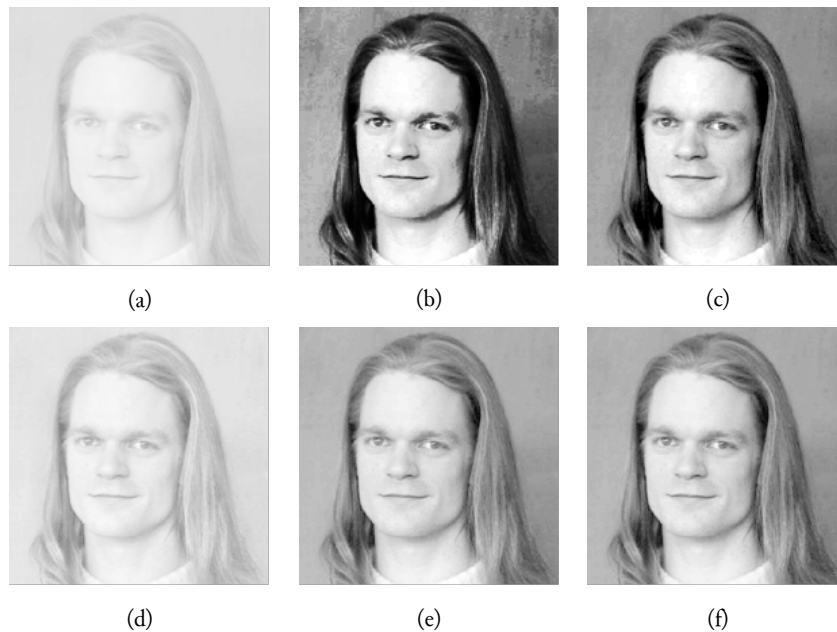


Fig. 3. (a) Original image, a fighter jet image (source: NumPy / SciPy Recipes for Image Processing: Intensity Normalization and Histogram Equalization), (b) HE, (c) Contrast stretching, (d) Clahe, (e) GTSHE, and (f) the proposed methods.

In the case of the fighter jet image (Fig. 4), the problem of tight range contrast of gray-level values is enhanced perfectly with the proposed method. The GTSHE approach generates a satisfactory result but tends exhibit over brightness. The HE and contrast stretching methods achieved low enhancement, and no enhancement is shown by Clahe. The poor-quality contrast aerials image in Fig. 5 is enhanced well with all methods except HE, but the proposed and GTSHE methods show better performance.

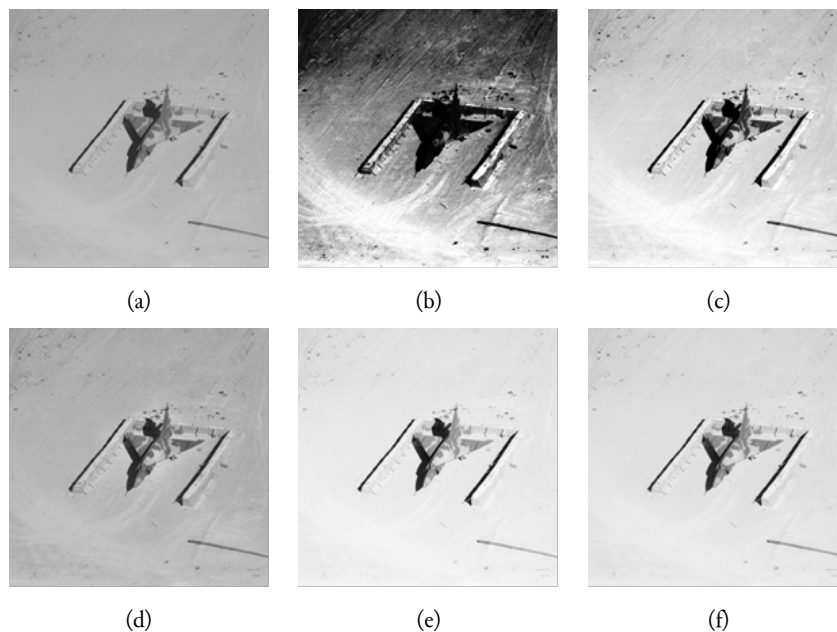


Fig. 4. (a) Original image, a fighter jet image (miscellaneous dataset, 7.1.02 image), (b) HE, (c) Contrast stretching, (d) Clahe, (e) GTSHE, and (f) the proposed methods.

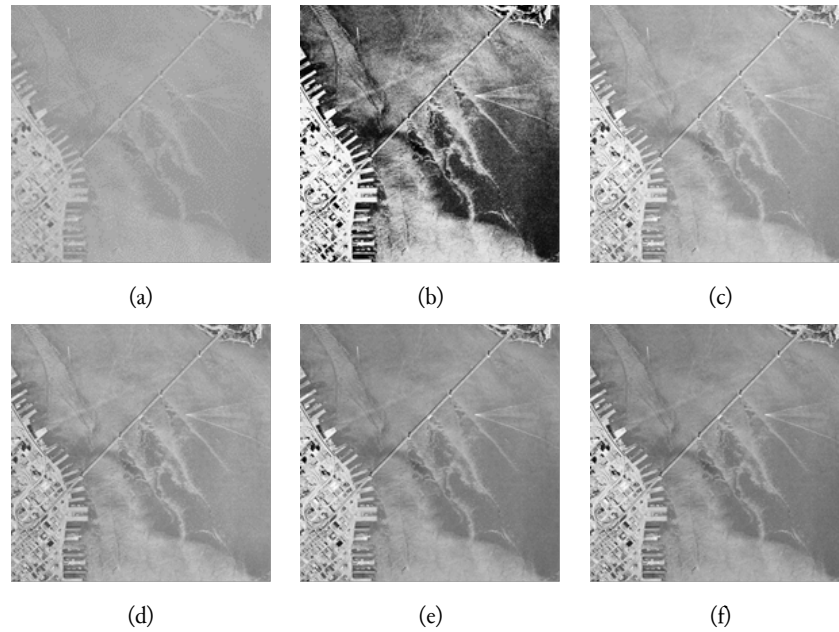


Fig. 5. (a) Original image, an aerial image (aerials dataset, 2.2.06 image), (b) HE, (c) Contrast stretching, (d) Clahe, (e) GTSHE, and (f) the proposed methods.

To show adaptivity, the focal methods were applied on an extremely large and good quality contrast image (Fig. 6). The proposed method, GTSHE, and contrast stretching show minor changes and preserved the original quality. HE and Clahe show major changes and the over-enhancement problem with fine images.

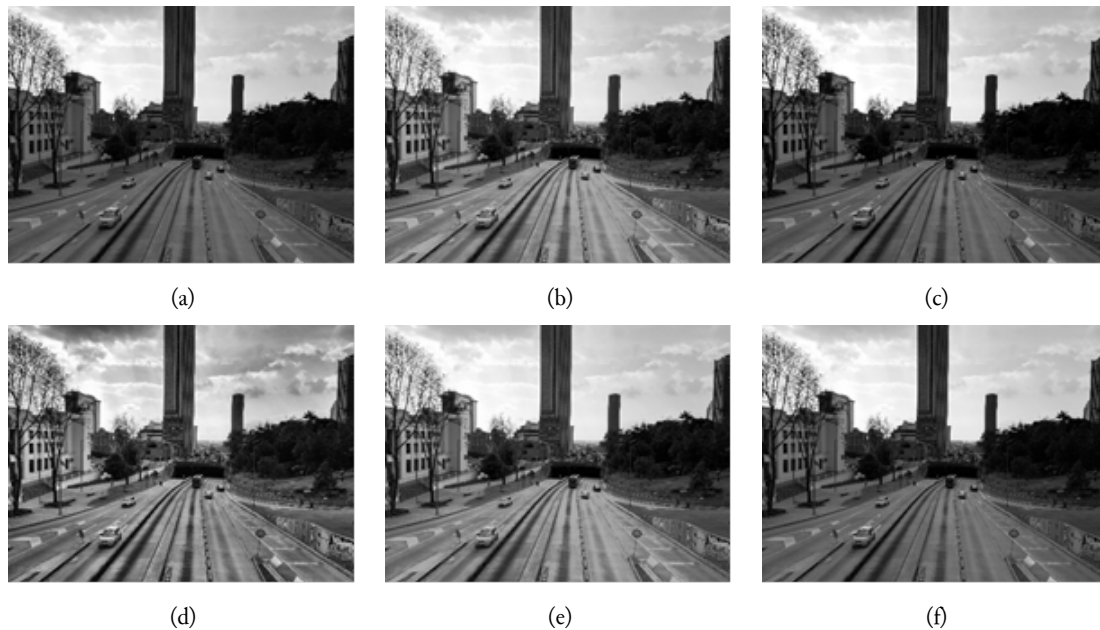


Fig. 6. (a) Original image, the Im_78 image from CG-1050 dataset, (b) HE, (c) Contrast stretching, (d) Clahe, (e) GTSHE, and (f) the proposed methods.

Applied to Lena image (Fig. 7), the HE, proposed approach, and GTSHE method show a good performance. To show the results clearly, zooming was applied. The proposed method shows more harmonic contrast than its counterparts which, in turn, reveal some rapid changes between the enhanced gray-scale levels.

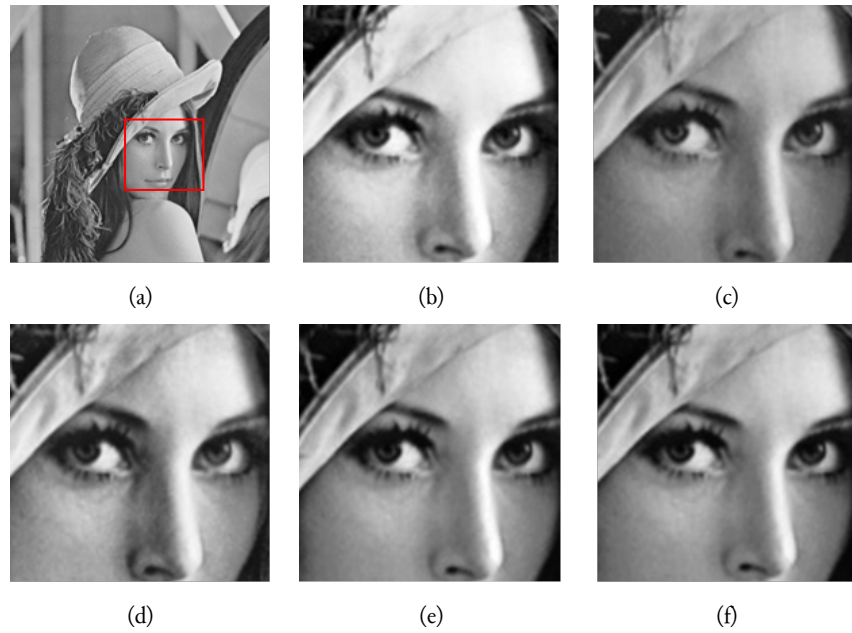


Fig. 7. (a) Original image, a focus on sub-image of Lena, (b) HE, (c) Contrast stretching, (d) Clahe, (e) GTSHE, and (f) the proposed methods.

3.2. Statistical experiments

Visual experiences are clearly understandable to humans, but they do not constitute a scientific process for accurately and effectively evaluating the performance of the methods involved. This approach is not an analytical scientific measurement technique, and its estimation is not based on any meaningful measurements. Moreover, humans may not recognize some problems and noise. To overcome these features, this work performs an evaluation experiment using previously mentioned benchmark datasets and statistical measurements that involve the average information content (entropy), contrast, PSNR, and SSIM.

Entropy presents the average information content in an image. It is a popular measurement for evaluating the amount of information in the image. In this work the term entropy is the main measurement used to evaluate the main objective of this work. Its formula presented in an equation (10):

$$Entropy = \sum_{l=0}^{255} e(l) = - \sum_{l=0}^{255} p(l) \log_2 p(l) \quad (10)$$

Where $p(l)$ is the probability of the gray value l in the image.

The contrast ratio presents the difference in the gray values between occurrences in the image. In low contrast states, the contrast ratio is small. It is presented by calculating the standard division value of pixels in the image.

The PSNR is a widely use measurement approach to evaluate image quality and signals in general. In this case, it provides a value about the contrast that shows the enhancement quality of the processed image by the original image. The following formula presents the of PSNR equation (11):

$$PSNR = 10 \log_{10} \left(\frac{255^2}{MSE} \right) \quad (11)$$

$$MSE = \frac{1}{(X \times Y)} \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} (Original_{image}(x, y) - Processed_{image}(x, y))^2 \quad (12)$$

where the size of image presented in X and Y , and the current positions of the pixel are presented by x and y .

The SSIM measured the quality of images by estimating the similarity between two images. The SSIM considers structural information changes such as contrast in the resulting image. The SSIM is computed according to different sub-images, the measure between the two sub-images x and y of image in size X×Y are presented by the following equation:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)} \quad c_1 = (0.01 \times L)^2, c_2 = (0.03 \times L)^2 \quad (13)$$

where L is the dynamic range of the pixel-values, and μ_x, μ_y are the mean of x and y. σ_x^2, σ_y^2 is the variance of x and y, σ_{xy} the covariance of x and y.

Table 1 indicates that the entropy and the absolute changes in the entropy among the methods and the original images. The proposed method is the optimal approach in preserving image information. The said method has entropy values like those of the original images and shows no difference between the original and enhanced images for all datasets.

Table 1. The entropy and absolute difference in entropy values between the original images and enhanced images of the Cleha [47], CS [29], HE [35], GTSHE [46], and proposed methods for the aerials, CG-1050 dataset, and Miscellaneous images.

	Aerials		CG-1050 dataset		Miscellaneous	
	Entropy	abs(Difference)	Entropy	abs(Difference)	Entropy	abs(Difference)
Original	11.91	0	7.13	0	10.33	0
Cleha[47]	13.82	1.91	8.84	1.71	12.02	1.69
CS[29]	13.12	1.21	6.93	0.2	10.43	0.1
HE [35]	11.75	0.16	6.99	0.14	10.04	0.29
GTSHE [7]	11.9	0.01	7.12	0.01	10.29	0.04
Proposed	11.91	0	7.13	0	10.32	0.01

Conversely, the other methods have lower or higher entropy values (Fig. 8). Thus, each involved method lost information of or added noises to the enhanced images. As shown in Table 1 and Fig. 8, the Cleha method performed satisfactorily in visual experiments but was worst in the case of preserving original information in processed images. GTSHE is the second-best approach but losses some information in all datasets.

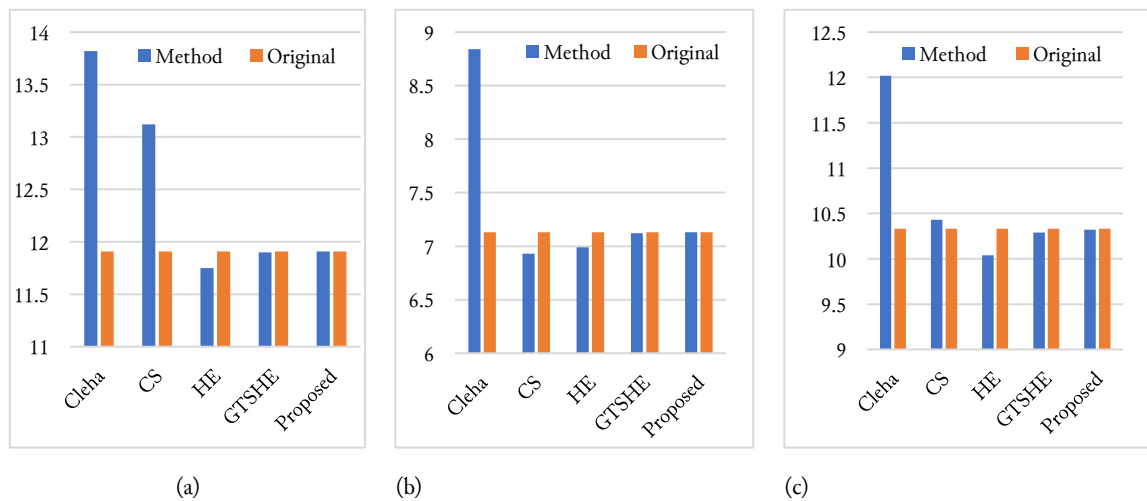


Fig. 8. The entropy values between the original and enhanced images of the Cleha, CS, HE, GTSHE, and Proposed methods for (a) Aerials, (b) CG-1050 dataset, and (c) Miscellaneous datasets.

Table 2 shows the results of the contrast, PSNR, and SSIM measurements. These measurements cannot give a direct meaningful evaluation in low contrast encasement because they cannot distinguish between fine enhancement, over-enhancement, under-enhancement, and noises. However, analysis of all values allows us to recognize the degrees of performance of each method. The contrast values show the differences between new gray-values of the enhanced images. A higher value means high contrast and more visual appearance, but we should ensure that the high values do not cause the loss information and produces noises and over-enhancement. Therefore, it should be attached with high values of SSIM and PSNR.

Table 2. The Contrast, PSNR and SSIM of the Cleha, CS, HE, GTSHE, and Proposed method for Aerials, CG-1050 dataset, and Miscellaneous datasets images.

	Aerials			CG-1050 dataset			Miscellaneous		
	Contrast	PSNR	SSIM	Contrast	PSNR	SSIM	Contrast	PSNR	SSIM
Original	27.71	Inf.	1	42.72	Inf.	1	43.12	Inf.	1
Cleha	46.54	19.51	0.77	47.86	27.71	0.74	52.27	23.04	0.85
CS	40.78	20.77	0.89	66.13	20.4	0.81	63.82	19.19	0.87
HE	73.56	13.76	0.57	75.11	16.18	0.61	74.89	15.6	0.65
GTSHE	49.12	18.9	0.83	54.06	25.91	0.76	56.2	20.36	0.91
Proposed	46.37	20.61	0.86	54.43	24.21	0.73	54.46	21.84	0.93

According to the previous explanation and the results in Table 2, the proposed method, followed by the GTSHE show the best performances. With the aerial dataset images, the proposed method produces higher contrast (46.37) than that of the original images (27.71), accompanied by the high values of SSIM and PSNR (0.86 and 20.61, respectively). The CS have high values of SSIM and PSNR (0.89 and 20.77, respectively) but scored the lowest contrast results (40.78) than all methods. The HE method produces high contrast values at 73.56 but has low SSIM and PSNR results (0.57 and 13.76). in case of the CG-1050 dataset, the proposed and GTSHE methods achieve an almost similar performance. Clahe presented high PSNR and SSIM values but had low contrast values (47.86). By comparison, HE has high contrast by (75.11) but scored the worst PSNR and SSIM values. With the Miscellaneous dataset, the proposed method scored a high contrast value (54.46) comparing with the original images (43.12) and achieved high SSIM and PSNR values (0.93 and 21.84, respectively).

In summary, the entropy measurement shows that the proposed method is the only approach that preserves the information and details in enhanced images as in the original images. Moreover, the visual evaluation shows that the proposed method satisfactorily improves the low contrast and the image quality. The results of the contrast, PSNR, and SSIM measurements indicate a high performance of the proposed method relative to the other techniques with all datasets.

3.3. Discussion

The previous experiments aim to evaluate each method's ability to improve image quality by improving low contrast levels in images by preserving the details, information, and features of processed images in the same original images. According to the literature review, Clahe [47], CS, HE [35], GTSHE [46][7] methods are popular histogram equalization methods that address many challenges of the low contrast problem. Therefore, high performance is expected from using these techniques to maintain the images' information as similar to the original. According to the results, the following conclusions were reached:

HE method [35] is a simple and popular method for enhancing the contrast of a histogram. The approach is suitable for improving the low contrast in a wide range of images. However, it highlights unnecessary information in several types of images. That feature led to undesirable effects and deterioration in the visual features which is reflected in the loss of information. CS [29] is also a simple and popular method to improve low contrast by stretching the histogram to its range values. It is an

acceptable method to improve the contrast in a wide range of images. However, CS may produce a slight improvement in several types of images and some noise effect. This outcome degrades unwanted effects or generates noise that distorts the original features of the images, thereby resulting in missing information or adding unwanted information to the processed image. Clahe [47] is one of the recent and popular contrast enhancement methods. It deals perfectly with the usual challenges in degraded images. In many cases, however, Clahe fails to perform efficiently. It may cause over-enhancement or under-enhancement. In addition, the main problem of this method is that it does not preserve the entropy. Clahe has worst entropy preservation among the involved methods. GTSHE [46] [7] enhances the low contrast and the visual appearance more effectively than previous methods. However, it shows extra brightness in many cases. In addition, the GTSHE outperforms other methods in the case of entropy, but it fails to preserve the entropy perfectly and in the same of original images.

The proposed method enhances the low contract and the visual properties of images. The approach also enhances the visual appearance of images efficiently. Moreover, the experiments indicate that the proposed methods perfectly preserve the entropy and image contents as comparable to the original images. Its results' entropy meets with all original images' entropy of involved datasets in the experiments.

Based on the above outcomes, the methods were effective on many images but presented some disparities. They reveal some weakness in their performances in the case of special images. In addition, they did not retain the original information of the images. Nevertheless, the results prove that the proposed method is the best in terms of preserving the original information details. The suggested approach improves contrast quality in all images without producing side effects, over-enhancement, and under-enhancement under all image states.

4. Conclusion

This research proposes a contrast enhancement method based on a histogram equalization approach that preserves the entropy of the enhanced image like that of the original image. The proposed method extracts the occurrences and histogram of the gray values in the original image. Then, the occurrences and the clipped histogram are equalized. Finally, the new pixel values are calculated by a proposed transferring function. To validate the performance of the proposed method, it compared with widely use benchmark methods such as Clahe, CS, HE, and GTSHE. Experiments using benchmark datasets and entropy, contrast, PSNR, and SSIM measurements are conducted to evaluate the performance. The results show that the proposed method is the only approach that preserves the information and details in enhanced images as in the original images. In addition, the results of the contrast, PSNR, and SSIM measurements indicate a high performance of the proposed method relative to the other techniques with all datasets. The proposed method satisfactorily improves the low contrast and the image quality without producing side effects.

Acknowledgment

The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (23UQU4361009DSR02).

Declarations

Author contribution. The contribution or credit of the author must be stated in this section.

Funding statement. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (23UQU4361009DSR02).

Conflict of interest. The author declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

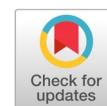
- [1] Y. Wang, R. Wan, W. Yang, H. Li, L.-P. Chau, and A. Kot, "Low-light image enhancement with normalizing flow," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, vol. 36, no. 3, pp.

- 2604–2612. doi : [10.1609/aaai.v36i3.20162](https://doi.org/10.1609/aaai.v36i3.20162).
- [2] A. K. Bhandari, S. Shah Nawazuddin, and A. K. Meena, “A Novel Fuzzy Clustering-Based Histogram Model for Image Contrast Enhancement,” *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 2009–2021, 2020, doi: [10.1109/TFUZZ.2019.2930028](https://doi.org/10.1109/TFUZZ.2019.2930028).
 - [3] D. C. Prakash, R. C. Narayanan, N. Ganesh, M. Ramachandran, S. Chinnasami, and R. Rajeshwari, “A study on image processing with data analysis,” in *AIP Conference Proceedings*, 2022, vol. 2393, no. 1, p. 20225, doi : [10.1063/5.0074764](https://doi.org/10.1063/5.0074764).
 - [4] T. Rahman *et al.*, “Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images,” *Comput. Biol. Med.*, vol. 132, p. 104319, 2021, doi : [10.1016/j.combiomed.2021.104319](https://doi.org/10.1016/j.combiomed.2021.104319).
 - [5] C. R. Nithyananda, A. C. Ramachandra, and Preeti, “Review on Histogram Equalization based Image Enhancement Techniques,” *Int. Conf. Electr. Electron. Optim. Tech. ICEEOT 2016*, pp. 2512–2517, 2016, doi: [10.1109/ICEEOT.2016.7755145](https://doi.org/10.1109/ICEEOT.2016.7755145).
 - [6] F. Lv, Y. Li, and F. Lu, “Attention Guided Low-Light Image Enhancement with a Large Scale Low-Light Simulation Dataset,” *Int. J. Comput. Vis.*, vol. 129, no. 7, pp. 2175–2193, 2021, doi: [10.1007/s11263-021-01466-8](https://doi.org/10.1007/s11263-021-01466-8).
 - [7] S. Chi Liu *et al.*, “Enhancement of Low Illumination Images based on an Optimal Hyperbolic Tangent Profile,” *Comput. Electr. Eng.*, vol. 70, pp. 538–550, 2018, doi: [10.1016/j.compeleceng.2017.08.026](https://doi.org/10.1016/j.compeleceng.2017.08.026).
 - [8] P. Paikrao, D. Doye, M. Bhalerao, and M. Vaidya, “A Combined Method for Document Image Enhancement Using Image Smoothing, Gray-Level Reduction and Thresholding,” in *Advancements in Smart Computing and Information Security: First International Conference, ASCIS 2022, Rajkot, India, November 24–26, 2022, Revised Selected Papers, Part I*, 2023, pp. 35–48, doi : [10.1007/978-3-031-23092-9_4](https://doi.org/10.1007/978-3-031-23092-9_4).
 - [9] R. W. Ibrahim, H. A. Jalab, F. K. Karim, E. Alabdulkreem, and M. N. Ayub, “A medical image enhancement based on generalized class of fractional partial differential equations,” *Quant. Imaging Med. Surg.*, vol. 12, no. 1, p. 172, 2022, doi : [10.21037/qims-21-15](https://doi.org/10.21037/qims-21-15).
 - [10] M. Jian, X. Liu, H. Luo, X. Lu, H. Yu, and J. Dong, “Underwater image processing and analysis: A review,” *Signal Process. Image Commun.*, vol. 91, p. 116088, 2021, doi: [10.1016/j.image.2020.116088](https://doi.org/10.1016/j.image.2020.116088).
 - [11] W. A. Mustafa and M. M. M. Abdul Kader, “Contrast Enhancement Based on Fusion Method: A Review,” *J. Phys. Conf. Ser.*, vol. 1019, no. 1, 2018, doi: [10.1088/1742-6596/1019/1/012025](https://doi.org/10.1088/1742-6596/1019/1/012025).
 - [12] A. Asokan, D. E. Popescu, J. Anitha, and D. J. Hemanth, “Bat algorithm based non-linear contrast stretching for satellite image enhancement,” *Geosci.*, vol. 10, no. 2, pp. 1–12, 2020, doi: [10.3390/geosciences10020078](https://doi.org/10.3390/geosciences10020078).
 - [13] M. Kanmani and V. Narasimhan, “Swarm intelligent based contrast enhancement algorithm with improved visual perception for color images,” *Multimed. Tools Appl.*, vol. 77, no. 10, pp. 12701–12724, 2018, doi: [10.1007/s11042-017-4911-7](https://doi.org/10.1007/s11042-017-4911-7).
 - [14] Q. C. Tian and L. D. Cohen, “A variational-based fusion model for non-uniform illumination image enhancement via contrast optimization and color correction,” *Signal Processing*, vol. 153, pp. 210–220, 2018, doi: [10.1016/j.sigpro.2018.07.022](https://doi.org/10.1016/j.sigpro.2018.07.022).
 - [15] H. Rahman and G. C. Paul, “Tripartite sub-image histogram equalization for slightly low contrast gray-tone image enhancement,” *Pattern Recognit.*, vol. 134, p. 109043, 2023, doi : [10.1016/j.patcog.2022.109043](https://doi.org/10.1016/j.patcog.2022.109043).
 - [16] H. Singh, A. Kumar, L. K. Balyan, and G. K. Singh, “Swarm intelligence optimized piecewise gamma corrected histogram equalization for dark image enhancement,” *Comput. Electr. Eng.*, vol. 70, pp. 462–475, 2018, doi : [10.1016/j.compeleceng.2017.06.029](https://doi.org/10.1016/j.compeleceng.2017.06.029).
 - [17] H. Singh, A. Kumar, L. K. Balyan, and H.-N. Lee, “Optimally sectioned and successively reconstructed histogram sub-equalization based gamma correction for satellite image enhancement,” *Multimed. Tools*

- Appl.*, vol. 78, no. 14, pp. 20431–20463, 2019, doi : [10.1007/s11042-019-7383-0](https://doi.org/10.1007/s11042-019-7383-0).
- [18] S. Kansal, S. Purwar, and R. K. Tripathi, “Image contrast enhancement using unsharp masking and histogram equalization,” *Multimed. Tools Appl.*, vol. 77, no. 20, pp. 26919–26938, 2018, doi: [10.1007/s11042-018-5894-8](https://doi.org/10.1007/s11042-018-5894-8).
- [19] J. Xiong *et al.*, “Application of Histogram Equalization for Image Enhancement in Corrosion Areas,” *Shock Vib.*, vol. 2021, 2021, doi: [10.1155/2021/8883571](https://doi.org/10.1155/2021/8883571).
- [20] P. Kandhway, A. K. Bhandari, and A. Singh, “A novel reformed histogram equalization based medical image contrast enhancement using krill herd optimization,” *Biomed. Signal Process. Control*, vol. 56, p. 101677, 2020, doi: [10.1016/j.bspc.2019.101677](https://doi.org/10.1016/j.bspc.2019.101677).
- [21] K. G. Dhal, A. Das, S. Ray, J. Gálvez, and S. Das, “Histogram Equalization Variants as Optimization Problems: A Review,” *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1471–1496, 2021, doi: [10.1007/s11831-020-09425-1](https://doi.org/10.1007/s11831-020-09425-1).
- [22] V. S. Padmavathy and R. Priya, “Image contrast enhancement techniques—a survey,” *Int. J. Eng. Technol.*, vol. 7, no. 2.33 Special Issue 33, pp. 466–469, 2018, doi: [10.14419/ijet.v7i1.1.10146](https://doi.org/10.14419/ijet.v7i1.1.10146).
- [23] S. F. Tan and N. A. M. Isa, “Exposure based multi-histogram equalization contrast enhancement for non-uniform illumination images,” *IEEE Access*, vol. 7, pp. 70842–70861, 2019, doi : [10.1109/ACCESS.2019.2918557](https://doi.org/10.1109/ACCESS.2019.2918557).
- [24] Y. Huang, Y. Li, and Y. Zhang, “A Retinex image enhancement based on L channel illumination estimation and gamma function,” vol. 137, no. Jiaet, pp. 312–317, 2018, doi: [10.2991/ijaet-18.2018.55](https://doi.org/10.2991/ijaet-18.2018.55).
- [25] W. Zhang, P. Zhuang, H.-H. Sun, G. Li, S. Kwong, and C. Li, “Underwater image enhancement via minimal color loss and locally adaptive contrast enhancement,” *IEEE Trans. Image Process.*, vol. 31, pp. 3997–4010, 2022, doi : [10.1109/TIP.2022.3177129](https://doi.org/10.1109/TIP.2022.3177129).
- [26] C. Ding, L. Tang, L. Cao, X. Shao, W. Wang, and S. Deng, “Preprocessing of multi-line structured light image based on Radon transformation and gray-scale transformation,” *Multimed. Tools Appl.*, vol. 80, no. 5, pp. 7529–7546, 2021, doi: [10.1007/s11042-019-08031-z](https://doi.org/10.1007/s11042-019-08031-z).
- [27] X. Fu, D. Zeng, Y. Huang, X. P. Zhang, and X. Ding, “A Weighted Variational Model for Simultaneous Reflectance and Illumination Estimation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 2782–2790, 2016, doi: [10.1109/CVPR.2016.304](https://doi.org/10.1109/CVPR.2016.304).
- [28] L. Florea, C. Florea, and C. Ionascu, “Avoiding the Deconvolution: Framework Oriented Color Transfer for Enhancing Low-Light Images,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, pp. 936–944, 2016, doi: [10.1109/CVPRW.2016.121](https://doi.org/10.1109/CVPRW.2016.121).
- [29] B. Bataineh and K. H. Almotairi, “Enhancement Method for Color Retinal Fundus Images Based on Structural Details and Illumination Improvements,” *Arab. J. Sci. Eng.*, vol. 46, no. 9, pp. 8121–8135, 2021, doi: [10.1007/s13369-021-05429-6](https://doi.org/10.1007/s13369-021-05429-6).
- [30] R. R. Hussein, Y. I. Hamodi, and R. A. Sabri, “Retinex theory for color image enhancement: A systematic review,” *Int. J. Electr. Comput. Eng.*, vol. 9, no. 6, pp. 5560–5569, 2019, doi: [10.11591/ijece.v9i6.pp5560-5569](https://doi.org/10.11591/ijece.v9i6.pp5560-5569).
- [31] P. Li, F. Wang, Y. Liang, and X. Zhang, “Single Image Defogging Method Based on Adaptive Modified Dark Channel Value,” vol. 91, no. Msbda, pp. 148–153, 2019, doi: [10.2991/msbda-19.2019.23](https://doi.org/10.2991/msbda-19.2019.23).
- [32] M. N. Aziz, T. W. Purboyo, and A. L. Prasasti, “A survey on the implementation of image enhancement,” *Int. J. Appl. Eng. Res.*, vol. 12, no. 21, pp. 11451–11459, 2017. Available at : [Semanticscholar](https://www.semanticscholar.org/).
- [33] D. Singh and V. Kumar, “Comprehensive survey on haze removal techniques,” *Multimed. Tools Appl.*, vol. 77, no. 8, pp. 9595–9620, 2018, doi: [10.1007/s11042-017-5321-6](https://doi.org/10.1007/s11042-017-5321-6).
- [34] N. Dey, “Uneven illumination correction of digital images: A survey of the state-of-the-art,” *Optik (Stuttg.)*,

- vol. 183, no. February, pp. 483–495, 2019, doi: [10.1016/j.ijleo.2019.02.118](https://doi.org/10.1016/j.ijleo.2019.02.118).
- [35] R. C. Gonzalez and R. E. Woods, “Digital image processing.” Prentice hall Upper Saddle River, NJ, 2002. Available at: sdeuoc.ac.id.
- [36] Q. Xu, H. Jiang, R. Scopigno, and M. Sbert, “A novel approach for enhancing very dark image sequences,” *Signal Processing*, vol. 103, pp. 309–330, 2014, doi: [10.1016/j.sigpro.2014.02.013](https://doi.org/10.1016/j.sigpro.2014.02.013).
- [37] H. Yoon, Y. Han, and H. Hahn, “Image contrast enhancement based sub-histogram equalization technique without over-equalization noise,” *World Acad. Sci. Eng. Technol.*, vol. 50, p. 2009, 2009. Available at : [GoogleScholar](https://www.google.com/scholar).
- [38] Y. Wang, Q. Chen, and B. Zhang, “Image enhancement based on equal area dualistic sub-image histogram equalization method,” *IEEE Trans. Consum. Electron.*, vol. 45, no. 1, 1999, doi: [10.1109/30.754419](https://doi.org/10.1109/30.754419).
- [39] S. Der Chen and A. R. Ramli, “Contrast enhancement using recursive mean-separate histogram equalization for scalable brightness preservation,” *IEEE Trans. Consum. Electron.*, vol. 49, no. 4, pp. 1301–1309, 2003, doi: [10.1109/TCE.2003.1261233](https://doi.org/10.1109/TCE.2003.1261233).
- [40] S. Der Chen and A. R. Ramli, “Minimum mean brightness error bi-histogram equalization in contrast enhancement,” *IEEE Trans. Consum. Electron.*, vol. 49, no. 4, pp. 1310–1319, 2003, doi: [10.1109/TCE.2003.1261234](https://doi.org/10.1109/TCE.2003.1261234).
- [41] K. S. Sim, C. P. Tso, and Y. Y. Tan, “Recursive sub-image histogram equalization applied to gray scale images,” *Pattern Recognit. Lett.*, vol. 28, no. 10, pp. 1209–1221, 2007, doi: [10.1016/j.patrec.2007.02.003](https://doi.org/10.1016/j.patrec.2007.02.003).
- [42] K. Singh and R. Kapoor, “Image enhancement using Exposure based Sub Image Histogram Equalization,” *Pattern Recognit. Lett.*, vol. 36, no. 1, pp. 10–14, 2014, doi: [10.1016/j.patrec.2013.08.024](https://doi.org/10.1016/j.patrec.2013.08.024).
- [43] N. Singh, L. Kaur, and K. Singh, “Histogram equalization techniques for enhancement of low radiance retinal images for early detection of diabetic retinopathy,” *Eng. Sci. Technol. an Int. J.*, vol. 22, no. 3, pp. 736–745, 2019, doi: [10.1016/j.jestch.2019.01.014](https://doi.org/10.1016/j.jestch.2019.01.014).
- [44] L. Zhuang and Y. Guan, “Image Enhancement via Subimage Histogram Equalization Based on Mean and Variance,” *Comput. Intell. Neurosci.*, vol. 2017, 2017, doi: [10.1155/2017/6029892](https://doi.org/10.1155/2017/6029892).
- [45] J. R. Tang and N. A. Mat Isa, “Bi-histogram equalization using modified histogram bins,” *Appl. Soft Comput. J.*, vol. 55, pp. 31–43, 2017, doi: [10.1016/j.asoc.2017.01.053](https://doi.org/10.1016/j.asoc.2017.01.053).
- [46] K. H. Almotairi, “A Global Two-Stage Histogram Equalization Method for Gray-Level Images,” *J. ICT Res. Appl.*, vol. 14, no. 2, 2020, doi : [10.5614/10.5614/itbj.ict.res.appl.2020.14.2.1](https://doi.org/10.5614/10.5614/itbj.ict.res.appl.2020.14.2.1).
- [47] S. M. Pizer *et al.*, “Adaptive histogram equalization and its variations,” *Comput. vision, Graph. image Process.*, vol. 39, no. 3, pp. 355–368, 1987, doi : [10.1016/S0734-189X\(87\)80186-X](https://doi.org/10.1016/S0734-189X(87)80186-X).

Multi-step CNN forecasting for COVID-19 multivariate time-series



Haviluddin ^{a,1,*}, Rayner Alfred ^{b,2}

^a Department of Informatics, Faculty of Engineering, Universitas Mulawarman, East Kalimantan, Indonesia

^b Knowledge Technology Research Group, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Sabah, Malaysia

¹ haviluddin@unmul.ac.id; ² ralfred@ums.edu.my

* corresponding author

ARTICLE INFO

Article history

Received March 4, 2023

Revised April 16, 2023

Accepted April 26, 2023

Available online May 2, 2023

Keywords

Multivariate time-series

CNN

Smoothing technique

COVID-19

ABSTRACT

The new coronavirus (COVID-19) has spread to over 200 countries, with over 36 million confirmed cases as of October 10, 2020. As a result, numerous machine learning models capable of forecasting the epidemic worldwide have been produced. This paper reviews and summarizes the most relevant machine learning forecasting models for COVID-19. The dataset is derived from the world health organization (WHO) COVID-19 dashboard, and it contains official daily counts of COVID-19 cases, fatalities, and vaccination use reported by countries, territories, and regions. We propose various convolutional neural network (CNN) based models such as CNN, single exponential smoothing CNN (S-CNN), moving average CNN (MA-CNN), smoothed moving average CNN (SMA-CNN), and moving average smoothed CNN (MAS-CNN). Here, MAPE and MSE are used to assess the suggested models. MAPE is frequently used to compare accuracy across time series with different scales. MSE, the model must strive for a total forecast equal to the entire demand. That is, optimizing MSE seeks to create a forecast that is right on average and so unbiased. The final result shows that SMA-CNN outperformed its baselines in both MAPE and MSE. The main contribution of this novel forecasting approach is a more accurate result as a base of the strategy of preventing COVID-19 spreads.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Past disease outbreaks have affected humanity, making WHO and national authorities worldwide fight these pandemics. The COVID-19 pandemic, first found in Wuhan, China, in December 2019, remains a major problem [1]. According to the WHO, 213 nations and territories have COVID-19 [2]. COVID-19 can spread through direct physical contact, coughing, or sneezing [3]. In this case, COVID-19 has a 14-day incubation period, which is key to its spread [4]. Therefore, the precise projection of recovered and contaminated COVID-19 cases is essential for monitoring the outbreak and implementing effective methods to prevent its spread.

Biomedical informatics is important to any study attempt to treat COVID-19 patients. Artificial intelligence (AI) is becoming more common in healthcare systems to detect diseases [5] and perform clinical diagnoses [6]. This outbreak demonstrates the need and viability of utilizing AI to anticipate outbreaks. AI tools have excellent feature extraction capabilities [7], which help public health experts make decisions. AI can quickly extract information from health reports [8], social media [9], news [10], and media [11]. AI approaches have advanced in the recent decade. Machine learning [12] and deep learning [13] can automatically extract significant elements from complex data. Moreover, deep learning

has advanced computer vision [14], robotics [15], medical imaging [16], chemistry [17], and forecasting [18]. Several studies in the literature focused on modeling, predicting, and forecasting COVID-19 spread based on recorded COVID-19 time series data to comprehend and manage this pandemic.

Deep learning approaches have gained popularity in time-series modeling and analysis due to their generalization and nonlinear approximation [19]. Deep learning models are created by automatically combining neural network layers and extracting significant information from vast data [20]. This approach has been studied in several machine-learning applications. For example, the LSTM model predicts new COVID-19 cases in Canada from January-March 2020 [21]. In Russia, Peru, and Iran, an enhanced LSTM model predicts COVID-19 epidemic patterns [22]. Moreover, SVR, LSTM, BiLSTM, and GRU are used to forecast COVID-19 time-series data in 10 nations, and the result shows that BiLSTM data accessible through June 27, 2020, show higher performance [23].

Due to its performance, deep learning has been effectively applied to various real-world prediction challenges, including time-series forecasting. They make accurate forecasts despite the noisy and chaotic character of time-series forecasting. CNN is a popular, efficient deep learning approach [24]. CNN models can filter input data noise and extract more valuable characteristics for the final prediction model [25]. Standard CNNs are "feed-forward neural networks" that use filters and pooling layers, well-suited for spatial autocorrelation data but not complex and extended temporal relationships. Therefore, removing noisy samples improves temporal data representation and forecasting system accuracy by highlighting relevant patterns. Smoothing strategies help track data seasonality and improve deep learning performance [26]. Due to their simplicity and firm performance in time series forecasting, most strategies use moving averages [27] and seasonal exponential smoothing [28]. Smoothing improves interpretability and integrates the data series changing pattern into the prediction model. Furthermore, a CNN model automatically generates forecasts from the smoothed results.

This paper proposed various CNN-based models such as CNN, single exponential smoothing CNN (S-CNN), moving average CNN (MA-CNN), smoothed moving average CNN (SMA-CNN), and moving average smoothed CNN (MAS-CNN). These multi-step CNN models may provide an accurate multivariate time-series analysis by modifying CNN to select and optimize the best smoothing techniques. The model and the baselines were then implemented in the various time series from the big five countries' COVID-19 dataset. The contribution of the research are.

- To increase the accuracy of the multivariate time-series forecasting analysis by employing innovative CNN in conjunction with smoothing approach optimization.
- To implement an innovative forecasting model that can adapt to different multivariate time series COVID-19 datasets from the United States of America, India, Brazil, France, and Germany.
- To produce a higher level of prediction accuracy by experimental study and validation of the model compared to the CNN, S-CNN, MA-CNN, SMA-CNN, and MAS-CNN.

The remaining parts of this work are structured as described below. The section titled "CNN-based Forecasting Using Smoothing Approach" provides a detailed explanation of the smoothing technique used for the CNN time series and an explanation of the experimental design. The dataset, the data normalization procedure, the forecasting process, and the key performance indicator are all presented in the "Materials and Methods" section. The "Results and Discussion" section presents the findings and an in-depth examination of the experiments. A summary of the research is provided here, along with a discussion of the numerical experiments. The last section, conclusions, gives a summary of the overall findings as well as potential areas for further research.

2. Method

To carry out research in a manner that is more methodical, we planned the experiment in the manner depicted in Fig. 1. We used a variety of datasets to evaluate the effectiveness of several smoothed CNN compared to basic CNN. Fig. 1 shows that the experimental design employed in this investigation included five scenarios. The five scenarios are as follows: (1) data is processed directly with CNN; (2)

data is smoothed using single exponential smoothing and then processed with CNN (S-CNN); (3) data is smoothed using MA and then processed with CNN (MA-CNN); (4) data is smoothed using CNN. Smoothing using single exponential smoothing first, then MA (SMA-CNN); and (5) smoothing using MA first, followed by smoothing using single exponential smoothing again, which is then processed using CNN (MAS-CNN).

The data quality can be improved by using data smoothing [29]. When applied to time-series data, the smoothing method gets excellent results after removing any outliers that could be present in the data [30]. This method can be easily comprehended and applied successfully in the new study without referring to or taking parameters from previous investigations. By taking the average of the previous values in a time series, smoothing processes make predictions more accurate. The algorithm provides a weighting value assigned to past observations to reduce noise, smooth the value of fluctuations in the data being utilized, and anticipate future values. In general, there are various common types of data smoothing. Single exponential smoothing (S) [31] and moving average (MA) [32] are two more frequent types of data smoothing. Smoothing is a strategy that can assist researchers in predicting trends when they are asked to do a forecasting task.

The exponential window function is utilized in the single exponential smoothing (S) method, a rule-of-thumb approach to smoothing time-series data [33]. Exponential functions are used to apply weights diminishing at an exponential rate over time. It is easy to understand and apply when making judgments based on the user's past assumptions, such as seasonality, and it does not require much time. The Moving Average (MA) is a formula used to examine data points [34]. It begins with creating a series of averages of various subsets of the complete dataset and then continues connecting those averages in the shape of a line.

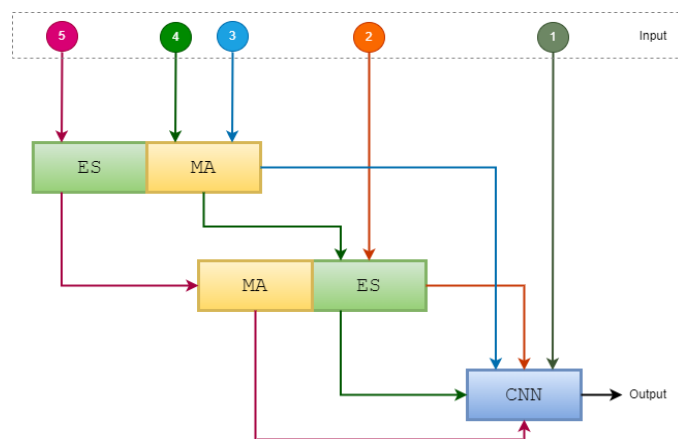


Fig. 1. Experimental Design

The use of single exponential smoothing by itself is not sufficient since it has the drawback of not being appropriate for anticipating data in seasonal and long-term periods, and the accuracy obtained is still inadequate. This makes the use of single exponential smoothing insufficient. In light of this, the motivation for the hybridization of smoothing techniques derived from single exponential smoothing with moving average (SMA) or vice versa moving average with single exponential smoothing (MAS) stems from the findings of this research.

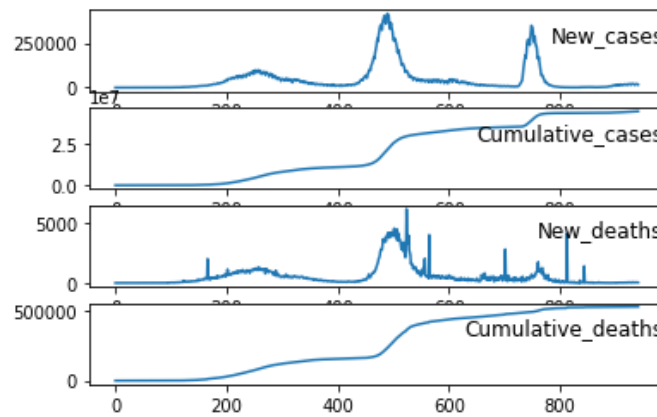
2.1. Dataset

This research used five datasets through the application of COVID-19, which had information from five countries with the highest number of instances anywhere in the globe. The dataset was obtained from the publicly available WHO website, which may be viewed and downloaded at <https://covid19.who.int/WHO-COVID-19-global-data.csv>. The WHO website is open to the general public. The information about the five countries mentioned above is included in Table 1. The time that will be looked at for this research starts on January 3, 2020, and goes through August 1, 2022.

Table 1. Big Five Countries COVID-19 Dataset

Dataset	Country	Number of Cases
1	America	90.213.060
2	India	44.036.275
3	Brazil	33.813.587
4	France	32.881.176
5	Germany	30.903.673

The new deaths attribute stores information on the number of deaths that occur each day, while the cumulative deaths attribute stores information on the total number of deaths that have occurred. The new cases attribute stores information on the number of new cases that occur each day, while the cumulative cases attribute stores the cumulative number of additional cases. The data utilized in this research is illustrated in Fig. 2, which provides a visual representation of the data.

**Fig. 2.** Visualization of Dataset

2.2. Data Normalization

Scaling a character into a particular range required by the activation function can only be accomplished through data normalization, an essential component of CNN [35]. The process of data normalization is utilized to address this issue. Since one of the primary goals of data normalization is to ensure the quality of the data before it is given to any model, its impact on the performance of any model is significant. The Min-Max normalization method was utilized in this research. Even though it is ineffective in dealing with outliers, the technique ensures that all characteristics have the same scale. The Min-Max formula is shown in (1), which produces normalized data with smaller intervals that fall inside the range 0–1 [36].

$$x' = \frac{(x - \min_x)}{(\max_x - \min_x)} \quad (1)$$

x' is the outcome of normalizing the data, x is the data that has to be normalized, \min_x is the minimum value of all the data, and \max_x is the maximum value of all the data.

2.3. Forecasting Process

In this study, the dataset used for predicting with CNN is first smoothed using single exponential smoothing and moving averages. Single exponential smoothing is employed. One may observe the equation for single exponential smoothing in (2).

$$S_t = S_{t-1} + \alpha(X_t - S_{t-1}) \quad (2)$$

The smoothed data S_t is a result of smoothing the raw data $\{X_t\}$. The smoothing factor α , is a variable that specifies the smoothing level [37]. The interval for α is between 0 and 1 ($0 \leq \alpha \leq 1$) [38]. When α is close to 1, the learning process is accelerated because the smoothing effect is diminished. In contrast, values of α closer to 0 have a greater smoothing effect and are less sensitive to changes in the

recent past [39]. Not all cases have the same value for α . Therefore, we determine the optimal dataset's properties smoothing factor value based on the dataset's properties [40]. The optimal alpha for single exponential smoothing is derived from (3). Then there is no need to manually attempt each α value from 0 to 1.

$$ptimum \alpha = \frac{(X_{max} - X_{min}) - \frac{1}{n} \sum_{i=1}^n X_t}{X_{max} - X_{min}} \tag{3}$$

So that the optimum single exponential smoothing (S_t) to improve the CNN algorithm performance used comes from the substitution of (3) to (2) results in the following (4).

$$S_t = S_{t-1} + \frac{(X_{max} - X_{min}) - \frac{1}{n} \sum_{i=1}^n X_t}{X_{max} - X_{min}} (X_t - S_{t-1}) \tag{4}$$

In the meantime, smoothing using moving average (MA) considers all of the data and uses a somewhat extended backward period. Data from the past are never left out of the computation. However, their weight in the final result is relatively minimal due to the nature of the moving average. It can illustrate ongoing trends while simultaneously eliminating fluctuations thanks to noise reduction. The data is smoothed using a moving average of either one month or thirty days when it is smoothed using MA. It is possible to visualize the MA in (5).

$$MA = \frac{(x_1 + x_2 + \dots + x_n)}{n} \tag{5}$$

MA is the outcome of smoothing the data using x , where x is the definition of each data point, and n is the number of periods.

The CNN algorithm is the primary focus of this research. CNN employs the fundamental Neural Network (NN) algorithm with additional layers. Because of its effectiveness, CNN has garnered much attention in computer vision and image processing. CNN uses a convolution layer that can process the spatial information in images, while fully connected layers are equipped with a memory that allows them to store information from time-series data. The input given to the model, an image matrix for computer vision problems and a 1D array for time series forecasting, is the only thing that differentiates computer vision problems from time series problems. The observation sequence can treat the raw input data as a one-dimensional array, which the CNN model can then read and filter. Therefore, the use of this theory in the time-series analysis is possible. CNN architecture as shown in Fig. 3

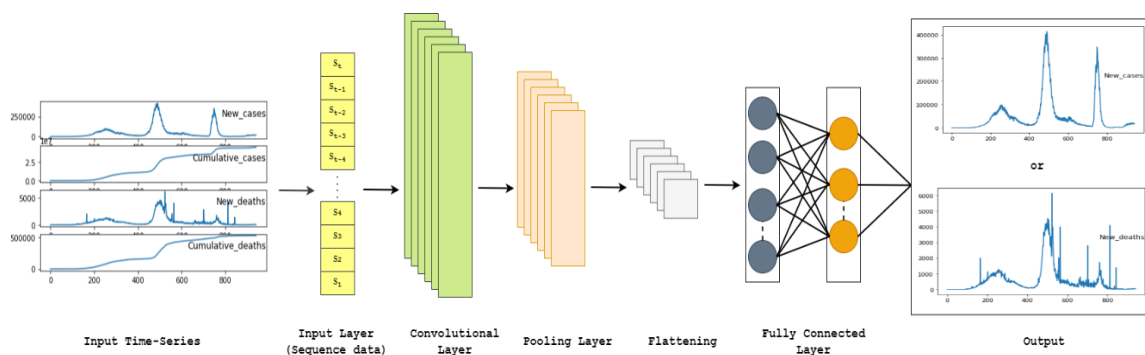


Fig. 3. CNN architecture

CNN's architecture comprises the following layers: an input layer, a convolutional layer, a pooling layer, a flattening, and a fully connected layer, as well as an output layer. Convolutional and pooling layers are designed to filter input data and extract valuable information for a fully connected network layer. Convolutional layers use raw input data and kernels to create new feature values. This technique was designed to extract image features from structured matrices. The convolution kernel (filter) is a narrow window containing coefficient values in matrix form. All these procedures result in a convolved

matrix representing a feature value specified by the filter coefficients and dimension size. By applying alternative convolution kernels to the input data, additional convolved features can be formed, usually more helpful than the original beginning features, boosting the model's performance.

A nonlinear activation function follows convolutional layers. Two typical activation functions are the sigmoid function and the rectified linear unit (ReLU). Both can be stated using (6) and (7) [41]. A pooling layer subsamples convolved features to create a lower-dimensional matrix. As with the convolutional layer, the pooling layer uses a small sliding window to take the values of each patch of convolved features and output one new value. Maximum and average pooling calculate each patch's maximum and average values. The pooling layer creates additional matrices that summarize the convolutional layer's features. Small input changes will not affect pooled output values, making the system more robust.

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (6)$$

$$f(x) = \max(0, x) \quad (7)$$

A list of CNN parameters can be adjusted in several different ways, depending on the application. Research [40] provides the basis for establishing the CNN parameters used in this investigation. In this work, we modified the parameter settings in the fully connected layer by optimizing the hyperparameter tuning using particle swarm optimization (PSO) [42], was done so that everything would not be precisely the same. The main reason is that the fully connected layer of the CNN reflects a more comprehensive set of features than the convolution layer. Each neuron in a fully connected layer is connected to all of the neurons in the layer below it [43]. A list of CNN forecast component parameters can be seen in Table 2.

Table 2. The list of CNN forecast component parameters

Layer	Parameter	Value
Convolutional	Type of convolutional	Conv1D
	The number of convolutional	1
	The number of filters	32
	The filter size	1
Pooling	The activation function	ReLU
	Type of pooling	MaXPooling1D
	The number of pooling layer	1
	Size of the pooling window	1
Flattent	Dropout	0.2
	The number of flatten layer	1
*Fully connected	The number of hidden layers	2
	The number of units or neuron	64
	The Activation function output	ReLU
	Loss function	MSE
	Type of optimizer	Adam
	The number of epochs	100
	The batch size	64

2.4. Key Performance Indicator

All experiments in this research were evaluated using key performance indicators, mean absolute percentage error (MAPE), and mean square error (MSE). In order to display errors in a manner that indicates accuracy, the MAPE metric is applied [27]. The MSE is a metric that can be used to detect outliers in a prediction system that has been created [44]. The MAPE and MSE values should be lower for a more accurate results prediction. The equation of MAPE and MSE can be seen in (8) and (9) [45].

$$MAPE = \sum_{t=1}^n \frac{|(A_t - F_t)|}{n \cdot A_t} \times 100 \quad (8)$$

$$MSE = \sum_{t=1}^n \frac{(A_t - F_t)^2}{n} \quad (9)$$

3. Results and Discussion

All of the data had been smoothed down by the experimental design of this research. At the same time, CNN was applied to the data. The outcomes of the forecasting evaluation were obtained in the form of MAPE and MSE, which can be viewed in [Table 3](#) and [Table 4](#).

Table 3. MAPE of New Cases

Method	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
CNN	10.90335	9.98161	10.56073	9.78018	10.87463
S-CNN	10.48021	9.73711	9.07266	9.54466	10.77463
MA-CNN	10.46680	9.41301	9.05266	9.48466	10.71020
SMA-CNN	10.38977	9.08656	9.03067	9.08466	10.28978
MAS-CNN	10.53081	9.52832	9.05067	9.22466	10.74021

The MAPE for the new case forecasts is presented in [Table 3](#). According to the table, we can conclude that the performance of CNN can be enhanced by utilizing exponential smoothing (S), moving average (MA), and its combination. Because of the treatment, there is a decline in the value of the MAPE. When applied to Dataset 3, SMA-CNN achieves the most outstanding performance of 9.03067. While using CNN, the worst possible acceptable result is 10.90335 for Dataset 1. As a result, SMA-CNN is the most effective algorithm for forecasting across all of the datasets included in this research.

Table 4. MSE of New Cases

Method	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
CNN	0.00283	0.00337	0.00492	0.00431	0.00593
S-CNN	0.00252	0.00336	0.00432	0.00381	0.00445
MA-CNN	0.00216	0.00290	0.00402	0.00193	0.00443
SMA-CNN	0.00199	0.00166	0.00362	0.00185	0.00299
MAS-CNN	0.00257	0.00298	0.00382	0.00225	0.00312

There is a connection between sensitivity and the application of MSE in performance testing. According to [Table 4](#), the MSE is reduced when the utilized algorithm is more complicated. In other words, SMA-CNN has a higher sensitivity than any other CNN-based algorithm that was tested for this study. Because the value of MSE is not very high, it can be deduced that the algorithm under consideration can recognize anomalies within the datasets used for forecasting. The results shown in [Table 5](#).

Table 5. MAPE of New Deaths

Method	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
CNN	10.59285	9.84640	9.63077	9.68018	10.97463
S-CNN	10.53449	9.31762	9.12146	9.67486	10.87463
MA-CNN	10.50479	9.20877	9.03994	9.21712	10.77463
SMA-CNN	10.33826	9.09776	9.02606	9.09712	10.38479
MAS-CNN	10.57165	9.24941	9.03624	9.13712	10.83463

[Table 5](#) are comparable to those in [Table 3](#). The MAPE that is the lowest is 9.02606, while the MAPE that is the greatest is 10.97463 (Dataset 5 using CNN). In every dataset, the performance of SMA-CNN was superior to that of CNN, S-CNN, MA-CNN, and MAS-CNN. The value reflects the

consistency of the accuracy of the forecast, although, under the same situations (Datasets 1, 2, and 5), MA-CNN performs somewhat better than MAS-CNN does. MSE of New Deats as show in Table 6.

Table 6. MSE of New Deats

Method	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
CNN	0.00602	0.00307	0.00832	0.00257	0.00373
S-CNN	0.00531	0.00302	0.00553	0.00194	0.00324
MA-CNN	0.00518	0.00216	0.00323	0.01390	0.00285
SMA-CNN	0.00456	0.00197	0.00263	0.00174	0.00209
MAS-CNN	0.00536	0.00243	0.00303	0.00214	0.00305

Table 6 has a similar pattern to Table 4. According to the data presented in the table, SMA-CNN has the smallest MSE, with 0.00174 being the best possible result (Dataset 4). This score, which is very near zero, indicates that all CNN variations can make accurate predictions. The MSE for SMA-CNN is the lowest of all the datasets.

S-CNN performs more accurately than CNN in most elements of the comparison, resulting from the application of optimal alpha, which was carried out [40]. The optimal value of alpha compels the process of smoothing to arrive at its optimal state, which results in a result that is both quick and accurate. In addition, the PSO hyperparameter tuning can produce an ideal model by applying the algorithm, thereby reducing errors in the dataset [42]. This smoothing process focuses more on values with the moving average timing in a given period, which can make the data more stable. MA-CNN may also improve results because of this smoothing process's attention to these values. It has been demonstrated that the smoothing technique known as smoothing moving average (SMA) has a performance that, when combined with exponential smoothing and moving average, can make time-series data prone to high volatility more stable.

On the other hand, MAS-CNN has been shown to have worse performance than MA-CNN in several tests. The series has no choice but to take on a linear form due to the exponential smoothing process after the moving average. Because of this condition, the absolute and mean square errors may increase. As a result of the MAPE and MSE, the SMA-CNN is considered the most practical combination of the moving average and exponential smoothing. We can use other exponential smoothings for further research, such as double exponential smoothing and triple exponential smoothing.

4. Conclusion

This research aims to improve the efficiency of CNN, an algorithm frequently utilized for image processing, by applying a smoothing strategy to its time-series analysis. Based on the investigation findings, one may conclude that the SMA-CNN model with the optimal smoothing factor performs significantly better than the other CNN-based forecasting smoothing technique techniques. The SMA-CNN model used in this investigation yields the highest-quality assessment results. The usage of moving averages in combination with single exponential smoothing is continued as a data preparation strategy since it considerably improves the effectiveness of the forecasting algorithm. Although the results of this research have addressed the study's objectives, there are still some limitations. The implementation of smoothing techniques that are optimized in CNN methods is the primary subject of this study. As a result, research will be conducted shortly to determine how applying this strategy to more sophisticated deep learning algorithms (such as LSTM, DBN, and RBF) would affect the results. The next item we will concentrate on is conducting a more in-depth study of the various smoothing methods that may be implemented using double or triple-exponential smoothing. In the future study that is conducted, both beta optimal and gamma optimum will be taken into consideration.

Acknowledgment

This work was supported by Laboratory of Artificial Intelligence, Universitas Mulawarman, Indonesia and Laboratory of Machine Learning, Universiti Malaysia Sabah, Malaysia.

Declarations

Author contribution. All authors contributed equally as the main contributor of this paper. All authors read and approved the final paper.

Funding statement. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

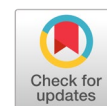
- [1] S. Zhao *et al.*, "Imitation dynamics in the mitigation of the novel coronavirus disease (COVID-19) outbreak in Wuhan, China from 2019 to 2020," *Ann. Transl. Med.*, vol. 8, no. 7, pp. 448–448, Apr. 2020, doi: [10.21037/atm.2020.03.168](https://doi.org/10.21037/atm.2020.03.168).
- [2] S. Chahar and P. K. Roy, "COVID-19: A Comprehensive Review of Learning Models," *Arch. Comput. Methods Eng.*, vol. 29, no. 3, pp. 1915–1940, May 2022, doi: [10.1007/s11831-021-09641-3](https://doi.org/10.1007/s11831-021-09641-3).
- [3] S. S. Diwan, S. Ravichandran, R. Govindarajan, and R. Narasimha, "Understanding Transmission Dynamics of COVID-19-Type Infections by Direct Numerical Simulations of Cough/Sneeze Flows," *Trans. Indian Natl. Acad. Eng.*, vol. 5, no. 2, pp. 255–261, Jun. 2020, doi: [10.1007/s41403-020-00106-w](https://doi.org/10.1007/s41403-020-00106-w).
- [4] L. A. Abraham, T. C. Brown, and S. A. Thomas, "How COVID-19's Disruption of the U.S. Correctional System Provides an Opportunity for Decarceration," *Am. J. Crim. Justice*, vol. 45, no. 4, pp. 780–792, Aug. 2020, doi: [10.1007/s12103-020-09537-1](https://doi.org/10.1007/s12103-020-09537-1).
- [5] M. Mirbabaie, S. Stieglitz, and N. R. J. Frick, "Artificial intelligence in disease diagnostics: A critical review and classification on the current state of research guiding future direction," *Health Technol. (Berl.)*, vol. 11, no. 4, pp. 693–731, Jul. 2021, doi: [10.1007/s12553-021-00555-5](https://doi.org/10.1007/s12553-021-00555-5).
- [6] S. Kaur *et al.*, "Medical Diagnostic Systems Using Artificial Intelligence (AI) Algorithms: Principles and Perspectives," *IEEE Access*, vol. 8, pp. 228049–228069, 2020, doi: [10.1109/ACCESS.2020.3042273](https://doi.org/10.1109/ACCESS.2020.3042273).
- [7] R. Liu, B. Yang, E. Zio, and X. Chen, "Artificial intelligence for fault diagnosis of rotating machinery: A review," *Mech. Syst. Signal Process.*, vol. 108, pp. 33–47, Aug. 2018, doi: [10.1016/j.ymssp.2018.02.016](https://doi.org/10.1016/j.ymssp.2018.02.016).
- [8] M. Jamshidi *et al.*, "Artificial Intelligence and COVID-19: Deep Learning Approaches for Diagnosis and Treatment," *IEEE Access*, vol. 8, pp. 109581–109595, 2020, doi: [10.1109/ACCESS.2020.3001973](https://doi.org/10.1109/ACCESS.2020.3001973).
- [9] C. Cuello-Garcia, G. Pérez-Gaxiola, and L. van Amelsvoort, "Social media can have an impact on how we manage and investigate the COVID-19 pandemic," *J. Clin. Epidemiol.*, vol. 127, pp. 198–201, Nov. 2020, doi: [10.1016/j.jclinepi.2020.06.028](https://doi.org/10.1016/j.jclinepi.2020.06.028).
- [10] S. Zeadally, E. Adi, Z. Baig, and I. A. Khan, "Harnessing Artificial Intelligence Capabilities to Improve Cybersecurity," *IEEE Access*, vol. 8, pp. 23817–23837, 2020, doi: [10.1109/ACCESS.2020.2968045](https://doi.org/10.1109/ACCESS.2020.2968045).
- [11] R. Vaishya, M. Javaid, I. H. Khan, and A. Haleem, "Artificial Intelligence (AI) applications for COVID-19 pandemic," *Diabetes Metab. Syndr. Clin. Res. Rev.*, vol. 14, no. 4, pp. 337–339, Jul. 2020, doi: [10.1016/j.dsx.2020.04.012](https://doi.org/10.1016/j.dsx.2020.04.012).
- [12] A. Holzinger, P. Kieseberg, E. Weippl, and A. M. Tjoa, "Current Advances, Trends and Challenges of Machine Learning and Knowledge Extraction: From Machine Learning to Explainable AI," in *Lecture Notes in Computer Science*, 2018, pp. 1–8, doi: [10.1007/978-3-319-99740-7_1](https://doi.org/10.1007/978-3-319-99740-7_1).
- [13] A. Ignatov *et al.*, "AI Benchmark: All About Deep Learning on Smartphones in 2019," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, Oct. 2019, pp. 3617–3635, doi: [10.1109/ICCVW.2019.00447](https://doi.org/10.1109/ICCVW.2019.00447).
- [14] Y.-J. Cao *et al.*, "Recent Advances of Generative Adversarial Networks in Computer Vision," *IEEE Access*, vol. 7, pp. 14985–15006, 2019, doi: [10.1109/ACCESS.2018.2886814](https://doi.org/10.1109/ACCESS.2018.2886814).
- [15] Y. Sun, B. Pan, and Y. Fu, "Lightweight Deep Neural Network for Real-Time Instrument Semantic Segmentation in Robot Assisted Minimally Invasive Surgery," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3870–3877, Apr. 2021, doi: [10.1109/LRA.2021.3066956](https://doi.org/10.1109/LRA.2021.3066956).
- [16] X. Zhang *et al.*, "Deep Learning Based Analysis of Breast Cancer Using Advanced Ensemble Classifier and

- Linear Discriminant Analysis,” *IEEE Access*, vol. 8, pp. 120208–120217, 2020, doi: [10.1109/ACCESS.2020.3005228](https://doi.org/10.1109/ACCESS.2020.3005228).
- [17] J. Sun, P. Veeffkind, P. van Velthoven, and P. F. Levelt, “Aerosol Absorption Over Land Derived From the Ultra-Violet Aerosol Index by Deep Learning,” *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 9692–9710, 2021, doi: [10.1109/JSTARS.2021.3108669](https://doi.org/10.1109/JSTARS.2021.3108669).
- [18] X.-H. Le, D.-H. Nguyen, S. Jung, M. Yeon, and G. Lee, “Comparison of Deep Learning Techniques for River Streamflow Forecasting,” *IEEE Access*, vol. 9, pp. 71805–71820, 2021, doi: [10.1109/ACCESS.2021.3077703](https://doi.org/10.1109/ACCESS.2021.3077703).
- [19] A. Mahmoud and A. Mohammed, “A Survey on Deep Learning for Time-Series Forecasting,” in *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*, A. Hassanien Aboul Ella and Darwish, Ed., Cham: Springer International Publishing, 2021, pp. 365–392, doi: [10.1007/978-3-030-59338-4_19](https://doi.org/10.1007/978-3-030-59338-4_19).
- [20] W. Zhong, N. Yu, and C. Ai, “Applying big data based deep learning system to intrusion detection,” *Big Data Min. Anal.*, vol. 3, no. 3, pp. 181–195, Sep. 2020, doi: [10.26599/BDMA.2020.9020003](https://doi.org/10.26599/BDMA.2020.9020003).
- [21] S. Bahri, M. Kdayem, and N. Zoghlami, “Long Short-Term Memory based RNN for COVID-19 disease prediction,” in *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, Mar. 2021, pp. 901–906, doi: [10.1109/ICIT46573.2021.9453534](https://doi.org/10.1109/ICIT46573.2021.9453534).
- [22] P. Wang, X. Zheng, G. Ai, D. Liu, and B. Zhu, “Time series prediction for the epidemic trends of COVID-19 using the improved LSTM deep learning method: Case studies in Russia, Peru and Iran,” *Chaos, Solitons & Fractals*, vol. 140, p. 110214, Nov. 2020, doi: [10.1016/j.chaos.2020.110214](https://doi.org/10.1016/j.chaos.2020.110214).
- [23] Y. Fei, “Analysis the use of machine learning algorithm-based methods in predicting COVID-19 infection,” in *Proceedings of the 2nd International Symposium on Artificial Intelligence for Medicine Sciences*, Oct. 2021, pp. 88–94, doi: [10.1145/3500931.3500948](https://doi.org/10.1145/3500931.3500948).
- [24] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J. Big Data*, vol. 8, no. 1, p. 53, Dec. 2021, doi: [10.1186/s40537-021-00444-8](https://doi.org/10.1186/s40537-021-00444-8).
- [25] L. Ren, J. Dong, X. Wang, Z. Meng, L. Zhao, and M. J. Deen, “A Data-Driven Auto-CNN-LSTM Prediction Model for Lithium-Ion Battery Remaining Useful Life,” *IEEE Trans. Ind. Informatics*, vol. 17, no. 5, pp. 3478–3487, May 2021, doi: [10.1109/TII.2020.3008223](https://doi.org/10.1109/TII.2020.3008223).
- [26] T. Cheng, F. Harrou, F. Kadri, Y. Sun, and T. Leiknes, “Forecasting of Wastewater Treatment Plant Key Features Using Deep Learning-Based Models: A Case Study,” *IEEE Access*, vol. 8, pp. 184475–184485, 2020, doi: [10.1109/ACCESS.2020.3030820](https://doi.org/10.1109/ACCESS.2020.3030820).
- [27] S. Karasu, A. Altan, S. Bekiros, and W. Ahmad, “A new forecasting model with wrapper-based feature selection approach using multi-objective optimization technique for chaotic crude oil time series,” *Energy*, vol. 212, p. 118750, Dec. 2020, doi: [10.1016/j.energy.2020.118750](https://doi.org/10.1016/j.energy.2020.118750).
- [28] M. B. A. Rabbani *et al.*, “A Comparison Between Seasonal Autoregressive Integrated Moving Average (SARIMA) and Exponential Smoothing (ES) Based on Time Series Model for Forecasting Road Accidents,” *Arab. J. Sci. Eng.*, vol. 46, no. 11, pp. 11113–11138, Nov. 2021, doi: [10.1007/s13369-021-05650-3](https://doi.org/10.1007/s13369-021-05650-3).
- [29] S. Smyl, “A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting,” *Int. J. Forecast.*, vol. 36, no. 1, pp. 75–85, Jan. 2020, doi: [10.1016/j.ijforecast.2019.03.017](https://doi.org/10.1016/j.ijforecast.2019.03.017).
- [30] R. K. M. Malhi *et al.*, “Applicability of Smoothing Techniques in Generation of Phenological Metrics of *Tectona grandis* L. Using NDVI Time Series Data,” *Remote Sens.*, vol. 13, no. 17, p. 3343, Aug. 2021, doi: [10.3390/rs13173343](https://doi.org/10.3390/rs13173343).
- [31] O. Mbaye, S. Konare, M. L. Ba, and A. Diop, “Short-Term Holt Smoothing Prediction Model of Daily COVID-19 Reported Cumulative Cases,” in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 2021, pp. 15–27, doi: [10.1007/978-3-030-90556-9_2](https://doi.org/10.1007/978-3-030-90556-9_2).
- [32] X. Yang and Y. Ni, “Least-squares estimation for uncertain moving average model,” *Commun. Stat. - Theory Methods*, vol. 50, no. 17, pp. 4134–4143, Sep. 2021, doi: [10.1080/03610926.2020.1713373](https://doi.org/10.1080/03610926.2020.1713373).
- [33] J. F. Rendon-Sanchez and L. M. de Menezes, “Structural combination of seasonal exponential smoothing forecasts applied to load forecasting,” *Eur. J. Oper. Res.*, vol. 275, no. 3, pp. 916–924, Jun. 2019, doi:

[10.1016/j.ejor.2018.12.013](https://doi.org/10.1016/j.ejor.2018.12.013).

- [34] M. Li, X. Liu, and F. Ding, "The filtering-based maximum likelihood iterative estimation algorithms for a special class of nonlinear systems with autoregressive moving average noise using the hierarchical identification principle," *Int. J. Adapt. Control Signal Process.*, vol. 33, no. 7, pp. 1189–1211, Jul. 2019, doi: [10.1002/acs.3029](https://doi.org/10.1002/acs.3029).
- [35] M. Yedroudj, F. Comby, and M. Chaumont, "Yedroudj-Net: An Efficient CNN for Spatial Steganalysis," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 2092–2096, doi: [10.1109/ICASSP.2018.8461438](https://doi.org/10.1109/ICASSP.2018.8461438).
- [36] A. R. F. Dewandra, A. P. Wibawa, U. Pujiyanto, A. B. P. Utama, and A. Nafalski, "Journal Unique Visitors Forecasting Based on Multivariate Attributes Using CNN," *Int. J. Artif. Intell. Res.*, vol. 6, no. 1, 2022, doi: [10.29099/ijair.v6i1.274](https://doi.org/10.29099/ijair.v6i1.274).
- [37] C. Ning and F. You, "Data-driven decision making under uncertainty integrating robust optimization with principal component analysis and kernel smoothing methods," *Comput. Chem. Eng.*, vol. 112, pp. 190–210, Apr. 2018, doi: [10.1016/j.compchemeng.2018.02.007](https://doi.org/10.1016/j.compchemeng.2018.02.007).
- [38] J. Mi, L. Fan, X. Duan, and Y. Qiu, "Short-Term Power Load Forecasting Method Based on Improved Exponential Smoothing Grey Model," *Math. Probl. Eng.*, vol. 2018, pp. 1–11, 2018, doi: [10.1155/2018/3894723](https://doi.org/10.1155/2018/3894723).
- [39] C. Pan, Y. Chen, L. Wang, and Z. He, "Lithium-ion Battery Remaining Useful Life Prediction Based on Exponential Smoothing and Particle Filter," *Int. J. Electrochem. Sci.*, vol. 14, pp. 9537–9551, Oct. 2019, doi: [10.20964/2019.10.15](https://doi.org/10.20964/2019.10.15).
- [40] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujiyanto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed Convolutional Neural Network," *J. Big Data*, vol. 9, no. 1, p. 44, Dec. 2022, doi: [10.1186/s40537-022-00599-y](https://doi.org/10.1186/s40537-022-00599-y).
- [41] Z. Shen, X. Fan, L. Zhang, and H. Yu, "Wind speed prediction of unmanned sailboat based on CNN and LSTM hybrid neural network," *Ocean Eng.*, vol. 254, p. 111352, Jun. 2022, doi: [10.1016/j.oceaneng.2022.111352](https://doi.org/10.1016/j.oceaneng.2022.111352).
- [42] A. Pranolo, Y. Mao, A. P. Wibawa, A. B. P. Utama, and F. A. Dwiyanto, "Robust LSTM With Tuned-PSO and Bifold-Attention Mechanism for Analyzing Multivariate Time-Series," *IEEE Access*, vol. 10, pp. 78423–78434, 2022, doi: [10.1109/ACCESS.2022.3193643](https://doi.org/10.1109/ACCESS.2022.3193643).
- [43] S. Liu, H. Ji, and M. C. Wang, "Nonpooling Convolutional Neural Network Forecasting for Seasonal Time Series With Trends," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 8, pp. 2879–2888, Aug. 2020, doi: [10.1109/TNNLS.2019.2934110](https://doi.org/10.1109/TNNLS.2019.2934110).
- [44] D. J. Kovacs *et al.*, "Membrane fouling prediction and uncertainty analysis using machine learning: A wastewater treatment plant case study," *J. Memb. Sci.*, vol. 660, p. 120817, Oct. 2022, doi: [10.1016/j.memsci.2022.120817](https://doi.org/10.1016/j.memsci.2022.120817).
- [45] S. Khullar and N. Singh, "Water quality assessment of a river using deep learning Bi-LSTM methodology: forecasting and validation," *Environ. Sci. Pollut. Res.*, vol. 29, no. 9, pp. 12875–12889, Feb. 2022, doi: [10.1007/s11356-021-13875-w](https://doi.org/10.1007/s11356-021-13875-w).

Hand-object interaction recognition based on visual attention using multiscopic cyber-physical-social system



Adnan Rachmat Anom Besari ^{a,b,1,*}, Azhar Aulia Saputra ^{a,2}, Wei Hong Chin ^{a,3}, Kurnianingsih ^{c,4}, Naoyuki Kubota ^{a,5}

^a Department of Mechanical Systems Engineering, Graduate School of Systems Design, Tokyo Metropolitan University, Tokyo 191-0065, Japan

^b Department of Information and Computer Engineering, Politeknik Elektronika Negeri Surabaya, Sukolilo, Surabaya 60111, Indonesia

^c Department of Electrical Engineering, Politeknik Negeri Semarang, Jl.Prof. Sudarto, Tembalang, Semarang 50275, Indonesia

¹ anom@pens.ac.id; ² aa.saputra@tmu.ac.jp; ³ weihong@tmu.ac.jp; ⁴ kurnianingsih@polines.ac.id; ⁵ kubota@tmu.ac.jp

* corresponding author

ARTICLE INFO

Article history

Received August 26, 2022

Revised November 14, 2022

Accepted April 13, 2023

Available online May 2, 2023

Keywords

Telemedicine

First-person vision

Hand-eye coordination

Independent rehabilitation

Occupational therapy

ABSTRACT

Computer vision-based cyber-physical-social systems (CPSS) are predicted to be the future of independent hand rehabilitation. However, there is a link between hand function and cognition in the elderly that this technology has not adequately supported. To investigate this issue, this paper proposes a multiscopic CPSS framework by developing hand-object interaction (HOI) based on visual attention. First, we use egocentric vision to extract features from hand posture at the microscopic level. With 94.87% testing accuracy, we use three layers of graph neural network (GNN) based on hand skeletal features to categorize 16 grasp postures. Second, we use a mesoscopic active perception ability to validate the HOI with eye tracking in the task-specific reach-to-grasp cycle. With 90.75% testing accuracy, the distance between the fingertips and the center of an object is used as input to a multi-layer gated recurrent unit based on recurrent neural network architecture. Third, we incorporate visual attention into the cognitive ability for classifying multiple objects at the macroscopic level. In two scenarios with four activities, we use GNN with three convolutional layers to categorize some objects. The outcome demonstrates that the system can successfully separate objects based on related activities. Further research and development are expected to support the CPSS application in independent rehabilitation.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Rehabilitation is required for patients recovering from neurological illnesses, particularly hand stroke. However, roughly two-thirds of hand stroke survivors have visual impairments because of visual field loss, double vision, and perception issues [1]. Vision problems may lead to different adverse outcomes, including grasping objects because of their limited range of motion and vision. Individuals find it more convenient to receive rehabilitation treatment at home rather than in a hospital setting. However, visiting patients' homes is difficult during the COVID-19 outbreak. Telemedicine allows therapists to monitor rehabilitation patients via online therapy or video visits [2]. Because of limited therapeutic staff or privacy concerns, patients can not use this method indefinitely. Cyber-physical-social system (CPSS) [3] seamlessly integrates physical and social space in cyberspace. This technology can deliver valuable information which not available from typical physical sensors.

Hand movement progress in poststroke patients has been studied using cyber-physical systems (CPS) [4]. This study has been conducted to track individual hand therapy. Hand monitoring can be done in

two ways: contact and noncontact. Most studies employ the contact method, where patients wear a device in their hands. In rehabilitation research, wearable hand devices are commonly used as contact methods. It provides accurate data by utilizing a variety of sensors, including flex, accelerometers, and hall-effect sensors [5]. However, this strategy has drawbacks, such as high equipment costs and uncomfortable use.

Consequently, scientists are investigating new possibilities using noncontact techniques, such as low-cost computer vision systems. Still, suppose this method does not consider some social aspects, such as dealing with privacy issues, getting justification from a clinical background, and understanding user experiences to provide better benefits. In that case, it may fail in user acceptance. Noncontact techniques for detecting human action recognition [6] still deal with complicated processing, multiple interpretations, and privacy concerns. Hence, an egocentric vision like smart glasses is required to address these issues [7]. Egocentric vision has the advantages of reducing privacy concerns, monitoring mobile devices, and attracting attention during activities. Hand-object interaction (HOI) recognition using egocentric vision should be studied further in rehabilitation applications [8].

HOI recognition is increasingly being used in post-stroke therapy to track patients' progress [9]. This recognition study outperforms full-body human interaction detection. This vision analyzes images captured by an on-body camera, such as smart glasses or an action cam. Still, HOI recognition is more straightforward because only hands and objects are detected, especially when using egocentric vision [10]. However, when using this vision, the expansion of HOI recognition encounters several challenges, particularly in 2D images. First, researchers frequently encounter data on hand and object invalid contact detection. This issue arises because 2D image data cannot express depth data. As a result, the system could not pinpoint the precise location of the hand and objects. Numerous approaches can address the issues, such as learning with the interaction point [11]. However, these approaches are limited to recognizing a single object type, and problems may arise when detecting many objects.

Furthermore, the application of this method is limited to identifying hands and objects but not considering persons with visual impairment. Because of their visual impairment, patients struggle to make physical hand movements and apply their recognition ability. The egocentric vision that focuses on HOI recognition has the potential to monitor both physical and cognitive rehabilitation simultaneously. This vision system, for example, could use hand skeletal estimations and the kinematic finger model to assess the person's physical condition. Besides that, specific cameras are outfitted with eye tracking to gather data on the user's visual attention. Gaining user experience and developing cognitive abilities are required. Therefore, while handling objects, it's crucial to pay attention to hand movements and vision.

The study's primary contributions are: (1) We apply egocentric vision and feature extraction to observe hand posture at the microscopic level. We use three layers of graph neural network (GNN) as a feature-based classifier to differentiate 16 grasp poses when interacting with objects. (2) At the mesoscopic level, we use active perception to validate HOI recognition with eye tracking in the task-specific reach-to-grasp cycle. To identify the connection of the hand with an object, we use the distance between the fingertips and the center of an object as inputs to a multi-layer gated recurrent unit (MGRU) based on recurrent neural networks (RNN) architecture. (3) We implement a cognitive ability at the macroscopic level by incorporating visual attention. In two different scenarios, we use the object relation as the input of a GNN node classifier with three convolutional layers to separate objects based on related activities. This method's output indicates the object's relationship in activities based on personal behavior.

The structure of this article is as follows. Section 2 discusses the research in hand rehabilitation monitoring and our proposed method for improving HOI recognition based on visual attention. Section 3 discusses the findings and assesses the efficacy of the proposed framework. Finally, Section 4 discusses the research's findings and some future directions.

2. Method

Recent research trends in hand movements analysis using egocentric vision have been widely used to advance the field of rehabilitation. Some researchers are interested in introducing the patient's hand behavior, for example, by using fingertip detection when using a therapy ball [12], monitoring hands in spinal cord injury patients [13]–[15], and stroke patients [9] [16]. Other research focuses on computational problem solutions, such as the high cost of additional equipment and pixel-level observations [17] and overcoming occlusion, inference, and contact [18]. Many studies employ egocentric approaches, like GoPro wearable cameras or datasets like Deeplab-VGG16, EgoHand, EPIC-ADL, and multi-datasets [19]. Existing research focuses solely on physical hand evaluation. It is uncommon to find studies that combine physical hand and cognitive abilities [20], such as in hand-eye coordination research for predicting the “next active object” shortly [21]. Table 1 shows the research on hand rehabilitation with egocentric vision in the last 5 years.

Table 1. The research of hand rehabilitation with egocentric vision (2018–2022)

No.	Research	Application (Sensor Types/Dataset)	Methods
1.	Qurratu'aini <i>et al.</i> [12]	Fingertips gripping a therapy ball for hand recovery. (HD Logitech C615 Web Camera with 1920 × 1080)	Speeded Up Robust Features (SURF) descriptors, K-mean clustering, and Support Vector Machine (SVM).
2.	Li <i>et al.</i> [17]	Overcoming expensive equipment and pixel-level annotations. (Deeplab-VGG16 Dataset)	Un-supervised hand segmentation using a fully convolutional neural network (FCN).
3.	Likitlersuang <i>et al.</i> [13]	Monitoring spinal cord injury patients' hand usage at home. (GoPro Hero4 with 1920 × 1080/30fps)	Fast R-CNN (hand detection), Contour Selection (hand segmentation), and Functional Measure Extraction (interaction detection).
4.	Visée <i>et al.</i> [14]	Home monitoring of SCI patient's upper limb function. (GoPro Hero4 with 1920 × 1080/30fps)	YOLOv2 (hand detection), DAT (hand tracking), and Random Forest Classifiers (interaction detection).
5.	Xu <i>et al.</i> [16]	Developing a low-cost technology to monitor stroke patient hand motions and gestures. (EgoHand datasets)	CNN-based hand motion and gesture detection.
6.	Tsai <i>et al.</i> [9]	Evaluating hand functions after a stroke. (GoPro Hero 5 with 1280 × 720/30fps)	YOLOv2 (hand detection), UNET (hand tracking); Random Forest Classifiers (interaction detection).
7.	Jiang <i>et al.</i> [21]	Visual attention and hand posture to predict the next active object. (EPIC and ADL Dataset)	The deep neural network model combines visual and hand cues.
8.	Bandini <i>et al.</i> [16]	Assessing hand usage at home after a cervical spinal cord injury. (GoPro Hero 5 Black (1280 × 720/30fps)	Deep learning model (hand localization), HOID-Net (interaction detection), statistical analysis.
9.	Lee <i>et al.</i> [18]	Detecting hand motion tracking with robustness against occlusion, interference, and contact. (Stereo camera, intelligent gloves, and IMU)	Visual-Inertial Skeleton Tracking (VIST)
10.	Our proposed method	Develop HOI based on visual attention using multiscope CPSS for physical-cognitive rehabilitation support. (Tobii Pro 3 Eye tracker 1920 × 1080/30fps)	GNN graph classifier (grasp pose classification), MGRU (multivariate time-series for HOI classification), and GNN node classifier (object classification).

Previous work in our laboratory focused on nonverbal communication for socially integrated robot companions using directed learning [22]. This study examines person's intents and abilities when reaching for and gripping objects. However, determining individual's preferences for using the robot companion as a third person is difficult [23]. Occlusion limits the third-person perspective, which depends on many different viewpoints. From an egocentric standpoint, it is critical to support the current system. With a case study on the Chopsticks Manipulation Test, we examined the significance of combining finger joint angle estimation and a visual attention measurement in hand rehabilitation [24]. Our previous work used a multiscope method to address dynamic locomotion in a legged robot [25]

and simulation for human-robot interactions [26]. We propose a multiscope approach for developing a CPSS for HOI recognition based on visual attention based on this experience. Fig. 1 depicts the framework of HOI recognition based on visual attention using multiscope CPSS. The subsections explain GNN, microscopic, mesoscopic, and macroscopic levels. Each subsection outlines methodology, algorithm, and development in detail.

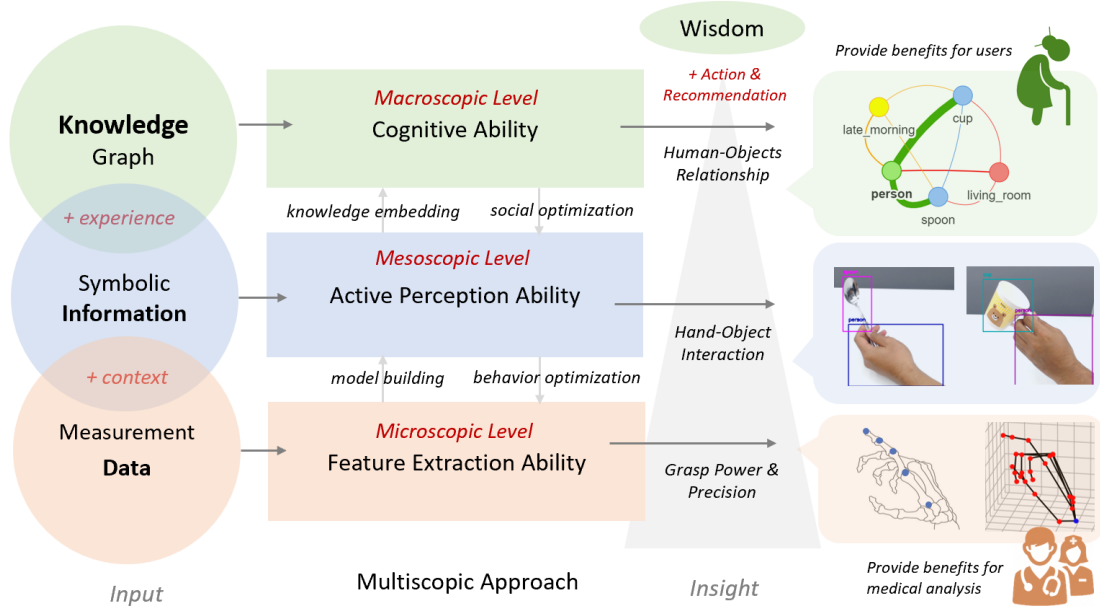


Fig. 1. The framework of HOI recognition is based on visual attention using multiscope CPSS

Promoting independent hand rehabilitation from the ground up is critical, from the physical to the cognitive. This study presents the CPSS framework, which employs the multiscope method to address the following technical issues: (a) Classify hand data at the microscopic level utilizing feature extraction abilities; data from vision sensors will be the input for this level. Using symbolic information, we develop a feature extraction capability to estimate hand and finger posture. (b) Using active perception ability, extract HOI at the mesoscopic level. At the mesoscopic level, this information will be used as input. As a knowledge graph, we will build active perceptions' ability to categorize HOI. (c) Discover the object relationship via macroscopic cognition ability. At this level, the graph data will be used as input. We hone our cognitive abilities to demonstrate recommendations based on human behavior.

2.1. Graph Neural Networks

GNN is neural network architecture used to learn the representations of graphs data and has become a popular learning model for prediction tasks on nodes, graphs, and links. The basic idea of GNN is to learn suitable graph data representation for neural networks [27]. Before we talk about GNN, we should look at the basic mathematics of graph structure data in computer science. A graph G can be a part of set attributed graphs \mathcal{G} . GNN use all graph information as an input, including the node features and the connections stored in the adjacency matrix. A graph G is defined by the following equation:

$$G = (V, E, X), G \in \mathcal{G} \quad (1)$$

Where $V = \{v_1, \dots, v_n\}$ is a set of nodes and $E = \{e_{a,b}, \dots, e_{i,j}\}$ is a set of ordered couples representing the connection between two nodes belonging to V . Each node comes with $X = \{x_v\}$ as a set of node attributes where $v \in V$. GNN output new representation called embeddings for each node. These node embeddings contain the structural and feature information of other nodes in the graph. The embeddings can finally be used to perform predictions. We embed each node through several rounds of message passing. This paradigm can be broken down into (a) initialization, (b) aggregation, and (c) update. We initialize each node v at layer $k = 0$ as the first round of message passing with the following equation:

$$h_{G,v}^{(0)} = x_v, v \in V \quad (2)$$

Suppose $h_{G,v}^{(k)}$ represents the node embeddings for some vertex v at layer k -th, where node feature x_v of all nodes $v \in V$ in graph G . Second, we do the aggregation function for each node v with the following equations:

$$m_{G,v}^{(k)} = f_{Agg}^{(k)}(h_{G,u}^{(k-1)}), 1 \leq k \leq K. \quad (3)$$

$$= \frac{1}{|N(v)|} \sum_{u \in N(v)} W_{i,j} h_{G,u}^{(k-1)}, i \neq j, 1 \leq i, j \leq |V|. \quad (4)$$

We utilize the next step of the neural message passing scheme where we localized node v from their neighbors $N(v)$. The node feature $h_u^{(k-1)}$ of all nodes $u \in N(v)$ in graph G are iteratively aggregating and stored in $m_{G,v}^{(k)}$ as aggregation function $f_{Aggregate}^{(k)}$. The $N(v) \subset V$ denotes the neighborhood of $v \in V$. The aggregation function performs a significant role and is shared by all nodes within an iteration. An average, degree-normalized sum or coordinate-wise min or max may also replace the sum.

Third, we employ the last step of the neural message passing scheme where the node feature $h_{G,v}^{(k-1)}$ of all nodes $v \in V$ in graph G are iteratively updated by the aggregating result from their neighbors $N(v)$ with the following equations:

$$h_{G,v}^{(k)} = f_{Up}^{(k)}(h_{G,v}^{(k-1)}, m_{G,v}^{(k)}). \quad (5)$$

$$= \sigma(W_{i,i} h_{G,v}^{(k-1)} + m_{G,v}^{(k)}), 1 \leq i \leq |V|. \quad (6)$$

The update function $f_{Up}^{(k)}$ is usually a weighted combination with learnable weight matrices. The update functions are implemented as a fully connected layer that alternates linear transformations and coordinate-wise nonlinear activations σ such as *ReLU*, *tanh*, or *sigmoid*. Fig 2 shows the graph representation including the graph with node feature and message passing mechanism.

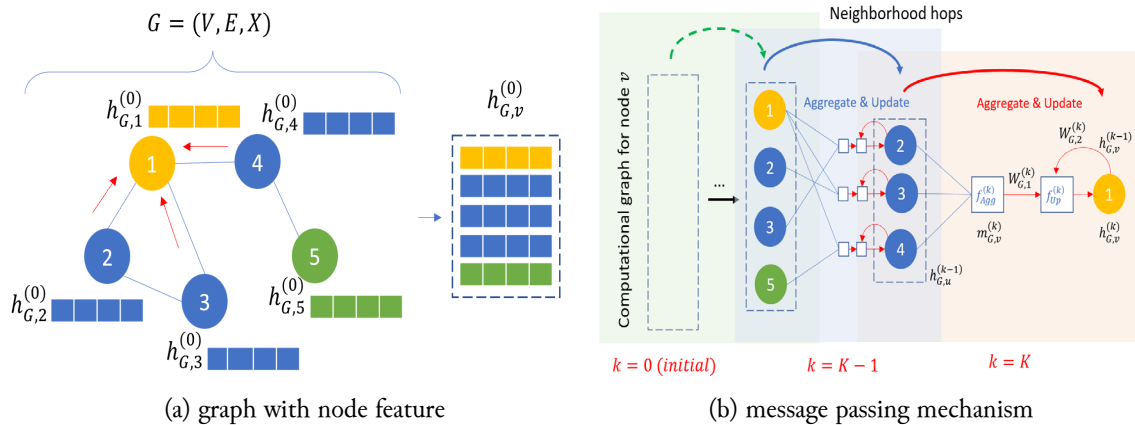


Fig. 2. Graph representation

The final node representation $h_{G,v}^{(K)}$ is the last layer, possibly concatenated with a linear classifier. If we want to make a graph-level prediction, all node embeddings can be aggregated into a unified graph embedding $H_G^{(K)}$ with f_{Read} . The most popular method is to take the average of node embeddings. We add up all the node features of all nodes $h_{G,v}^{(K)}$ in the K -th layer and then dividing by the number of nodes, as the following equation:

$$H_G = f_{Read}(h_{G,v}^{(K)}). \quad (7)$$

$$= \frac{1}{|V|} \sum_{v \in V} h_{G,v}^{(K)} \quad (8)$$

In the end, we use a linear transformation based on a fully connected layer with W_{proj} as weight projection and $arg \max$ function to determine which class corresponds to the graph input with the highest probability, as the following equation:

$$y_i = H_G W_{proj} \quad (9)$$

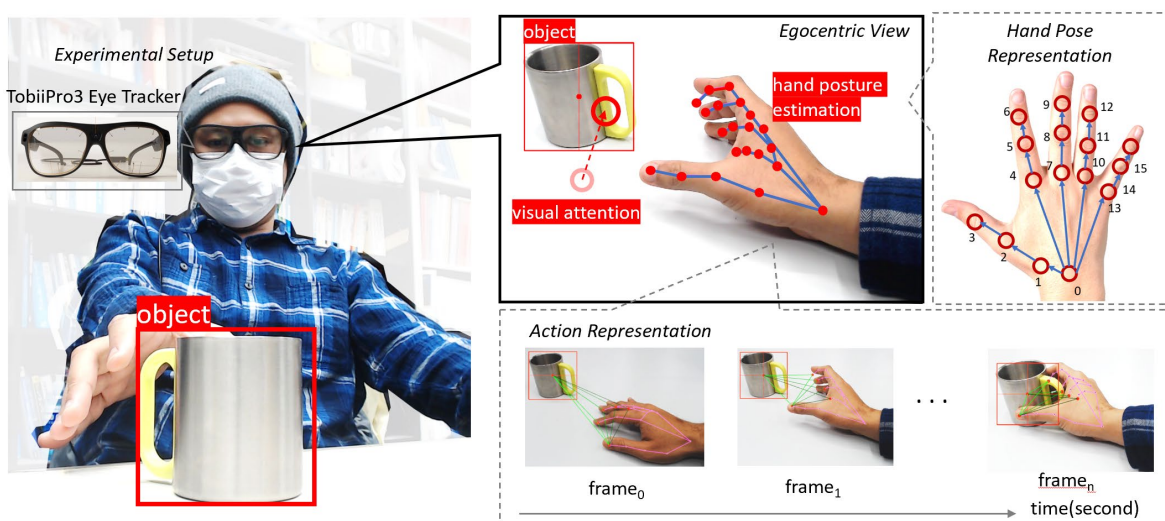
$$\hat{y} = arg \max_i y_i \quad (10)$$

We utilize Pytorch Geometric (PyG) as the GNN development framework. Our design uses high-level graph computation with PyG to teach scene interpretation. Because our graph classification datasets are small, we do a mini-batch for the graphs before inputting them into a GNN to guarantee full GPU utilization. PyG automatically takes care of batching multiple graphs into a single giant graph.

2.2. Microscopic Level: Feature Extraction Ability

The experimental setup and hand pose classification comprise the feature extraction ability at the microscopic level. We use egocentric vision with smart glasses facing the table and the objects. A participant facing down directly at the object wore Tobii Pro Glasses 3 smart glasses [28]. The smart glasses have a 1920×1080 -pixel resolution and a frame rate of 25 frames per second. To capture hands and objects, this camera is used in 16:9 scene camera format with a wide field of view of 106° with 95° horizontal and 63° vertical. Visual attention or awareness is predicted to be appropriately detected when the interaction between the hand and the object falls within the field of view range.

We used the YOLOv5 [24] model to extract the object's location from picture frames represented by the bounding box and labels. To improve this detection, the Simple Online Real-time Tracking (SORT) [29] technique was used. This framework excels at representation learning and applying it to object recognition and tracking applications. We employ MediaPipe [30] hand tracking to obtain estimated hand posture data. A construction designed specifically for complex perceptual channels that use rapid real-time inference. We only utilize hand posture prediction as supporting data to validate HOI recognition in our approach. Object detection and hand estimation provide us with two pieces of information. First, we can look for an object in a specific image, and then we can pinpoint the precise location of the hand and its feature in the two-dimensional image. Fig. 3 depicts the experiment's design, which includes the experimental setup of the systems and the implementation of an egocentric view of HOI recognition with visual attention.



(a) Experimental setup

(b) Egocentric vision with attention

Fig. 3. Design of the experiment

Many hand grasp variants and orientations are collected. We standardize the angular data to eliminate outliers. Then, we perform three positions in the vector-to-joint-angle conversion to obtain each finger feature. This transformation is accomplished by converting these three-dimensional coordinate points into an angle. Below is the equation used to calculate an angle from two vectors in three-dimensional coordinates:

$$\theta = \arccos\left(\frac{\overline{AB} \cdot \overline{BC}}{\|\overline{AB}\| \|\overline{BC}\|}\right) \tag{11}$$

We can compute \overline{AB} and \overline{BC} if we have the coordinates for three points (A, B, and C). The angle obtained by $A \rightarrow B \rightarrow C$ employing the right-hand rule from B continues using dot products, whereas $\|\overline{AB}\|$ determines the length of \overline{AB} , $\|\overline{BC}\|$ determines the size of \overline{BC} , and θ (theta) is the angle formed by two vectors. And then, we can get the dot product $\overline{AB} \cdot \overline{BC}$ as well as the lengths $\|\overline{AB}\|$ and $\|\overline{BC}\|$. After all, by replacing the equation, we rearrange the formula for determining θ . These joint angle values are used for features in the data graph. We represent the relationship between joints in a directed graph (digraph).

We use this GNN to classify 16 manipulation grasping poses. We acquire 130 data for every grasp pose in various orientations. Thus, we have 2080 graphs with divisions of 1600 for training and 480 for testing. Our data graph consists of 16 nodes consisting of all nodes connected by 15 edges. We utilize a graph input layer consisting of 15 joint angle nodes and one wrist node as the center of the finger connection. The layer of output consists of 16 nodes representing grasping posture. We train a final classifier on graph embedding. Before applying the final classifier on top of the graph, we employ Rectifier Linear Unit (ReLU) activation function to create localized node embeddings. We use three layers of GNN with the same training cycle: construct an optimizer, feed the model inputs, compute the loss, and optimize using autograd. A linear transformation layer and *argmax* function are used to classify and determine which class of grasp posture corresponds to the input with highest probability. We discuss training and testing outcomes in the results and discussion section. Fig. 4 shows the microscopic level design using a three-layer GNN for hand pose classification.

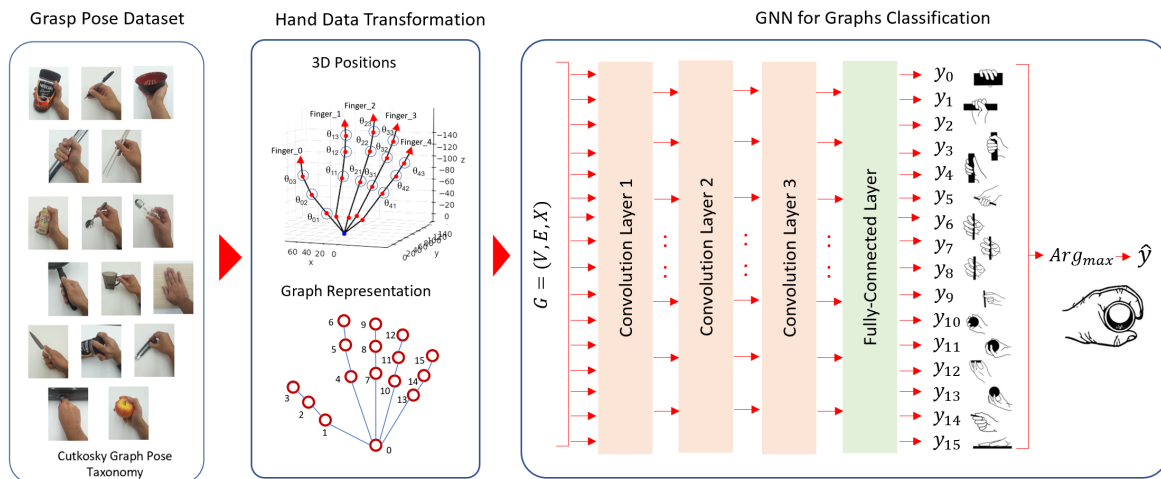


Fig. 4. The microscopic level design uses a three-layer GNN for hand pose classification

2.3. Mesoscopic Level: Active Perception Ability

This subsection describes the phases of active perception ability development at the mesoscopic level. These phases are object-centered coordinate transformation and validation of HOI recognition using the task-specific reach-to-grasp cycle. We accomplished a coordinate transformation centered on the object to simplify the validation procedure and get fewer data [31]. The goal of centering the item is to relocate the image coordinates' initials (0,0) to the middle of the object. We obtain a new center and identify the position of the new coordinates for every new frame. This method uses for any pixel (x_n, y_n) in the image plane. The joint finger position in the new coordinate plane contains

the object’s length (a_0) and width (b_0). The inner and outer borders of the object are acquired using this additional bounding box location information. Then, using the Pythagorean equation, we can estimate the distance between a point (a_n, b_n) to the center of the object coordinates (0,0). The distance d_n must be determined between the fingertip or finger joint and the object-centered coordinate. To get these properties, we then utilize a_n, b_n , and d_n to validate the HOI transformation.

Validation of HOI recognition is restricted to the reference grasp of the approved hand usage section in ICF for hand rehabilitation, notably in the case of the reach-to-grasp cycle [32]. The procedure has four distinct tasks, each of which is defined separately. The “wonder” task, which indicates that the user moves the hand without reaching for the object, is the first task displayed as the starting status. The second task is the “reach” task, which requires the subject to extend his arm and open his hand to the object. The third task is the “grasp,” in which the participant holds the object in any position. The task switches to a new state called transport when the user moves the holding object. The fourth task is the “released” task, which occurs when the individual pulls their open palm away from the item. Fig. 5 compares the traditional method to object-centered coordinates with visual attention in HOI recognition.

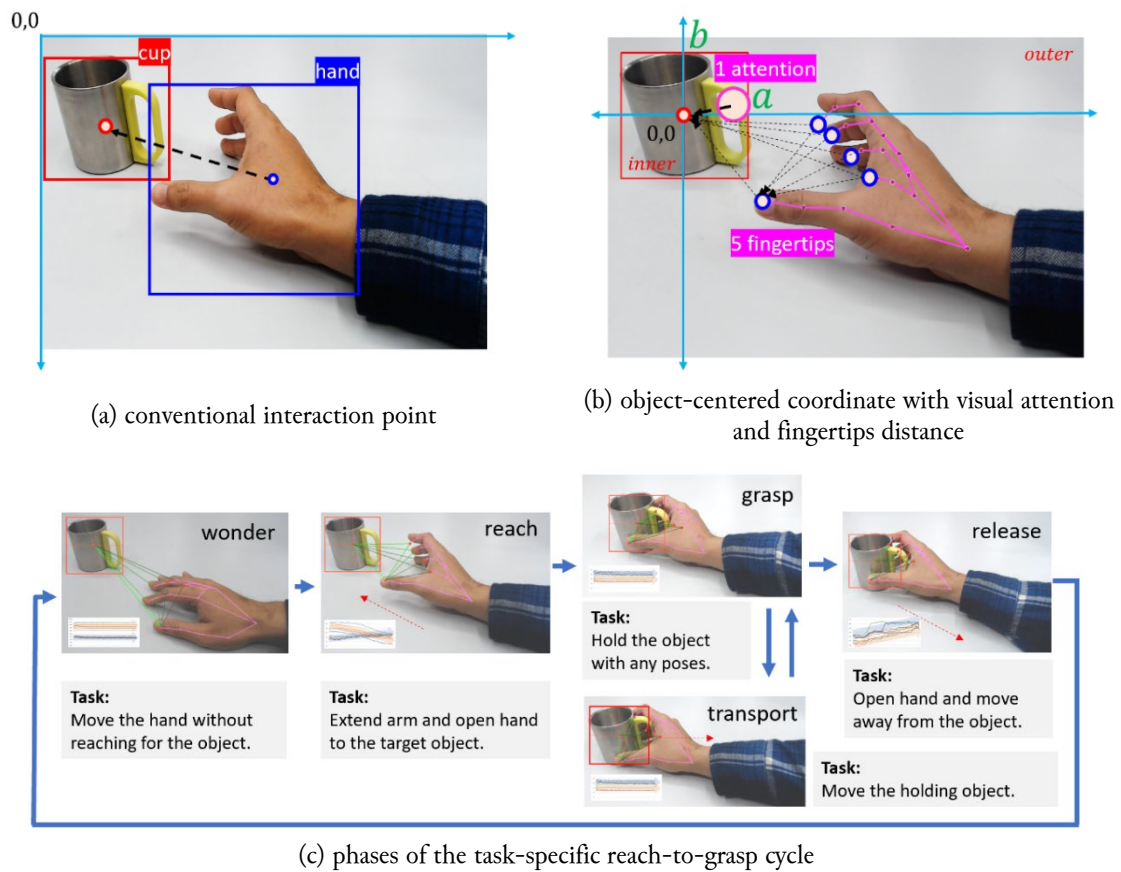


Fig. 5. The comparison method

In our initial stage, we utilize a single object as a reference. We picked a medium-sized cup with various hand postures. We use ten features: five elements of each fingertip’s distance to the center of the object (d_0, d_1, d_2, d_3, d_4), four elements of each fingertip’s distance to the thumb fingertip (e_1, e_2, e_3, e_4), and one visual attention (f_0). We obtain 50 frames per sample using our computer specs, which becomes our benchmark for estimating the length of a data stream. We analyze 10 data points in each picture capture series to obtain a real-time result. In our neural networks, we use this data as input for the learning system.

We use an RNN architecture to classify multivariate time series using an MGRU [32]. The total number of layers, input size, hidden size, and the number of recurrent layers are some variables that can

be changed in this design. To compute each element in every layer of an MGRU, we calculate as the following functions:

$$r_t = \sigma(W_{i,r}x_t + b_{i,r} + W_{h,r}h_{(t-1)} + b_{h,r}), \quad (12)$$

$$z_t = \sigma(W_{i,z}x_t + b_{i,z} + W_{h,z}h_{(t-1)} + b_{h,z}), \quad (13)$$

$$n_t = \sigma_h(W_{i,n}x_t + b_{i,n} + r_t * (W_{h,n}h_{t-1} + b_{h,n})), \quad (14)$$

$$h_t = (1 - z_t) * n_t + z_t * h_{t-1}. \quad (15)$$

The time is symbolized by t . The hidden states are represented by h_t ; the inputs are characterized by x_t , the hidden states of the layers at $t - 1$ is expressed by h_{t-1} or the early hidden states at the initial time o , and r_t, z_t, n_t are the resets, updates, and new gates. The sigmoid function is represented by σ and $*$ symbolized by the product. In the MGRU, the input $x_t^{(l)}$ of the l -th layer ($l \geq 2$) is the hidden states $h_t^{(l-1)}$ of the previous layers multiplied by dropout, $\delta_t^{(l-1)}$ where each $\delta_t^{(l-1)}$ is a Bernoulli random variable 0 with a probability of dropout. Fig. 6 illustrates the mesoscopic level design for multivariate time-series classification using MGRU-RNN.

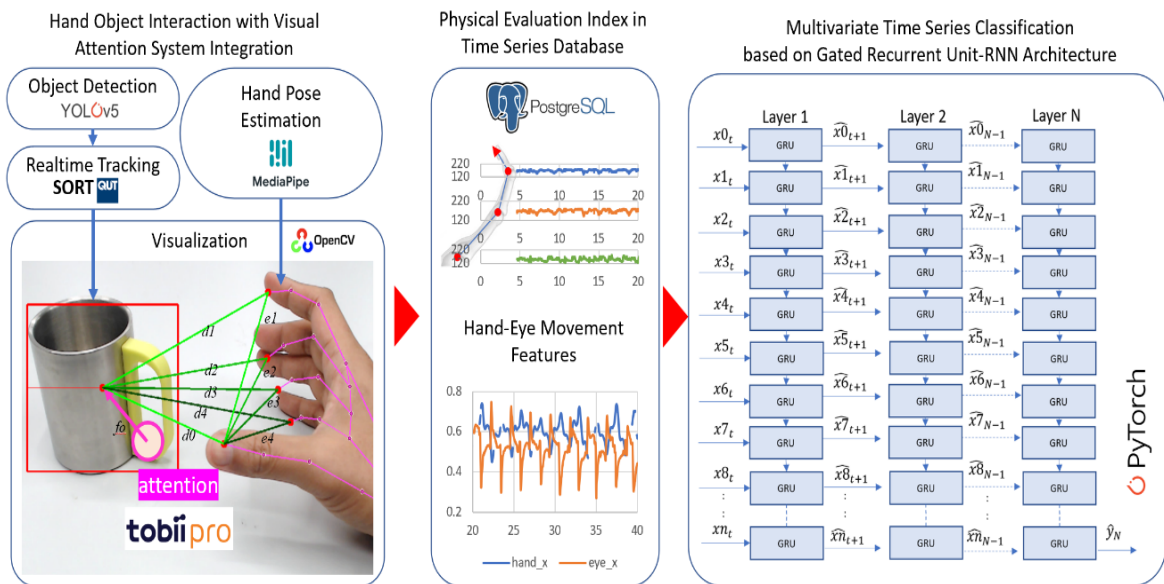


Fig. 6. The mesoscopic level design for multivariate time-series classification using MGRU-RNN

After building the MGRU-based RNN architecture, the next step is to generate a dataset to evaluate each action on HOI recognition. For each sequence, we collect 1–2 seconds of video with a minimum of 50 frames per sample. We compiled a collection of 100 videos of hands interacting with various objects. We shot the video using the following division: 25 data for the wonder, 25 data for the reach, 25 data for the grabbing job involving transportation, and 25 data for the release. We randomly divided the training and validation data into an 80:20 ratio. We decided this distinction was appropriate because the data we obtained was subjective. This experiment includes a responder. The system utilizes 80 movies and 20 videos for instruction. The training and testing outcomes are then discussed in the results and discussion section.

2.4. Macroscopic Level: Cognitive Ability

This subsection investigates the active perception ability development at the mesoscopic level. Using vision-based data collection, we create datasets for an object detection algorithm. The goal is to compile a list of captured objects at a given time. The Tobii Pro eyewear eye tracker sensor analyzes how the human eye responds to different environmental stimuli. Two scenarios are used to demonstrate human interaction with items found in everyday life. The scenario includes table activities, such as eating and

working. This experiment utilizes eleven different objects. In each scenario, we choose a participant to perform these exercises in sequence for approximately 5 minutes. Every time, the camera records these interactions and stores them in the Neo4j graph database. We create a system that generates a network of nodes and their interactions. The system computes each item's frequency of occurrence and relationships with others. The system then deduces the relationship between the objects and adds them to the graph's edge components. Fig. 7 depicts the lifelog graph dataset generation as the input of the macroscopic level.

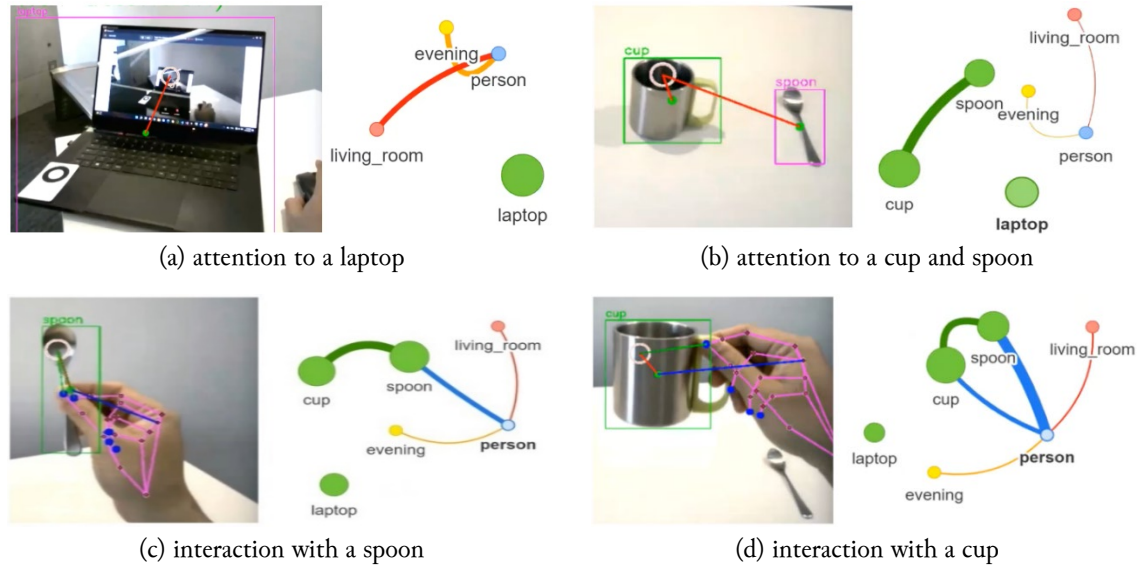


Fig. 7. Lifelog graph dataset generation at macroscopic level

The collected data will be converted into graph data structures. This data representation is increasingly being used to detect connections between nodes. When two objects are connected, the edges show how they relate. Symbolic logic is used to develop a graph. This paradigm's information can be processed by computer programs and stored in graph database structures. As in previous works, we create a graph structure from the collected data to demonstrate the relationship [32]. The spring model is used to create a graph structure from this data. The features can then be assigned to network nodes and edges. GNN recently broadened the scope of datasets using graph-based topologies. We use Kipf and Welling's approach to the GNN convolutional framework to perform node classification [33]. The convolution layer is implemented by passing in the node feature representation and the graph connectivity representation. The macroscopic level design for object classification using the GNN node classifier with three convolutional layers is shown in Fig. 8.

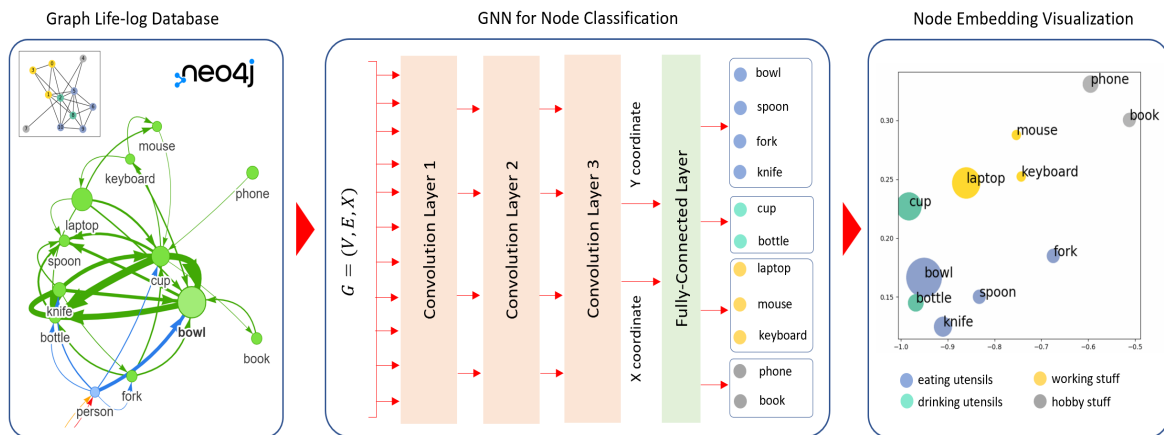


Fig. 8. The macroscopic level design for object classification using the GNN node classifier with three convolutional layers

We initialize the building blocks and define the flow of our network as a forward function. We present three convolution layers that aggregate 3-hop neighborhood information around each node. This layer reduces the node feature dimensionality from the number of the node to the number of the class (i. e., $11 \rightarrow 4 \rightarrow 4 \rightarrow 2$). *Tanh* nonlinearity is used to improve each convolutional layer. After that, we apply a single linear transformation as a classifier to map our nodes to the classes. We return both the final classifier output and the final node embeddings produced by the GNN convolution layer. The node embeddings are then processed by passing the initial node feature X and graph connectivity information V to the model and visualizing with two-dimensional embeddings. It generates an embedding of nodes that closely resembles the structure of the graph before training our model weights. Nodes of the same color are already closely clustered in the embedding space. Before training, the weights of our model are completely randomized. This indicates the conclusion that GNNs introduce a solid inductive bias, leading to similar embeddings for nodes close to each other in the input graph.

We train our network parameters using information about the activities of each node in the graph. We train the model by adding some objects with labels. Because our model is differentiable and parameterized, we observe how the embeddings react. We define a semi-supervised or transductive learning procedure by training against one node per class but are allowed to use the complete input graph data. We use a loss criterion to define our network architecture and start a gradient optimizer. Each round consists of a forward and backward pass to compute the gradient parameters of the loss derived from the forward pass. We compute node embeddings for all of our nodes, but only the training nodes are used to calculate the loss. This is implemented by filtering the output of the classifier out and ground-truth labels data to only contain the nodes in the training mask. The GNN model's three convolutional layers successfully separate the objects and classify the majority of the nodes.

This experiment focused on object classification in our everyday lives. The scenario creates a graph of ten daily items and their relationships in two scenarios of four activities. Next, the object is divided into two groups. In this semi-supervised learning scenario, only a person and single objects are labeled. Next, we discuss the training and testing outcomes in the results and discussion section.

3. Results and Discussion

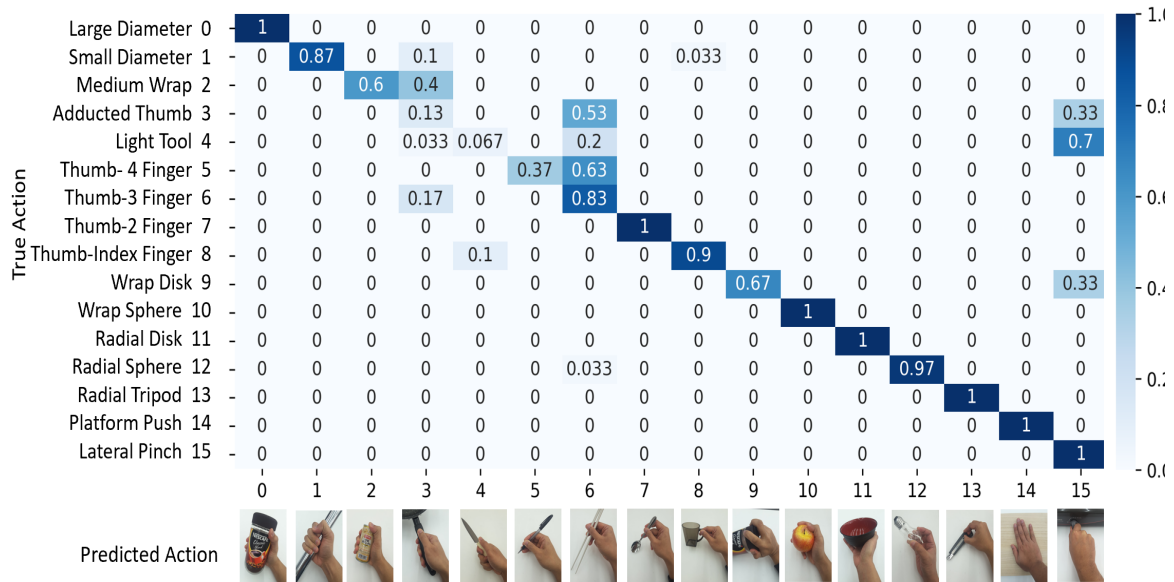
This part describes the results obtained at the microscopic, mesoscopic, and macroscopic levels. Then, we discuss the key points that must be emphasized in the current multiscope system development for future improvement.

3.1. Discussion on Microscopic Level

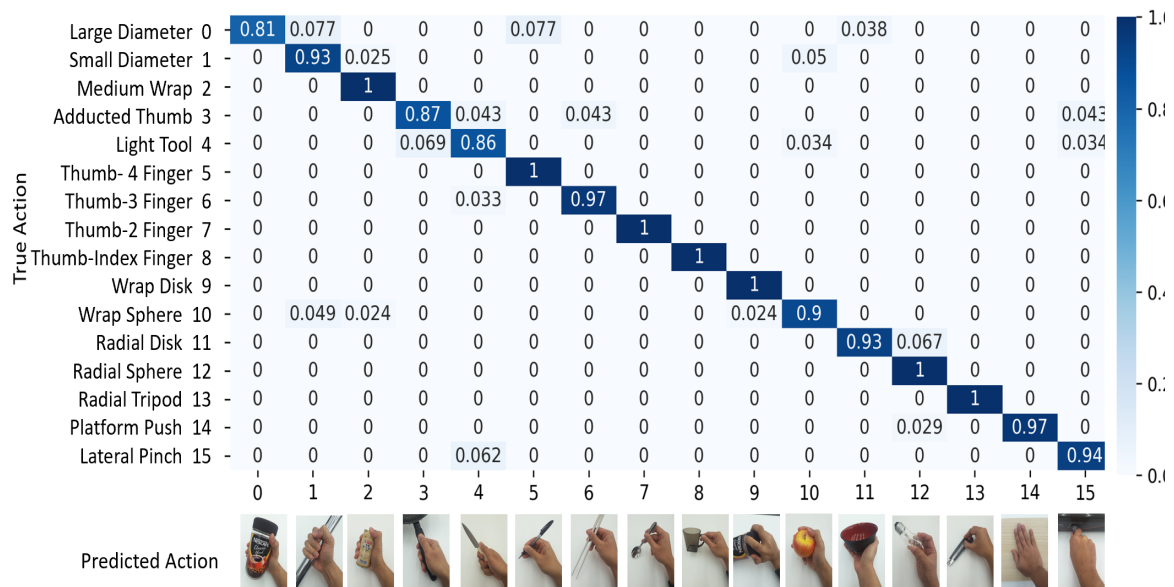
We have developed a microscopic level by designing feature extraction ability using an egocentric vision to observe hand and finger posture. The GNN learning results for classifying 16 grasp poses in a directed graph structure were reported. During training, the grasp acquisition collects input from the system. The loss was less than 0.1 after 1000 epochs. The experiment result demonstrated that the GNN for supervised classification is considered enough to be discussed. The performance of the classification is then validated during testing. The approach integrates the testing dataset into the model. GNN graph classifiers are evaluated by comparing the accuracy of our model's predictions to traditional models such as multi-layer perceptron (MLP). The confusion matrix of grasp poses classification at the 1000th epoch of GNN compared to MLP is shown in [Fig. 9](#).

All training for the GNN and MLP architectures has been completed for categorization. We used the learning outcomes model in the testing dataset to detect grasp posture. The testing accuracy for the 16 grasps pose using the GNN model is 94.87%. This result outperformed the accuracy of the MLP network, which scored 78.75% in our previous work [\[34\]](#). This discovery demonstrates that the gathered dataset of grasp postures contains essential characteristics. We could see that MLP is considered to fail to recognize three classes properly, namely, in classes 3, 4, and 5 (adducted thumb, light tool, thumb-4 finger). The results show that adding a three-layer GNN to the MLP can improve it. This result indicates that GNN has a pretty good accuracy value, between 0.8 to 1, to classify 16 grasp poses. However, there is no way to know how well the model has trained each hand grasp posture. We must test the model on

diverse datasets to estimate its success rate. Hence, we must first decide whether we need power or precision to compare each grasp. These assignments have a variety of similar grab poses. For future developments, the deep learning application in grasp analysis requires hyperparameter optimization. Detailing specific grab postures might boost their accuracy.



(a) MLP testing accuracy (77.75%)



(b) GNN testing accuracy (94.87%)

Fig. 9. The confusion matrix of grasp poses classification at the 1000th epoch

We summarize the basics microscopic level. It is reported in this section that the grasp pose categorization uses angle features. The egocentric vision with a single camera and eye tracker sensor module produced a homogeneous hand skeleton model in the form of a directed graph structure. The grasping posture was observed while the hand interacted with the item that generated the data. The data was collected and covered into angular attributes. The suggested approach was then tested using real-world grab position datasets. The categorization of grasp poses has been tested in a real-time application. Personal datasets, particularly from rehabilitation patients, are required to recognize grasp posture with multiple options for practical applications. In the future, we will employ the suggested technology to acknowledge a person's behavior when grasping the object in rehabilitation.

3.2. Discussion on Mesoscopic Level

Several experiments were carried out to test the proposed frameworks at the mesoscopic level. First, we conducted a single-participant task-specific reach-to-grasp cycle experiment. We extended the egocentric vision feature extraction capabilities discovered in our earlier research [34]. To make it easier to extract these characteristics, we used object-centered coordinate transformation to make it simple to extract these characteristics. Nonetheless, significant technical issues with the extraction occurred throughout the system's deployment. The first issue we discovered was that MediaPipe's hand position evaluation predicted only one frame. Object identification using YOLOv5 in conjunction with object tracking produces less accuracy in certain gripping poses because it does not use previous data. Hand and finger tracking with estimation filters, such as those found in the Leap Motion Controller [35], could be used to address this issue. Another issue is that the collected data is in 2D pixel units, whereas the egocentric approach is in three dimensions. Despite its limitations, the RGB camera may provide consistent results if the captured range is as far as the hand can reach, eliminating the need for precise data, such as millimeters. As a result, additional research may be conducted to improve the spread of this low-cost application.

Second, we evaluate the testing results regarding active perception ability in the task-specific reach-to-grasp action. We trained three RNN models five times: vanilla-RNN, MGRU, and long short-term memory (LSTM). With an average training time of 13.03 seconds (20.48 seconds), the MGRU is expected to outperform the RNN and LSTM in the 50th epoch. All RNN-based learning systems are well classified. The system's accuracy yields the best recognition results, with MGRU averaging 97.0% and 94.0% for LSTM. In this experiment, the MGRU outperformed the standard RNN. MGRU uses fewer tensor operations and takes less training time than LSTM. The results of the three RNN-based models, however, are nearly identical. In real-time testing, the average accuracy of MGRU is 90.75% and LSTM is 77.75%. The confusion matrix of HOI recognition at the 50th epoch of MGRU compared to LSTM is shown in Fig. 10.

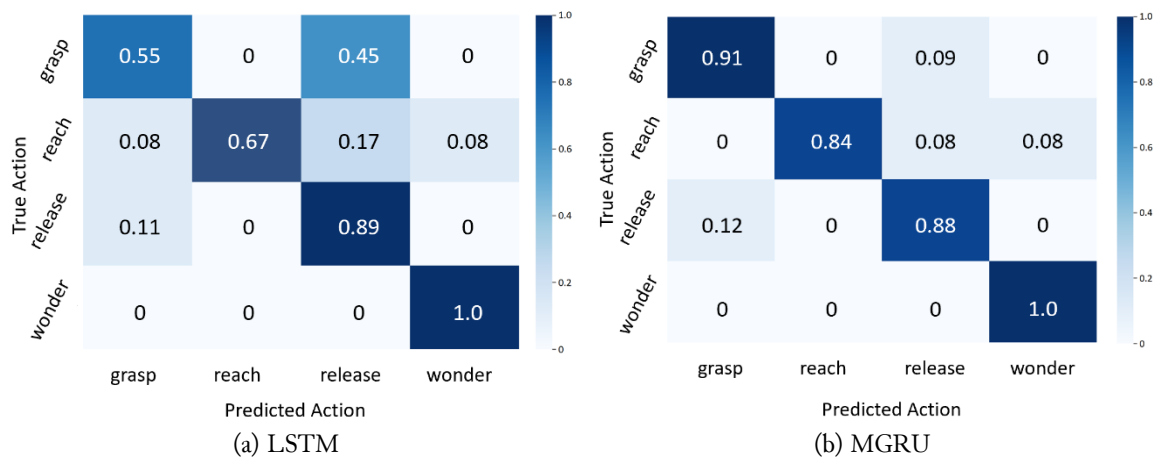


Fig. 10. The confusion matrix of HOI recognition at the 50th epoch

We investigate active perception capability by using an MGRU-based RNN to solve a multivariate time-series classification problem. By benchmarking multivariate time-series classification studies, this MGRU solves the vanishing and rising gradient problem of traditional RNNs [36]. MGRU is rated higher than vanilla-RNN and LSTM. Several studies show that for a simple model, the MGRU model integrates quickly and improves time-series identification performance. Compared with traditional algorithms, like the RNN and the LSTM, this learning technique improves accuracy. Even though we only use a few features for training, we achieve adequate accuracy.

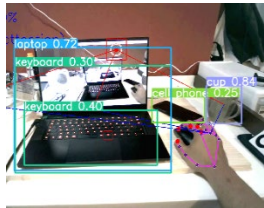
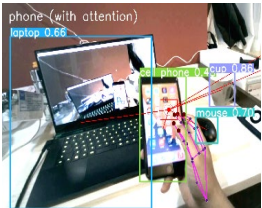
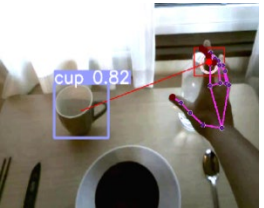
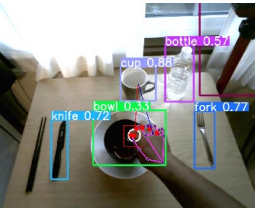
The overall mesoscopic level is summarized. For the task-specific reach-to-grasp cycle, we investigated visual attention to obtain information about hand action from objects. In this study, a new concept for independent rehabilitation in a patient with grasping and vision problems was developed. To

investigate HOI in real-world activities, we developed an egocentric viewpoint. Using active perception and hand skeleton model estimation, we successfully created object grasp detection. Then, we used RNN in conjunction with an MGRU-based architecture to classify essential hand activities. We analyzed our approach using a new dataset for object-grasping behavior. Our research has shown that our proposed method accurately verifies HOI recognition. In the future, we will work with the recommendation system [37] to solve nonstandard grasp pose and object affordance problems. We hope that this study will be used as the foundation of the macroscopic level for diverse and multi-object hand rehabilitation.

3.3. Discussion on Macroscopic Level

We present the macroscopic level learning stage findings in this subsection. We discovered that a GNN algorithm is intended to learn the graph data. Because the network has a small number of nodes and edges, we perform graph learning on each object in several daily activities and compare it with other objects [38]. Table 2 depicts data collection in four activities: (a) working using a laptop, (b) playing with a phone, (c) pouring water from a bottle into a cup, and (d) eating from a bowl. After a few epochs, the GNN learning system could perform semi-supervised classification in four cases. The graphic representation is used to calculate the distance between objects.

Table 2. Data collection in two scenarios with four different activities

Scenario A		Scenario B	
Activity 1	Activity 2	Activity 3	Activity 4
			
Working with a laptop	Playing with a phone	Pouring water into a cup	Taking food from a bowl

These findings imply that the relationship between objects in each scenario influences their class position. The experimental result for classifying some objects using the GNN node classifier is shown in Fig. 11. This diagram shows how the initial epoch can generate node embeddings similar to the graph structure. In the embedding space, nodes of the same color are close together, though some objects still overlap with other classes. The red line represents the laptop's relationship to the closest related object. The three-layer GNN model separated the communities linearly and correctly classified the nodes. Objects from Activities 1 and 2 appear close together as objects in Activities 3 and 4. As a result, after the 100th epoch, it is clear that GNN can separate two clusters that are far apart. This occurs because working on laptops and playing with phones occur in the same work environment as eating and drinking at the dining table.

The entire macroscopic level is summarized. We addressed graph learning research for object classification in the application. Based on the accumulated graph, we can observe how the system evolves. All interactions can be described as knowledge domains at the macroscopic level. A GNN application with weights on each attribute is required for input graphs with various contexts [39]. This method's goals are for semi-supervised categorization in daily activities related to objects. This system requires more dispersed personal datasets for real-world applications. This object classification method should be tested in several daily activities in real time. These technologies will be developed as a result of this work for future cognitive rehabilitation. Environmental constraints must be considered to tailor the rehabilitation system to specific issues. We hope that this study will help therapists and researchers by providing information unavailable in the clinic. We hope to collect patient samples for further validation and use this technology for rehabilitation in the future.

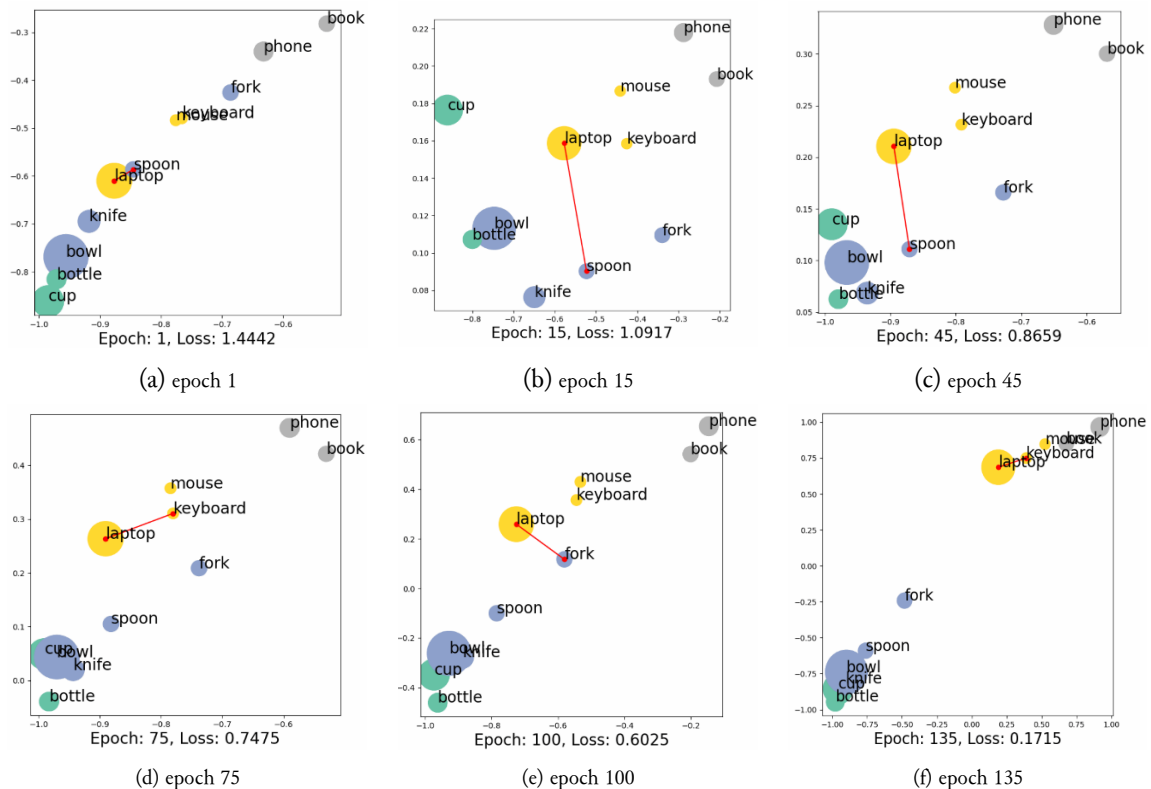


Fig. 11. The experimental result for classifying some objects using the GNN node classifier

4. Conclusion

This paper proposed HOI recognition based on visual attention using multiscope CPSS. Feature extraction capacity utilizing an egocentric vision has been designed to observe hand and finger posture at the microscopic level. The GNN successfully enhanced the MLP in classifying hand grasp pose with 94.87% average accuracy. At the mesoscopic level, an active perception ability has been proposed to validate HOI recognition with eye tracking in the task-specific reach-to-grasp cycle. Objects with hand skeletal tracking were combined as inputs to MGRU, which is based on RNN architecture and has 90.75% average accuracy in categorizing hand interactions with objects. At the macroscopic level, cognitive ability has been implemented by adding visual attention to describe human behavior when interacting with multiple objects. GNN node classifiers can differentiate between two scenarios with four main activities. The outcome demonstrates that the system can successfully separate some objects based on related activities. Further research is expected to benefit independent rehabilitation support and boost community self-efficacy.

Acknowledgment

The authors would like to acknowledge the scholarship support provided by the Japan Ministry of Education, Culture, Sports, Science, and Technology (MEXT). This work was partially supported by Japan Science and Technology Agency (JST), Moonshot R&D, with grant number JPMJMS2034, and Tokyo Metropolitan University (TMU) Local 5G research support.

Declarations

Author contribution. All authors contributed equally and approved the final paper.

Funding statement. This research is funded by Japan Science and Technology Agency (JST), Moonshot R&D, with grant number JPMJMS2034, and TMU local 5G research support.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

Data and Software Availability Statements

We develop applications using Python 3.8, Windows 11 operating system, and open-source frameworks, including OpenCV 4.6.0 for standard computer vision applications, YOLOv5 and SORT for tracking objects, and Mediapipe 0.8.10.1 for tracking hands. For the learning environment, we use Pytorch 1.8.2 with several additional graph learning features utilizing Pytorch Geometric 2.0.4. We employ the Tobii Glasses 1.12.11 software for the eye tracker sensor reader and the RTSP protocol. We use PostgreSQL as the time-series database and Neo4j as the graph database. The code and data set can be accessed at <https://github.com/anom-tmu/hoi-attention>.

Appendix

Appendix 1 shows the commonly used notations in this paper.

Appendix 1. Commonly used notations.

Notations	Descriptions
G	A graph $G \in \mathcal{G}$.
\mathcal{G}	The set of graphs.
V	The set of nodes in a graph.
v, u	A node $v, u \in V$
X	The set of node features in a graph
x_v	A feature vector in a node v
E	The set of edges in a graph
$e_{i,j}$	An edge $e_{i,j} \in E$
$N(v)$	The neighbors of a node v
$h_{G,v}$	The embedding vector of a node v in a graph G
$m_{G,v}$	The embedding vector of aggregation result
H_G	The embedding vector of a graph G
W	The set of weight / learnable model parameter
f	A function
k, K	The layer index
t, T	The time step/iteration index
i, j	The dimension of weight matrix
$\sigma(\cdot)$	The activation function
$ \cdot $	The length of a set

References

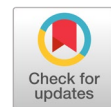
- [1] T. Singh, C. M. Perry, S. L. Fritz, J. Fridriksson, and T. M. Herter, "Eye Movements Interfere With Limb Motor Control in Stroke Survivors," *Neurorehabil. Neural Repair*, vol. 32, no. 8, pp. 724–734, Aug. 2018, doi: [10.1177/1545968318790016](https://doi.org/10.1177/1545968318790016).
- [2] M. Szekeres and K. Valdes, "Virtual health care & telehealth: Current therapy practice patterns," *J. Hand Ther.*, vol. 35, no. 1, pp. 124–130, Jan. 2022, doi: [10.1016/j.jht.2020.11.004](https://doi.org/10.1016/j.jht.2020.11.004).
- [3] P. Wang, L. T. Yang, J. Li, J. Chen, and S. Hu, "Data fusion in cyber-physical-social systems: State-of-the-art and perspectives," *Inf. Fusion*, vol. 51, pp. 42–57, Nov. 2019, doi: [10.1016/j.inffus.2018.11.002](https://doi.org/10.1016/j.inffus.2018.11.002).
- [4] A. Laghari, Z. A. Memon, S. Ullah, and I. Hussain, "Cyber Physical System for Stroke Detection," *IEEE Access*, vol. 6, pp. 37444–37453, Jun. 2018, doi: [10.1109/ACCESS.2018.2851540](https://doi.org/10.1109/ACCESS.2018.2851540).
- [5] A. Rashid and O. Hasan, "Wearable technologies for hand joints monitoring for rehabilitation: A survey," *Microelectronics J.*, vol. 88, pp. 173–183, Jun. 2019, doi: [10.1016/j.mejo.2018.01.014](https://doi.org/10.1016/j.mejo.2018.01.014).

- [6] A. A. Saputra, A. R. A. Besari, and N. Kubota, "Human Joint Skeleton Tracking Using Multiple Kinect Azure," in *2022 International Electronics Symposium (IES)*, Aug. 2022, pp. 430–435, doi: [10.1109/IES55876.2022.9888532](https://doi.org/10.1109/IES55876.2022.9888532).
- [7] M. Dousty and J. Zariffa, "Tenodesis Grasp Detection in Egocentric Video," *IEEE J. Biomed. Heal. Informatics*, vol. 25, no. 5, pp. 1463–1470, May 2021, doi: [10.1109/JBHI.2020.3003643](https://doi.org/10.1109/JBHI.2020.3003643).
- [8] M. Cai, K. Kitani, and Y. Sato, "Understanding hand-object manipulation by modeling the contextual relationship between actions, grasp types and object attributes," pp. 1–14, July. 2018. [Online]. Available at: <https://arxiv.org/abs/1807.08254v1>.
- [9] M.-F. Tsai, R. H. Wang, and J. Zariffa, "Identifying Hand Use and Hand Roles After Stroke Using Egocentric Video," *IEEE J. Transl. Eng. Heal. Med.*, vol. 9, pp. 1–10, 2021, doi: [10.1109/JTEHM.2021.3072347](https://doi.org/10.1109/JTEHM.2021.3072347).
- [10] A. R. A. Besari, A. A. Saputra, W. H. Chin, N. Kubota, and Kurnianingsih, "Hand-Object Interaction Detection based on Visual Attention for Independent Rehabilitation Support," in *2022 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2022, vol. 2022-July, pp. 1–6, doi: [10.1109/IJCNN55064.2022.9892903](https://doi.org/10.1109/IJCNN55064.2022.9892903).
- [11] T. Wang, T. Yang, M. Danelljan, F. S. Khan, X. Zhang, and J. Sun, "Learning Human-Object Interaction Detection Using Interaction Points," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 4115–4124, doi: [10.1109/CVPR42600.2020.00417](https://doi.org/10.1109/CVPR42600.2020.00417).
- [12] D. Qurratu'aini, A. Sophian, W. Sediono, H. Md Yusof, and S. Sudirman, "Visual-Based Fingertip Detection for Hand Rehabilitation," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 9, no. 2, p. 474, Feb. 2018, doi: [10.11591/ijeecs.v9.i2.pp474-480](https://doi.org/10.11591/ijeecs.v9.i2.pp474-480).
- [13] J. Likitlersuang, E. R. Sumitro, T. Cao, R. J. Visée, S. Kalsi-Ryan, and J. Zariffa, "Egocentric video: a new tool for capturing hand use of individuals with spinal cord injury at home," *J. Neuroeng. Rehabil.*, vol. 16, no. 1, p. 83, Dec. 2019, doi: [10.1186/s12984-019-0557-1](https://doi.org/10.1186/s12984-019-0557-1).
- [14] R. J. Visee, J. Likitlersuang, and J. Zariffa, "An Effective and Efficient Method for Detecting Hands in Egocentric Videos for Rehabilitation Applications," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 3, pp. 748–755, Mar. 2020, doi: [10.1109/TNSRE.2020.2968912](https://doi.org/10.1109/TNSRE.2020.2968912).
- [15] A. Bandini, M. Dousty, S. L. Hitzig, B. C. Craven, S. Kalsi-Ryan, and J. Zariffa, "Measuring Hand Use in the Home after Cervical Spinal Cord Injury Using Egocentric Video," *J. Neurotrauma*, vol. 39, no. 23–24, pp. 1697–1707, Dec. 2022, doi: [10.1089/neu.2022.0156](https://doi.org/10.1089/neu.2022.0156).
- [16] J. Xu, P. Mohan, F. Chen, and A. Nurnberger, "A Real-time Hand Motion Detection System for Unsupervised Home Training," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct. 2020, vol. 2020-October, pp. 4224–4229, doi: [10.1109/SMC42975.2020.9283261](https://doi.org/10.1109/SMC42975.2020.9283261).
- [17] Y. Li, L. Jia, Z. Wang, Y. Qian, and H. Qiao, "Un-supervised and semi-supervised hand segmentation in egocentric images with noisy label learning," *Neurocomputing*, vol. 334, pp. 11–24, Mar. 2019, doi: [10.1016/j.neucom.2018.12.010](https://doi.org/10.1016/j.neucom.2018.12.010).
- [18] Y. Lee, W. Do, H. Yoon, J. Heo, W. Lee, and D. Lee, "Visual-inertial hand motion tracking with robustness against occlusion, interference, and contact," *Sci. Robot.*, vol. 6, no. 58, Sep. 2021, doi: [10.1126/scirobotics.abe1315](https://doi.org/10.1126/scirobotics.abe1315).
- [19] G. Kapidis, R. Poppe, and R. C. Veltkamp, "Multi-Dataset, Multitask Learning of Egocentric Vision Tasks," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, 2021, pp. 1–1, doi: [10.1109/TPAMI.2021.3061479](https://doi.org/10.1109/TPAMI.2021.3061479).
- [20] K. Hesseberg, G. G. Tangen, A. H. Pripp, and A. Bergland, "Associations between Cognition and Hand Function in Older People Diagnosed with Mild Cognitive Impairment or Dementia," *Dement. Geriatr. Cogn. Dis. Extra*, vol. 10, no. 3, pp. 195–204, Dec. 2020, doi: [10.1159/000510382](https://doi.org/10.1159/000510382).
- [21] J. Jiang, Z. Nan, H. Chen, S. Chen, and N. Zheng, "Predicting short-term next-active-object through visual attention and hand position," *Neurocomputing*, vol. 433, pp. 212–222, Apr. 2021, doi: [10.1016/j.neucom.2020.12.069](https://doi.org/10.1016/j.neucom.2020.12.069).

- [22] R. Tanaka, J. Woo, and N. Kubota, "Nonverbal Communication Based on Instructed Learning for Socially Embedded Robot Partners," *J. Adv. Comput. Intell. Informatics*, vol. 23, no. 3, pp. 584–591, May 2019, doi: [10.20965/jaciii.2019.p0584](https://doi.org/10.20965/jaciii.2019.p0584).
- [23] M. Yani, A. R. A. Besari, N. Yamada, and N. Kubota, "Ecological-Inspired System Design for Safety Manipulation Strategy in Home-care Robot," in *2020 International Symposium on Community-centric Systems (CcS)*, Sep. 2020, pp. 1–6, doi: [10.1109/CcS49175.2020.9231354](https://doi.org/10.1109/CcS49175.2020.9231354).
- [24] A. R. A. Besari, A. A. Saputra, W. H. Chin, Kurnianingsih, and N. Kubota, "Finger Joint Angle Estimation With Visual Attention for Rehabilitation Support: A Case Study of the Chopsticks Manipulation Test," *IEEE Access*, vol. 10, no. September, pp. 91316–91331, 2022, doi: [10.1109/ACCESS.2022.3201894](https://doi.org/10.1109/ACCESS.2022.3201894).
- [25] A. A. Saputra, K. Wada, S. Masuda, and N. Kubota, "Multi-scopic neuro-cognitive adaptation for legged locomotion robots," *Sci. Rep.*, vol. 12, no. 1, p. 16222, Sep. 2022, doi: [10.1038/s41598-022-19599-2](https://doi.org/10.1038/s41598-022-19599-2).
- [26] K. Oshio, K. Kaneko, and N. Kubota, "Multi-scopic Simulation for Human-robot Interactions Based on Multi-objective Behavior Coordination," in *International Workshop on Advanced Computational Intelligence and Intelligent Informatics*, 2021, no. Iwaciii, pp. 3–8. [Online]. Available at: <https://iwaciii2021.bit.edu.cn/docs/2021-12/b3d6c84e7e244c6e89cf502ed15cdc17.pdf>.
- [27] P. Pradhymna, G. P. Shreya, and Mohana, "Graph Neural Network (GNN) in Image and Video Understanding Using Deep Learning for Computer Vision Applications," in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Aug. 2021, pp. 1183–1189, doi: [10.1109/ICESC51422.2021.9532631](https://doi.org/10.1109/ICESC51422.2021.9532631).
- [28] Y. J. R. De Kloe, I. T. C. Hooge, C. Kemner, D. C. Niehorster, M. Nyström, and R. S. Hessels, "Replacing eye trackers in ongoing studies: A comparison of eye-tracking data quality between the Tobii Pro TX300 and the Tobii Pro Spectrum," *Infancy*, vol. 27, no. 1, pp. 25–45, Jan. 2022, doi: [10.1111/inf.12441](https://doi.org/10.1111/inf.12441).
- [29] H. Fu, L. Wu, M. Jian, Y. Yang, and X. Wang, "MF-SORT: Simple Online and Realtime Tracking with Motion Features," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11901 LNCS, Springer, 2019, pp. 157–168, doi: [10.1007/978-3-030-34120-6_13](https://doi.org/10.1007/978-3-030-34120-6_13).
- [30] V. Chundururu, M. Roy, D. R. N. S, and R. G. Chittawadigi, "Hand Tracking in 3D Space using MediaPipe and PnP Method for Intuitive Control of Virtual Globe," in *2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC)*, Sep. 2021, pp. 1–6, doi: [10.1109/R10-HTC53172.2021.9641587](https://doi.org/10.1109/R10-HTC53172.2021.9641587).
- [31] A. R. Anom Besari, W. H. Chin, N. Kubota, and Kurnianingsih, "Ecological Approach for Object Relationship Extraction in Elderly Care Robot," in *2020 21st International Conference on Research and Education in Mechatronics (REM)*, Dec. 2020, pp. 1–6, doi: [10.1109/REM49740.2020.9313877](https://doi.org/10.1109/REM49740.2020.9313877).
- [32] R. Volcic and F. Domini, "The endless visuomotor calibration of reach-to-grasp actions," *Sci. Rep.*, vol. 8, no. 1, p. 14803, Oct. 2018, doi: [10.1038/s41598-018-33009-6](https://doi.org/10.1038/s41598-018-33009-6).
- [33] A. Pareja *et al.*, "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 04, pp. 5363–5370, Apr. 2020, doi: [10.1609/aaai.v34i04.5984](https://doi.org/10.1609/aaai.v34i04.5984).
- [34] A. R. Anom Besari, A. A. Saputra, W. H. Chin, N. Kubota, and Kurnianingsih, "Feature-based Egocentric Grasp Pose Classification for Expanding Human-Object Interactions," in *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, Jun. 2021, vol. 2021-June, pp. 1–6, doi: [10.1109/ISIE45552.2021.9576369](https://doi.org/10.1109/ISIE45552.2021.9576369).
- [35] A. Vysocký *et al.*, "Analysis of Precision and Stability of Hand Tracking with Leap Motion Sensor," *Sensors*, vol. 20, no. 15, p. 4088, Jul. 2020, doi: [10.3390/s20154088](https://doi.org/10.3390/s20154088).
- [36] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Min. Knowl. Discov.*, vol. 31, no. 3, pp. 606–660, May 2017, doi: [10.1007/s10618-016-0483-9](https://doi.org/10.1007/s10618-016-0483-9).
- [37] H. Hanafi, N. Suryana, and A. S. H. Basari, "Dynamic convolutional neural network for eliminating item sparse data on recommender system," *Int. J. Adv. Intell. Informatics*, vol. 4, no. 3, p. 226, Nov. 2018, doi: [10.26555/ijain.v4i3.291](https://doi.org/10.26555/ijain.v4i3.291).

-
- [38] R. Tanaka, J. Woo, and N. Kubota, "Action Acquisition Method for Constructing Cognitive Development System Through Instructed Learning," in *2019 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2019, vol. 2019-July, pp. 1–6, doi: [10.1109/IJCNN.2019.8852180](https://doi.org/10.1109/IJCNN.2019.8852180).
- [39] G. H. Martono, A. Azhari, and K. Mustofa, "An extended approach of weight collective influence graph for detection influence actor," *Int. J. Adv. Intell. Informatics*, vol. 8, no. 1, p. 1, Mar. 2022, doi: [10.26555/ijain.v8i1.800](https://doi.org/10.26555/ijain.v8i1.800).

Multi-granularity active learning based on the three-way decision



Wu Xiaogang ^{a,b,1,*}, Thitipong ^{a,2}

^a Vincent Mary School of Science & Technology, Assumption University, Bangkok, Thailand

^b School of Information Technology, Xingyi Normal University for Nationalities, Xingyi, China

¹ wkg817@163.com; ² thitipong@scitech.au.edu

* corresponding author

ARTICLE INFO

Article history

Received February 21, 2023

Revised March 25, 2023

Accepted April 7, 2023

Available online May 4, 2023

Keywords

Three-way decision

Multi-grained features

Active learning

Unlabeled samples

Classification algorithm

ABSTRACT

The reliance on data and the high cost of data labeling are the main problems facing deep learning today. Active learning aims to make the best model with as few training samples as possible. Previous query strategies for active learning have mainly used the uncertainty and diversity criteria, and have not considered the data distribution's multi-granularity. To extract more valid information from the samples, we use three-way decisions to select uncertain samples and propose a multi-granularity active learning method (MGAL). The model divides the unlabeled samples into three parts: positive, negative, and boundary region. Through active iterative training samples, the decision delay of the boundary domain can reduce the decision cost. We validated the model on five UCI datasets and the CIFAR10 dataset. The experimental results show that the cost of three-way decisions is lower than that of two-way decisions. The multi-granularity active learning achieves good classification results, which validates the model. In this case study, the reader can learn about the ideas and methods of the three-way decision theory applied to deep learning.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Deep learning networks have made extraordinary progress in artificial intelligence [1]–[4]. Multi-granularity structure is an important feature of deep networks, whether it is image, text, or speech data, where feature representations can be extracted at different granularities [5], [6]. Deep networks extract finer-grained features by processing data from low to high and building multi-level, multi-granular semantics [7]–[9]. Deep networks consist of multiple layers of small groups of neurons (e.g. convolutional kernels, etc.), each of which processes only a portion of the input image. In general, the lower the underlying layer, the smaller the receptive field and the finer the granularity of the features. The features of each layer are combined into a new receptive field, usually more significant than the receptive field of the previous layer, and this is the granularity of the features.

As in Fig. 1, the VGGNet16 [10] network goes through multiple layers of feature extraction to obtain a high-level representation of an image. Res2Net [11] is an improved multi-level, multi-granularity residual network. Fig. 2 shows the structure of each residual unit to obtain different features f_i ($i=1,2,\dots,s$) by fusion to increase the multi-granularity feature representation capability of the network. The hierarchical pyramid structure of the FPN network [12] mines multi-grained features of images and fuses the information of multi-grained features to obtain rich and powerful high-level features (Fig. 3). Training deep learning networks requires a lot of data and tag costs [13]. Active learning can reduce labeling costs by selecting the most informative samples of the dataset [14], [15].

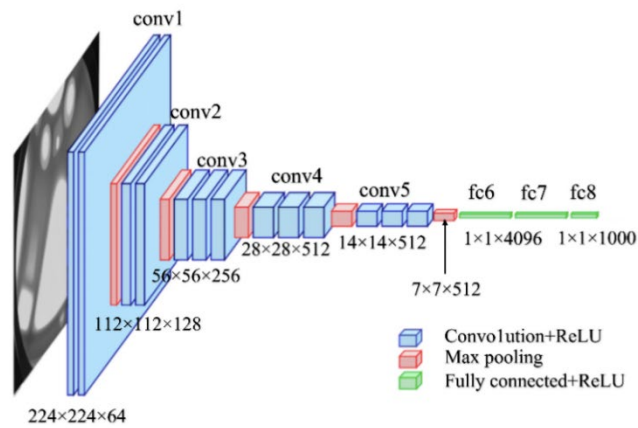


Fig. 1. Multi-grain convolutional features of VGG networks

Unlike traditional passive supervised learning, active learning methods train on labeled samples to obtain a priori knowledge, which is then used to evaluate the value of unlabeled samples. Therefore, how to efficiently select unclassified labeled samples with high classification contributions for annotation and add them to the existing training set is a key issue in active learning.

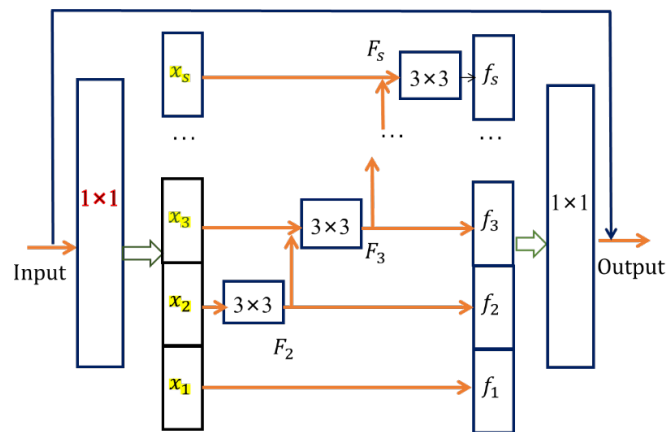


Fig. 2. Multi-granularity feature extraction with residual module

However, in active learning, the arbitrary uncertainty of the recognition samples may also lead to higher computational costs in iterative retraining [16]. To speed up training and extract features more efficiently, we introduce a three-way decision approach [17] in active learning networks, where each batch of new labels is trained incrementally and multi-granularity feature extraction of uncertain samples is performed using three-way decisions, which greatly reduces the computational demand of active learning methods.

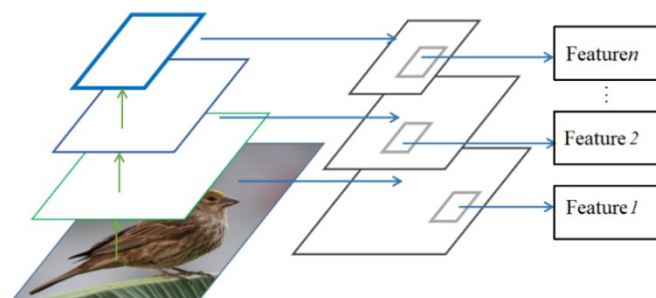


Fig. 3. Pyramidal multi-grain feature extraction for FPN networks

Three-way decision theory [18] is a multi-granularity approach to solving complex problems. It complements and extends classical two-way decision-making; It simulates solving problems for human minds by determining the acceptance and rejection domains and taking delayed confirmation decisions for inaccurate or uncertain information. By gathering more granular data, the uncertainty part is transformed into the realm of certainty (the domain and rejection), thereby reducing its inconsistency and improving the accuracy of decisions.

In this paper, our contributions are mainly as follows:

- A novel model combining deep and active learning is proposed to achieve dynamic incremental image recognition and classification of data.
- The sample selection strategy of active learning is improved to reduce the cost of iterative training.
- The hierarchical and multi-granularity characteristics of decision queries are achieved. Using three-way decision-making, representative samples from uncertain samples can be effectively extracted, improving the performance of the classifier.

Following the Introduction section, Section 2 describes the three-way decision and active learning algorithms. In Section 3, we discuss the network model parameters and experimental results, and Section 4 concludes the paper.

2. Method

2.1. Three-way decision

The three-way decision is a way to model a multi-level decision by dividing the whole into three distinct and related parts [19]. Humans can immediately judge entirely accepted and rejected; research and learning are required to make final judgments for uncertain things. The inaccuracy of an indefinite item is due to its different granularity. The solution to this uncertainty can be a granularity transformation of its attributes to refine the coarse granularity, thus making the uncertain object a definite object.

To implement the three-way decision, it is first necessary to introduce the decision function $f(x)$ of the entity, also called the evaluation function; then, a pair of threshold values α and β is introduced, and the event objects in the argument domain U are divided into positive $POS(U)$, boundary $BND(U)$ and negative domains $NEG(U)$ according to the decision state value and threshold value, corresponding to the acceptance, deferral and rejection rules of the three-way decision. (α, β) denotes the upper and lower approximation. The decision rules are defined as follows:

$$\begin{aligned} & \text{if } f(x) \geq \alpha, \text{ then } x \in POS(U) \\ & \text{if } \alpha < f(x) < \beta, \text{ then } x \in BND(U) \\ & \text{if } f(x) \leq \beta, \text{ then } x \in NEG(U) \end{aligned} \quad (1)$$

For a sample set U , C represents the description of the state U . For each sample $x \in U$, $f(x)$ represents the evaluation of the current definition of $Des(x)$. The three decision steps are: targets that satisfy C are assigned to $POS(U)$, those that do not are assigned to $NEG(U)$ and those that are difficult to judge are assigned to $BND(U)$. The cost of classifying a sample into the three domains is different. Assuming that S_P represents a sample in the positive domain and S_N represents a sample in the negative field, these two fields are divided into three regions, resulting in six different classification cost functions, as in (2).

$$\begin{aligned} \lambda_{PP} &= \lambda\{POS|L(x) = S_p\}, \lambda_{PN} = \lambda\{POS|L(x) = S_N\}, \\ \lambda_{NP} &= \lambda\{NEG|L(x) = S_p\}, \lambda_{NN} = \lambda\{NEG|L(x) = S_N\}, \end{aligned}$$

$$\lambda_{BP} = \lambda\{BND|L(x) = S_p\}, \lambda_{BN} = \lambda\{BND|L(x) = S_N\} \tag{2}$$

Where $L(x)$ is the conditional probability of classifying sample x into the positive domain, and the expected risk of classifying sample x into the three domains can be calculated using (3).

$$\begin{aligned} C(POS|x) &= \lambda_{PP}f(x) + \lambda_{PN}(1-f(x)) \\ C(NEG|x) &= \lambda_{NP}f(x) + \lambda_{NN}(1-f(x)) \\ C(BND|x) &= \lambda_{BP}f(x) + \lambda_{BN}(1-f(x)) \end{aligned} \tag{3}$$

In general, the cost of correctly classifying a positive domain sample into its region is small or even negligible, and the cost of delaying a decision on a sample should be lower than the cost of misclassification, i.e. :

$$\lambda_{PP} < \lambda_{BP} < \lambda_{NP}, \lambda_{NN} < \lambda_{BN} < \lambda_{PN} \tag{4}$$

Furthermore, the cost of incorrectly accepting a negative domain should be much higher than the cost of incorrectly rejecting a positive example, and based on the above equation, the decision threshold can be obtained as follows :

$$\alpha = \frac{\lambda_{PN} - \lambda_{BN}}{\lambda_{PN} - \lambda_{BN} + \lambda_{BP} - \lambda_{PP}}, \beta = \frac{\lambda_{BN} - \lambda_{NN}}{\lambda_{BN} - \lambda_{BN} + \lambda_{NP} - \lambda_{NN}} \tag{5}$$

The cost of partitioning the sample into boundary regions varies with the number of three-way decision steps. The richer the information, the less acceptable the delayed decision and the higher the cost of making the delayed decision. As the cost of delayed choices increases, the extent of the boundary region gradually shrinks or even disappears, and the three-way decision degenerates into a two-way decision.

2.2. Multi-granularity three-way decision model

The sequential three-way decision is a multi-granular decision model (Fig. 4) dealing with dynamic decision problems [20], [21].

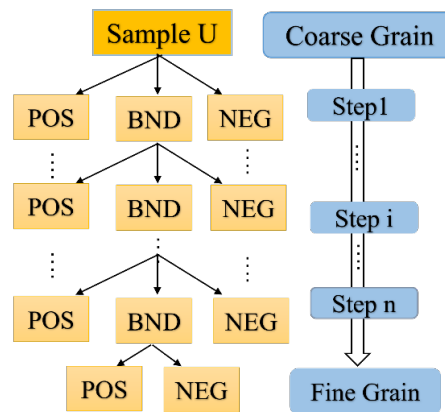


Fig. 4. Multi-grained decision model with sequential three-way decision

Assuming that the n -step decisions constitute n levels of granularity, $Des_i(x), f_i(x)$ represent the state description of sample x at step i and the evaluation of the decision maker, respectively, and the increasing accuracy of these evaluations as in (6).

$$(Des_1(x), f_1(x)) \preceq (Des_2(x), f_2(x)) \preceq \dots \preceq (Des_n(x), f_n(x)) \tag{6}$$

This constitutes a step-by-step cognitive process of moving from coarse to fine granularity of the target.

For objects divided into boundary regions, only the last layer uses a two-way decision, and the other granular layers use a three-way decision. Therefore, a reasonable decision threshold is set on each grain layer. It is a multi-granularity decision approach where the cost of each delayed decision step increases as i increases. The threshold value for each decision step is given in (7).

$$\alpha_i = \frac{\lambda_{PN} - \lambda_{BN}^i}{(\lambda_{PN} - \lambda_{BN}^i) + (\lambda_{BP}^i - \lambda_{PP})}$$

$$\beta_i = \frac{\lambda_{BP}^i - \lambda_{NN}}{(\lambda_{BN}^i - \lambda_{NN}) + (\lambda_{NP} - \lambda_{BP}^i)} \quad (7)$$

The final step of the three-way decision has $\alpha_n = \beta_n$, and assuming that $\lambda_{BN}^i / \lambda_{BP}^i = C$, the final threshold obtained can be calculated as follows:

$$\lambda_{BP}^n = \frac{\lambda_{PN}\lambda_{NP} - \lambda_{PP}\lambda_{NN}}{(\lambda_{PN} - \lambda_{NN}) + C(\lambda_{NP} - \lambda_{PP})}$$

$$\lambda_{BN}^n = C \frac{\lambda_{PN}\lambda_{NP} - \lambda_{PP}\lambda_{NN}}{(\lambda_{PN} - \lambda_{NN}) + C(\lambda_{NP} - \lambda_{PP})} \quad (8)$$

Fig. 5 shows the multi-granularity three-way decisions Classification algorithm. The Input was imbalanced dataset, attributes of multi-granularity information, decision cost thresholds of different granularity (α_i, β_i).

```

1   Initialize U1=U; POS=NEG=BND=∅
2   for i=1,2,...,n do
3     P(X|[xi])=P(X)P([xi|X])/P([xi]);
4     if(P(X|[xi] ≥ αi)
5       POS=POS ∪ POSi
6     else if(|P(X|[xi] ≤ βi)
7       NEG=NEG ∪ NEGi
8     else
9       Ui+1=BNDi
10    i=i+1
11  end for
12  if Un ≠ ∅ then
13    POSi = {x ∈ Un | P(X|[xi] ≥ γi};
14    NEGi = {x ∈ Un | P(X|[xi] < γi};
15    POS=POS ∪ POSi
16    NEG=NEG ∪ NEGi
17  End if

```

Fig. 5. Multi-granularity three-way decisions Classification algorithm

2.3. Active learning

The active learning method interacts with the classifier and the expert to obtain high-value samples to improve the classifier's performance. It avoids redundancy and unnecessary additions to data and reduces the cost of tagging large amounts of data [22]. Active learning is divided into two steps: 1) Sample selection: select the most valuable sample marks and then add them to the training set; 2) Model training: supervised learning is carried out to measure the classification performance of the classifier. In the active learning process, the two steps are executed alternately in an iterative. The algorithm terminates when a preset number of iterations or a preset classification performance is reached, as shown in Fig. 6.

In Fig. 6, an initial classifier must be trained by obtaining a portion of the used samples at the beginning. The selected examples will significantly influence the evaluation of the initial classifier.

Suppose the initial classifier has a certain level of accuracy. Suppose the initial classifier has a sure accuracy. In that case, it can minimize inefficient markers, speed up the active learning process to a certain extent, and achieve the same performance with fewer samples.

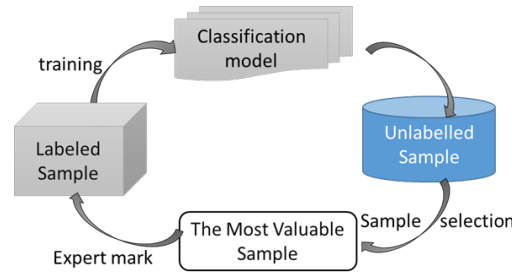


Fig. 6. Multi-granularity decision model with sequential three-way decision

The sample selection strategy determines the amount of information in the training set. The most straightforward method is random sampling, but more is needed to improve the generalization performance of the classifier. In contrast, generating high-entropy samples is a strategy for selecting the most valuable samples [23]. Based on the information entropy definition of the probability distribution, chosen examples by interval sampling are:

$$x^* = \operatorname{argmax}_{x \in U} (-\sum_i^n P(y = j_i | x) \log(P(y = j_i | x))), \quad (9)$$

where j_i denotes the most likely category i .

Fig. 7 shows the uncertainty entropy sampling algorithm. The Input was unlabeled samples U , sample queries: n , similarity threshold: s , and Output: L (set of samples to be labeled).

```

1 Initialization:  $t=0$ ;  $L=\emptyset$ 
2 While  $t < n$  do
3   Maximum similarity  $m=0$ 
4   Select sample:  $x_i \in U$ 
5   for each  $x_j \in L$  do
6     Computational similarity:  $\text{Sim}(x_i, x_j)$ 
7      $m = \max(m, \text{Sim}(x_i, x_j))$ 
8   end for
9   if  $(m < s)$  then
10     $L = L \cup x_i$ 
11     $t = t + 1$ 
12  End if
13   $U \leftarrow U - x_i$ 
14 End while
15 Return  $L$ 

```

Fig. 7. Uncertainty entropy sampling algorithm

We use the idea of non-maximum suppression in our uncertainty sampling strategy [24], where the first step of the algorithm selects the most informative sample of the current model and then determines whether the similarity between this sample and any sample in the set to be labeled is higher than a set threshold. If it is higher than this threshold, it is suppressed. Otherwise, the sample is placed in the set to be labeled L .

3. Results and Discussion

3.1. Multi-granularity active learning model and parameters

To reduce overfitting and training time, we start training by selecting a portion of the samples from the sample. We use a multi-granularity active learning network (MGAL) to extract features for image

classification. In Fig. 8, feature extraction of image information is used as image description $Des(x)$, and then the probability result $f(x)$ of classification is obtained using the softmax function. After training with partially labeled samples, the unlabeled samples are evaluated and decisions are made by Algorithm 1, and then Algorithm 2 is used to select the most useful samples.

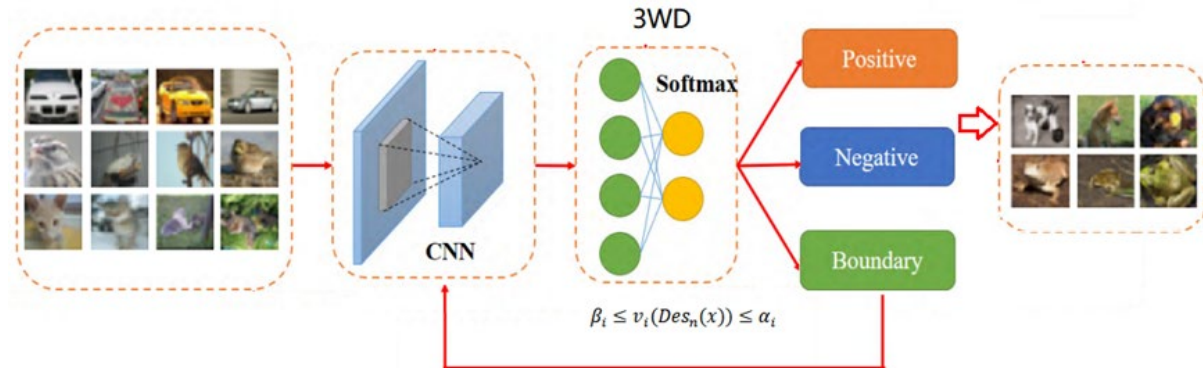


Fig. 8. Multi-granularity active learning model based on the three-way decision

The model makes a three-way decision on the classified sample set and calculates the classification cost for the three domains. The result with the lowest price is the decision result of the sample. So we add labeled samples to the previous step of the model. In each step model, these samples bring more information and make the model more accurately describe the sample set. The hyperparameters of the cost loss function in three-way decisions are shown in Table 1.

Table 1. Three-way decision cost

Decision	$L(x)=SP$	$L(x)=SN$
copy	$\lambda_{PP} = 0$	$\lambda_{PN} = 5$
Refuse	$\lambda_{NP} = 4$	$\lambda_{NN} = 0$
Delay decision	$\lambda_{BP}^i = 1 + \frac{i}{n}$	$\lambda_{BN} = 2 + \frac{i}{n}$

Confusion Matrix was used to analyse the classification results of model prediction, where TP represented the positive samples predicted into positive ones. TN is a positive class expected to be the negative class; FP is the negative samples indicated as positive samples; FN is the negative samples shown as negative ones and $TP+FP+TN+FN=$ Total number. The evaluation metrics are defined as follows.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (9)$$

The following experiments compare the accuracy of the experiments when 10%, 20%, 30%, 40%, 60% and 80% of the unlabeled sample set are added.

The algorithm divides the data set into the training set(10%), the unlabeled sample set(60%) and the test set (30%). The threshold value of redundant information reduction was set at 99.9%. The CNN network calculated the output value of the posterior probability of the unlabeled sample work—the number of expected labels per iteration to 10% of the initial sample set.

3.2. Experimental result on the UCI dataset

The five publicly available datasets from the UCI dataset [25], both binary multi-classified fields, were selected for part 1 of the experiment, as shown in Table 2. Four other active learning methods were selected for comparison experiments to validate the three-way decision-based active learning method (MGAL) proposed in this paper.

- Random selection [26] (Random).
- Representation Samples [27] (RA).

- Group-based approach [28] (GA)
- Cost-effective Active Learning [29] (CEA)

Table 2. Examples of UCI public datasets

No.	Datasets	Features	Samples	Categories
1	Pima	8	768	2
2	Sonar	59	208	2
3	CMC	9	1473	3
4	Vehicle	18	846	4
5	Yeast	7	1484	10

As can be seen from Table 3, Our proposed MGAL model performs relatively well on most of the sample sets and outperforms other active learning algorithms many times with the same proportion of unlabeled samples added. Fig. 9 shows the average results of the performance metrics on different datasets, and the MGAL model achieves significant improvement for the same increment.

Table 3. Accuracy of active learning models with different increments of unmarked samples

Datasets	Algorithm	Increment for unmarked samples(%)			
	10%	20%	40%	60%	80%
Pima	Random	0.7705	0.7769	0.789	0.7965
	RA	0.7774	0.7876	0.7831	0.7903
	GA	0.7745	0.7796	0.7914	0.7943
	CEA	0.7861	0.7913	0.8033	0.8062
	MGAL	0.8078	0.8091	0.8193	0.8357
Sonar	Random	0.7415	0.7745	0.8321	0.8512
	RA	0.7771	0.8109	0.828	0.8282
	GA	0.7323	0.7732	0.8207	0.8697
	CEA	0.7892	0.8126	0.8626	0.9141
	MGAL	0.8022	0.8237	0.8716	0.9172
CMC	Random	0.6509	0.6737	0.6862	0.6905
	RA	0.6658	0.6795	0.6997	0.6822
	GA	0.6579	0.6567	0.6688	0.671
	CEA	0.6872	0.6867	0.7148	0.7251
	MGAL	0.6956	0.7077	0.7297	0.7334
Vehicle	Random	0.7874	0.7947	0.8226	0.8362
	RA	0.7713	0.8001	0.8128	0.8325
	GA	0.7778	0.8093	0.812	0.8343
	CEA	0.7932	0.8193	0.8256	0.8446
	MGAL	0.8026	0.8231	0.8313	0.8522
Yeast	Random	0.7105	0.7251	0.7654	0.7806
	RA	0.7348	0.7577	0.7739	0.7841
	GA	0.7159	0.7255	0.7498	0.7889
	CEA	0.7481	0.7581	0.7835	0.8244
	MGAL	0.8073	0.8259	0.8379	0.8398

The experimental results show that the evaluation results of the classification algorithm gradually become better and close to a fixed value as unlabeled samples are added. The random active learning method, which selects unlabeled selections randomly, may cause inconsistent results. The Representative active learning method, which can choose the most representative samples, also suffers from the situation that the density of samples is high, but the value is not high; the CEA algorithm takes into account the cost of different labels and is second only to our algorithm in terms of effectiveness.

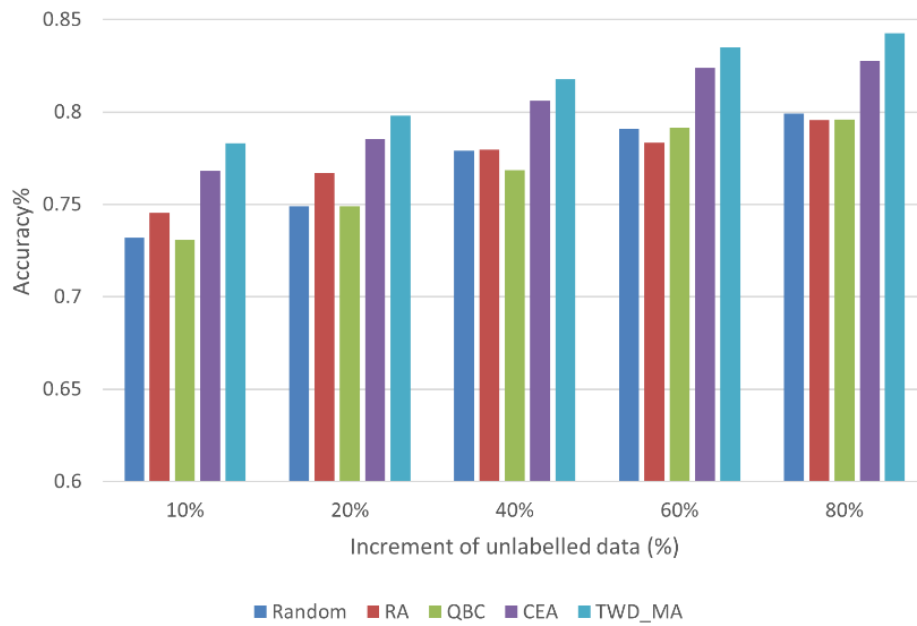


Fig. 9. Average classification accuracy on different data sets

3.3. Experiments on the dataset CIFAR10

Part 2 of the experiments compared the classification accuracy and decision cost of the three-way decision and with the classical two-way decision on the CIFAR10 dataset, which consists of 10 classes and 60,000 images, of which 80% were used for the training set, and 20% were used as the test set [30]. And a small portion of data containing labels was randomly selected as the initial training set. The similarity parameter $S1 = 0.70$ charged location for the data. The decision cost loss function parameters are shown in Table 1, and the classification accuracy and decision cost of the two-way and three-way decisions under different selection strategies were compared, as shown in Fig. 10, Fig. 11, and Fig. 12.

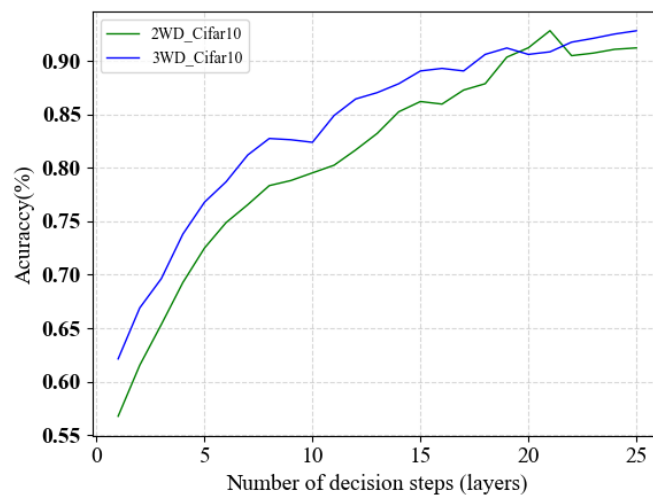


Fig. 10. Accuracy of three-way and two-way decisions

It is clear from the graph that the overall trend of the decision cost is the same for the three-way and two-way decisions. As the number of labeled samples increases and more time is spent on training, the network extracts more features, resulting in a more accurate description of the decision object and improving classification performance. For the same active learning strategy, the classification accuracy of the three-way decision is higher than that of the two-branch decision. At the same time, the cost is lower, which is the advantage of making delayed decisions with insufficient information.

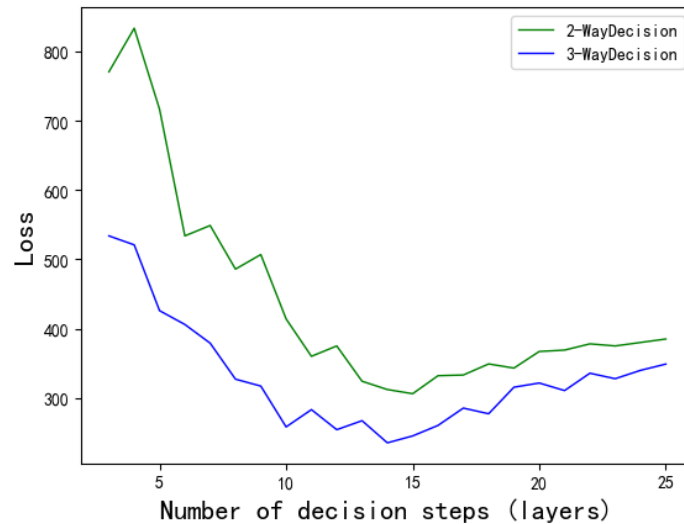


Fig. 11. Loss cost of three-way and two-way decisions

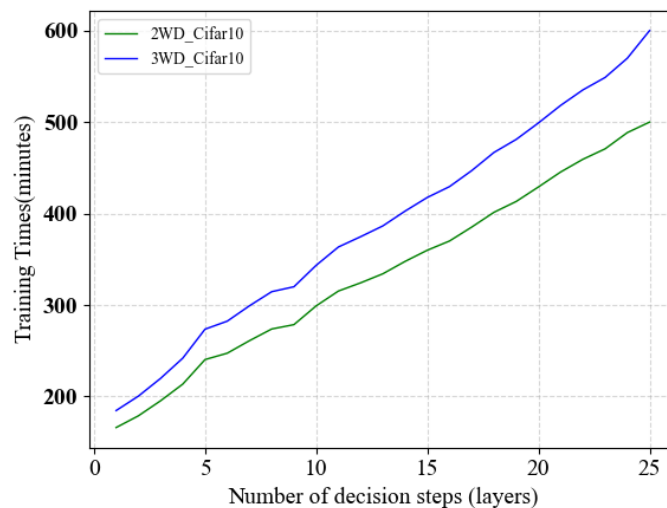


Fig. 12. Training time for three and two-way decisions

4. Conclusion

This paper proposes a new multi-granularity active learning classification model based on three-way decision theory. It can be used to label unlabeled samples and classify multi-granularity images. The three-way decision algorithm is used to divide the uncertain sample space. The suspicious samples are decomposed into three subdomains for three-way decision-making to select a representative and low-cost example and improve the decision efficiency. Compared with the two-way decision, the three-way decision can better handle the uncertainty and multi-granularity features in the samples. Experimental results on the UCI and CIFAR10 datasets show that the method significantly improves classification performance compared to other active learning methods. However, Our active learning algorithms can also increase computational complexity when dealing with high feature dimensions or large sample sizes. Future work will introduce downscaling and attribute reduction methods for complex datasets to further improve the efficiency of this model.

Acknowledgement

The author(s) thank Science Foundation of Minzu Normal University of Xingyi, China for supporting this research work.

Declarations

Author contribution. The 1st author's contribution to the paper is comprehensive and primary, with the 2nd author responsible for proofreading and advice.

Funding statement. This work was supported by the Science Foundation of Minzu Normal University of Xingyi (No. 20XYJS01).

Conflict of interest. The authors declare no conflict of interest.

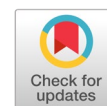
Additional information. All information is private for this paper.

References

- [1] S. Liu, Y. Wang, Q. Yu, H. Liu, and Z. Peng, "CEAM-YOLOv7: Improved YOLOv7 Based on Channel Expansion and Attention Mechanism for Driver Distraction Behavior Detection," *IEEE Access*, vol. 10, pp. 129116–129124, 2022, doi: [10.1109/ACCESS.2022.3228331](https://doi.org/10.1109/ACCESS.2022.3228331).
- [2] W. Wang *et al.*, "InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions," *arXiv*, pp.1-19, Nov. 2022, doi : [10.48550/arXiv.2211.05778](https://doi.org/10.48550/arXiv.2211.05778).
- [3] D. Su and P. Fung, "Improving Spoken Question Answering Using Contextualized Word Representation," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2020-May, pp. 8004–8008, May 2020, doi: [10.1109/ICASSP40776.2020.9053979](https://doi.org/10.1109/ICASSP40776.2020.9053979).
- [4] K. H. Choi and J. E. Ha, "Random Swin Transformer," *Int. Conf. Control. Autom. Syst.*, vol. 2022-November, pp. 1611–1614, 2022, doi: [10.23919/ICCAS55662.2022.10003789](https://doi.org/10.23919/ICCAS55662.2022.10003789).
- [5] W. Wang *et al.*, "mmLayout: Multi-grained MultiModal Transformer for Document Understanding," *Association for Computing Machinery (ACM).*, vol. 2022-Oct, pp. 4877–4886, doi: [10.48550/arXiv.2209.08569](https://doi.org/10.48550/arXiv.2209.08569).
- [6] J. Li, M. Wang, and X. Gong, "Transformer Based Multi-Grained Features for Unsupervised Person Re-Identification," *Proc. - 2023 IEEE/CVF Winter Conf. Appl. Comput. Vis. Work. WACVW 2023*, pp. 42–50, 2023, doi: [10.1109/WACVW58289.2023.00009](https://doi.org/10.1109/WACVW58289.2023.00009).
- [7] W. Guoyin, Y. Hong, W. Guoyin, and Y. Hong, "Multi-Granularity Cognitive Computing—A New Model for Big Data Intelligent Computing," *Front. Data Domputing*, vol. 1, no. 2, pp. 75–85, Jan. 2020, doi: [10.11871/jfdc.issn.2096-742X.2019.02.007](https://doi.org/10.11871/jfdc.issn.2096-742X.2019.02.007).
- [8] J. Chen, Z. Du, X. Sun, S. Zhao, and Y. Zhang, "A multi-granular network representation learning method," *Granul. Comput.*, vol. 6, no. 1, pp. 59–68, Jan. 2021, doi: [10.1007/s41066-019-00194-2](https://doi.org/10.1007/s41066-019-00194-2).
- [9] J. Chen, P. Wang, J. Liu, and Y. Qian, "Label Relation Graphs Enhanced Hierarchical Residual Network for Hierarchical Multi-Granularity Classification," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2022-June, pp. 4848–4857, 2022, doi: [10.1109/CVPR52688.2022.00481](https://doi.org/10.1109/CVPR52688.2022.00481).
- [10] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1-14, Sep. 2015, doi: [10.48550/arXiv.1409.1556](https://doi.org/10.48550/arXiv.1409.1556).
- [11] S. H. Gao, M. M. Cheng, K. Zhao, X. Y. Zhang, M. H. Yang, and P. Torr, "Res2Net: A New Multi-Scale Backbone Architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 652–662, Feb. 2021, doi: [10.1109/TPAMI.2019.2938758](https://doi.org/10.1109/TPAMI.2019.2938758).
- [12] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 936–944, Nov. 2017, doi: [10.1109/CVPR.2017.106](https://doi.org/10.1109/CVPR.2017.106).
- [13] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, "Scaling Vision Transformers," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2022-June, pp. 12094–12103, 2022, doi: [10.1109/CVPR52688.2022.01179](https://doi.org/10.1109/CVPR52688.2022.01179).
- [14] D. Yoo and I. S. Kweon, "Learning loss for active learning," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 93–102, Jun. 2019, doi: [10.1109/CVPR.2019.00018](https://doi.org/10.1109/CVPR.2019.00018).

- [15] J. Choi, I. Elezi, H. J. Lee, C. Farabet, and J. M. Alvarez, "Active Learning for Deep Object Detection via Probabilistic Modeling," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 10244–10253, 2021, doi: [10.1109/ICCV48922.2021.01010](https://doi.org/10.1109/ICCV48922.2021.01010).
- [16] J. Shao, Q. Wang, and F. Liu, "Learning to sample: An active learning framework," *Proc. - IEEE Int. Conf. Data Mining, ICDM*, vol. 2019–November, pp. 538–547, Nov. 2019, doi: [10.1109/ICDM.2019.00064](https://doi.org/10.1109/ICDM.2019.00064).
- [17] Y. Yao, "Three-way decision and granular computing," *Int. J. Approximate Reasoning.*, vol. 103, pp. 107–123, Dec. 2018, doi: [10.1016/J.IJAR.2018.09.005](https://doi.org/10.1016/J.IJAR.2018.09.005).
- [18] A. Campagner, F. Cabitza, and D. Ciucci, "Three-Way Decision for Handling Uncertainty in Machine Learning: A Narrative Review," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12179 LNAI, pp. 137–152, 2020, doi: [10.1007/978-3-030-52705-1_10](https://doi.org/10.1007/978-3-030-52705-1_10).
- [19] Y. Yao, "Tri-level thinking: models of three-way decision," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 5, pp. 947–959, May 2020, doi: [10.1007/S13042-019-01040-2](https://doi.org/10.1007/S13042-019-01040-2).
- [20] J. Qian, D. W. Tang, Y. Yu, X. B. Yang, and S. Gao, "Hierarchical sequential three-way decision model," *Int. J. Approx. Reason.*, vol. 140, pp. 156–172, Jan. 2022, doi: [10.1016/j.ijar.2021.10.004](https://doi.org/10.1016/j.ijar.2021.10.004).
- [21] W. Qian, Y. Zhou, J. Qian, and Y. Wang, "Cost-sensitive sequential three-way decision for information system with fuzzy decision," *Int. J. Approx. Reason.*, vol. 149, pp. 85–103, Oct. 2022, doi: [10.1016/j.ijar.2022.07.006](https://doi.org/10.1016/j.ijar.2022.07.006).
- [22] P. Liu, L. Wang, R. Ranjan, G. He, and L. Zhao, "A Survey on Active Deep Learning: From Model Driven to Data Driven," *Association for Computing Machinery (ACM).*, vol. 54, no. 10, pp. 1–34, Sep. 2022, doi: [10.1145/3510414](https://doi.org/10.1145/3510414).
- [23] C. Mayer and R. Timofte, "Adversarial sampling for active learning," *Proc. - 2020 IEEE Winter Conf. Appl. Comput. Vision, WACV 2020*, pp. 3060–3068, Mar. 2020, doi: [10.1109/WACV45572.2020.9093556](https://doi.org/10.1109/WACV45572.2020.9093556).
- [24] Y. Yang and M. Loog, "A benchmark and comparison of active learning for logistic regression," *Pattern Recognit.*, vol. 83, pp. 401–415, Nov. 2018, doi: [10.1016/j.patcog.2018.06.004](https://doi.org/10.1016/j.patcog.2018.06.004).
- [25] Dua, D. and Graff, C. "UCI Machine Learning Repository". Irvine, CA: University of California, School of Information and Computer Science, 2019, Available at : archive.ics.uci.edu/ml.
- [26] P. Ren *et al.*, "A Survey of Deep Active Learning," *Association for Computing Machinery (ACM).*, vol. 54, no. 9, pp. 1–40, Oct. 2021, doi: [10.1145/3472291](https://doi.org/10.1145/3472291).
- [27] X. Yan *et al.*, "A clustering-based active learning method to query informative and representative samples," *Appl. Intell.*, vol. 52, no. 11, pp. 13250–13267, Sep. 2022, doi: [10.1007/S10489-021-03139-Y](https://doi.org/10.1007/S10489-021-03139-Y).
- [28] B. Settles and M. Craven, "An Analysis of Active Learning Strategies for Sequence Labeling Tasks," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL Press, 2008, pp. 1070–1079, doi: [10.5555/1613715.1613855](https://doi.org/10.5555/1613715.1613855).
- [29] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-Effective Active Learning for Deep Image Classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 12, pp. 2591–2600, Dec. 2017, doi: [10.1109/TCSVT.2016.2589879](https://doi.org/10.1109/TCSVT.2016.2589879).
- [30] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," *Computer Science University Of Toronto*, pp. 1–60, 2009. Available at : <http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf>.

Predicting extreme events in the stock market using generative adversarial networks



Badre Labiad ^{a,1,*}, Abdelaziz Berrado ^{a,2}, Loubna Benabbou ^{b,3}

^a Equipe AMIPS, Ecole Mohammadia d'Ingénieurs, Mohammed V University in Rabat, Morocco

^b Department of Management Science, Université du Québec à Rimouski (UQAR), Campus de Lévis Québec Canada

¹ labiad.badre@gmail.com ; ² berrado@emi.ac.ma; ³ loubna_benabbou@uqar.ca

* corresponding author

ARTICLE INFO

Article history

Received August 26, 2022

Revised April 10, 2023

Accepted April 24, 2023

Available online May 30, 2023

Keywords

Extreme events prediction

Time series generation

Stock markets simulation

Generative adversarial networks

Long short-term memory

ABSTRACT

Accurately predicting extreme stock market fluctuations at the right time will allow traders and investors to make better-informed investment decisions and practice more efficient financial risk management. However, extreme stock market events are particularly hard to model because of their scarce and erratic nature. Moreover, strong trading strategies, market stress tests, and portfolio optimization largely rely on sound data. While the application of generative adversarial networks (GANs) for stock forecasting has been an active area of research, there is still a gap in the literature on using GANs for extreme market movement prediction and simulation. In this study, we proposed a framework based on GANs to efficiently model stock prices' extreme movements. By creating synthetic real-looking data, the framework simulated multiple possible market-evolution scenarios, which can be used to improve the forecasting quality of future market variations. The fidelity and predictive power of the generated data were tested by quantitative and qualitative metrics. Our experimental results on S&P 500 and five emerging market stock data show that the proposed framework is capable of producing a realistic time series by recovering important properties from real data. The results presented in this work suggest that the underlying dynamics of extreme stock market variations can be captured efficiently by some state-of-the-art GAN architectures. This conclusion has great practical implications for investors, traders, and corporations willing to anticipate the future trends of their financial assets. The proposed framework can be used as a simulation tool to mimic stock market behaviors.



This is an open access article under the [CC-BY-SA](#) license.



1. Introduction

Stock markets are dynamic and complex environments whose behavior is largely influenced by exogenous factors (such as political and geopolitical, economic, and natural). The influence of these factors can be so extreme that it can trigger certain crises such as bubbles, panics, booms, meltdowns, or crashes [1], [2]. For example, stock markets across the world reacted with extreme variations to the spread of the COVID-19 pandemic [3]–[6]. In such contexts, the challenge traders and investors face is proactively anticipating and integrating the impacts of such events into their trading strategies Fig. 1 shows the drop in the S&P 500 and five major emerging stock market indices during the early days of the COVID-19 pandemic (1st semester 2020).

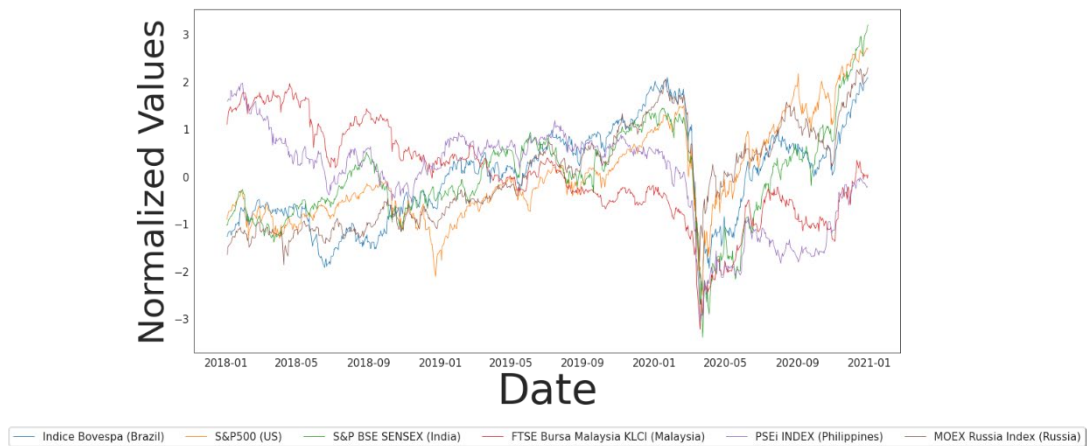


Fig. 1. Crash of S&P 500 and five major emerging stock market indices during the first days of the COVID-19 pandemic (1st semester 2020)

Efficiently simulating stock prices' extreme fluctuations would allow decision-makers and investors to make better-informed investment decisions and practice more efficient financial risk management. The classical procedure of evaluating trading strategies involves testing them on real, observed market data through a process called backtesting [7]. However, a major shortcoming of this procedure is its reliance on historical data, which offers only one view of market behavior. Efficient trading strategy calibration would require multiple market scenarios and a wide range of realistic price variations.

In this study, we designed a procedure to simulate stock-market conditions, even the extreme ones, by generating realistic synthetic market data and reproducing a wide range of price variations. This framework can be used as a practical tool to test and improve forecasting models, trading strategy calibration, portfolio management, market manipulation detection, and risk management.

The difficulties encountered while dealing with stock market data are multiple and varied (such as the nature of the factors influencing the variations; the non-linear, non-parametric, nonstationary, and arbitrary nature of the data; and extreme movements) [8]. Moreover, access to granular microstructure market data is limited and expensive. Access to a richer data set is of great importance for the research community. Therefore, the procedure proposed in this study also helps overcome data-access limitations for researchers, traders, and investors by creating synthetic datasets with the same statistical properties as the original data.

Understanding and modeling the temporal dynamics of stock markets to simulate their behaviors and predict their future movements, especially during periods of crisis and high volatility, is an objective that has caught the interest of researchers, traders, and investors for several years [9]–[11]. Deep learning [12] is one of the most promising techniques tested for stock-market prediction. This technique is inspired by the brain structure consisting of multi-layer neural networks [13] and overcomes the limitations of traditional machine-learning methods that cannot efficiently capture the complexity of financial data [14]. Motivated by the successes of deep recurrent learning models [12], especially in their ability to learn long-term dependency information, several studies have attempted to apply them to financial data, particularly to stock markets [15]–[18]. Deploying effective deep-learning models requires large and balanced datasets to get convincing results because small and unbalanced training datasets lead to overfitting and poor generalization [14]. Extreme variations in the stock market are rare events [15]. This is an additional difficulty from the modeling perspective. Furthermore, stock price data with extreme variations are unbalanced time-series datasets. To mitigate this challenge, algorithm solutions and data-oriented methods are used. The former consists of adapting algorithms to better handle minority class errors (extreme instances). The second, a more flexible approach, consists of data augmentation and oversampling (undersampling) of the minority (majority) class [19]. Generally, this technique is used to fix imbalanced data sets and missing data or create new synthetic data points when access to real data is limited [20], [21]. The synthetic data must be as realistic and flexible as the real one. Generative adversarial networks (GANs) are among the data augmentation techniques tested for

stock markets [22]–[25]. However, the majority of the above-mentioned studies assessed the quality of their results under normal market conditions. Extreme variations in the stock markets are extraordinary and rare circumstances. Generating data similar to extreme market situations requires additional conditioning and tuning of the generating process to produce realistic synthetic data.

Although GAN application for stock-market prediction is an active area of research, to the best of our knowledge, there is still a gap in the literature on the use of GANs for market simulation and extreme-event prediction. Considering the above observations, we designed a framework based on the GAN procedure using three state-of-the-art algorithms, which offer interesting features regarding the possibility of conditioning the extremeness of generated data, to simulate market behavior and produce new synthetic examples that are used to efficiently predict financial stock prices during extreme variations. We tested the framework on stock data from the S&P 500 and five emerging markets. In this study, GANs were employed as a data augmentation technique to create new realistic synthetic examples to tackle the scarcity of extreme observations in the training dataset and to simulate multiple scenarios of market evolution. After enriching the training dataset, a long short-term memory model (LSTM) was used for prediction tasks.

Our study makes the following key contributions:

- We propose a framework capable of producing synthetic data that simulates market behaviors: this work contributes to the new area of research in GAN usage for stock market simulation. This application of GANs yields the potential of overcoming issues related to limited data access.
- We illustrate that synthetic stock market data can be used to enhance training datasets and improve the forecasting of extreme events: by proposing a clear methodology to assess the quality of the generated data, we show that the synthetic real-looking examples can efficiently be used to perform some practical tasks such as the forecasting of stock prices. We trained an LSTM model on both real and synthetic data to forecast stock price variations and evaluated the improvement in the forecasting error metrics after using the new synthetic examples.
- We present a comparison between three different GAN architectures to perform extreme events prediction—which, according to our knowledge, is still an understudied application of GANs: the usage of GANs for extreme event modeling is a promising area of research, and this work illustrates the practical steps to undertake to get robust results.

The rest of the paper is organized as follows: The proposed framework is detailed in section 2. The experiment results are discussed in section 3. Finally, conclusions and possible extensions are discussed in the last section.

2. Method

2.1. Problem formulation

The underlying process of stock-price evolution is hard to model. Trying to capture the stock-market dynamics by explicitly developing handcrafted assumptions and rules is intractable and inefficient. However, the GAN framework offers the possibility of reproducing features of the underlying process of the stock market movements with high fidelity. We assumed that the generated data will enrich the training examples, particularly more examples from the tail of the underlying distribution, as our focus was on the prediction of extreme events.

Let $X_{1:T} = (X_1, X_2, \dots, X_T)$ be a sample of our dataset, where X_i represents the features of the i^{th} time-step of the sample. T represents the time window size. The objective was to allow the generator framework to reproduce high-quality data by learning a density $\hat{p}(X_{1:T})$ that approximated the best original density $p(X_{1:T})$, where $\hat{p}(X_{1:T})$ is the learned density that minimizes the following distance:

$$\text{Min } \hat{p} \text{ Dist}(\hat{p}(X_{1:T}) || p(X_{1:T})) \quad (1)$$

The metric (Dist) is the Jensen–Shannon Divergence [26].

In addition, the generative process described above was needed to efficiently capture the temporal transitions. More formally, we needed to accurately capture the conditional distribution $p(X_{T+1}, X_{T+2}, \dots, X_{T+k} | X_{1:T})$ that characterized temporal dependencies. The goal was to accurately predict $X_{T+1}, X_{T+2}, \dots, X_{T+k}$ by training a model using real and synthetic data. The synthetic data itself was generated using the sample $X_{1:T} = (X_1, X_2, \dots, X_T)$.

2.2. The proposed framework

In this section, we present the proposed framework for extreme events forecasting. As mentioned before, synthetic data were used in our study to overcome the rarity of extreme events in the stock market. Hence, by generating new synthetic data, we improved the richness of the initial dataset and gave the learning model new data to train with. To assess the fidelity and predictive power of the generated data, we used it to train a recurrent algorithm—namely, LSTM—and compared the quality of the predictions on the synthetic examples against the predictions on the real ones. Furthermore, quantitative tests were conducted to evaluate the fidelity of the synthetic data and its predictive power. The framework was composed of three modules: (1) the data generation module, (2) the forecasting module, and (3) the testing module Fig. 2 provides an overview of the proposed framework. More details about the components of the framework are presented in the following subsections.

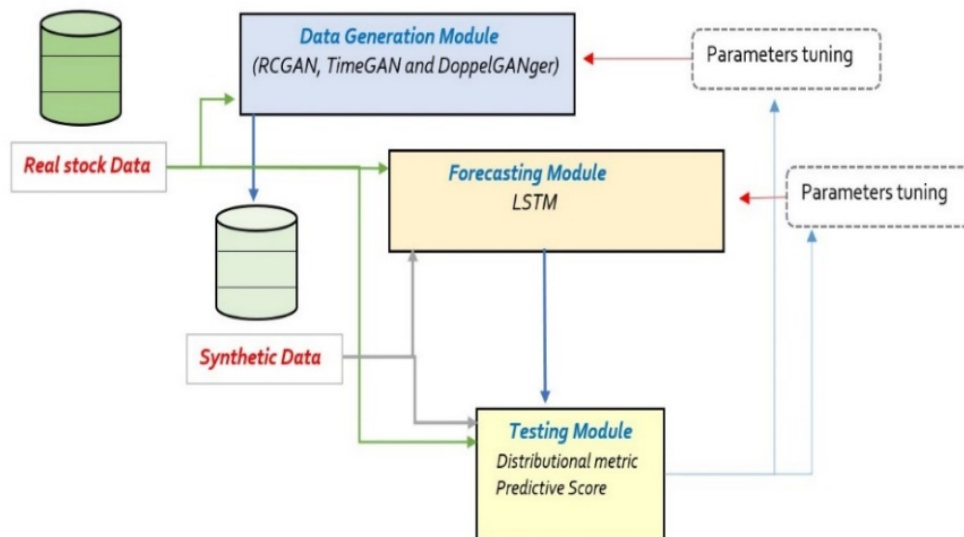


Fig. 2. Proposed framework

2.2.1 Data generation module

GANs [26] are a class of unsupervised learning algorithms composed of two deep adversarial systems: a generator G that tries to map a random noise z , drawn from a Gaussian, to a sample of real data x and a discriminator D that tries to distinguish between the real data and the generated one. The two systems are simultaneously trained in a two-player zero-sum game with a value function $V(G, D)$:

$$\min_G \max_D V(G, D) = E_{x \sim P_{data}(x)} [\log D(x)] + E_{z \sim P_z(z)} [\log(1 - D(G(z)))] \quad (2)$$

Here, $D(x)$ is the probability attributed by the discriminator that the real examples x are real. $D(G(z))$ is the probability attributed by the discriminator that a fake example is real. E is the expected value.

It is worth mentioning that training a GAN is a tricky task. A GAN can easily fail to converge because of many reasons such as vanishing gradients or mode collapse problems. To overcome GAN training issues, recent works have proposed modifications to the objective function to ensure smooth training [27].

We used GANs as data generation techniques to create synthetic training instances. We tested and compared three state-of-the-art generative algorithms: DoppelGANger (DG) <https://github.com/fjxmlzn/DoppelGANger>, TimeGAN <https://github.com/jsyoons0823/TimeGAN>, and Recurrent conditional GAN (RCGAN) <https://github.com/ratschlab/RCGAN> [28]. RCGAN [28] is a GAN variant that produces time series data. In RCGAN, both the generator (Fig. 3) and the discriminator are LSTM conditioned with auxiliary information.

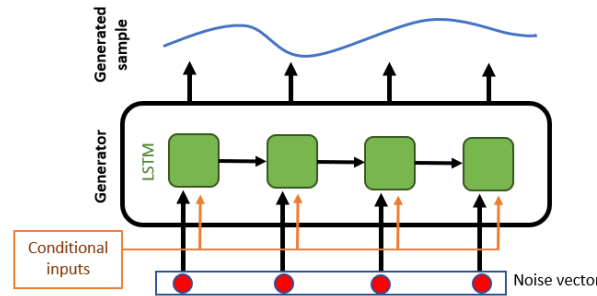


Fig. 3. Architecture of the RCGAN's generator

The conditional vector gave more context to the generator to produce data with properties similar to the real ones. The setting of RCGAN hyperparameters is shown in Table 1.

Table 1. Key hyperparameters of the RCGAN model

Hyperparameters	Values
RNN	LSTM
The number of layers of the generator/discriminator	3/3
The number of units in each layer of the generator/discriminator	20/20
Conditional inputs size/values	3 / $X_{t,High}$, $X_{t,Low}$, $X_{t,Open}$

TimeGAN [29] is basically made of four functions: recovery function, embedding function, sequence discriminator, and sequence generator. The training steps were performed in a way to make TimeGAN simultaneously learn the temporal dependencies, encode features, and generate representations Fig. 4 shows the main components of the TimeGAN architecture. The Key hyperparameters of the TimeGAN model could be seen in Table 2.

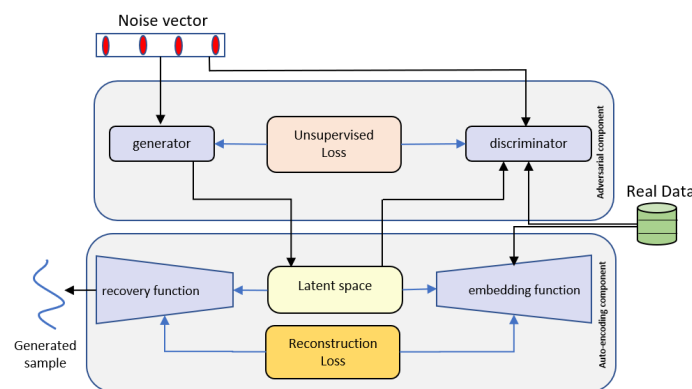


Fig. 4. Architecture of TimeGAN

Table 2. Key hyperparameters of the TimeGAN model

Hyperparameters	Values
Embedding network, generator, and discriminator	LSTM
The number of layers of the generator/discriminator/embedding	3/3/4
The number of units in each layer of the generator/discriminator/embedding	20/20/30
Latent space dimension	2

DG (Fig. 5) used recurrent neural networks (LSTM) as generators to capture long-term dependency [30]. It also used some innovative ideas to tackle the above-mentioned shortcoming of classical GAN architecture: (1) Bach-generation: To make the LSTM more efficient in generating sequential samples, instead of generating one record, DG produced S records at each pass. (2) Auto-normalization: To prevent mode collapse, the minimum/maximum of each time series was learned and generated by an independent generator and used as conditional input. (3) Attributes generation: Correlations between the time series and their attributes were captured by another generator that learned and generated the time series attributes used as conditioning inputs. This procedure differed from classical GAN approaches, where attributes and features are generated in the same step.

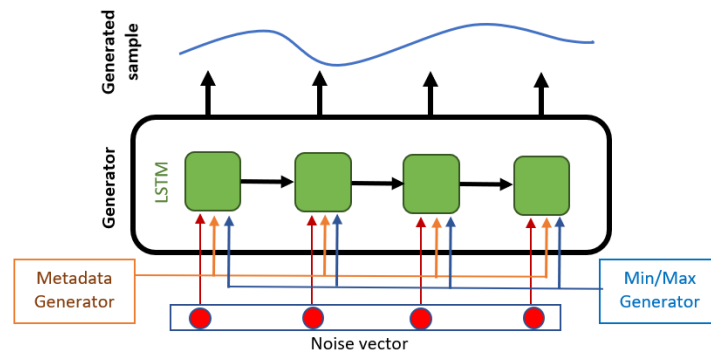


Fig. 5. Architecture of the DG's generator

To improve the fidelity of the generated distribution, DG also used an auxiliary discriminator that discriminated only on metadata. DG model hyperparameters setting are shown in Table 3.

Table 3. Key hyperparameters of the DG model

Hyperparameters	Values
S records generated by the bach-generation	4
Minimum/maximum generator: Multilayer perceptron (MLP) (input units:hidden units:output units)	2:4:2
input units: minimum/maximum	
Metadata generator: MLP (input units:hidden units:output units)	3:6:4
input units: $X_{t,High}$, $X_{t,Low}$, $X_{t,Open}$,	
Number of layers in the time series generator (LSTM)	3
Number of units in each layer of the time series generator (LSTM)	20

The noise feed to the generators was a vector of random values drawn from a Gaussian distribution. The dimension of this vector was a hyperparameter set to 10 in our experiments.

2.2.2 Forecasting module

For prediction purposes, the forecasting module used an LSTM. The LSTM cell (Fig. 6) had three gates: an input gate, a forget gate, and an output gate to add or remove information.

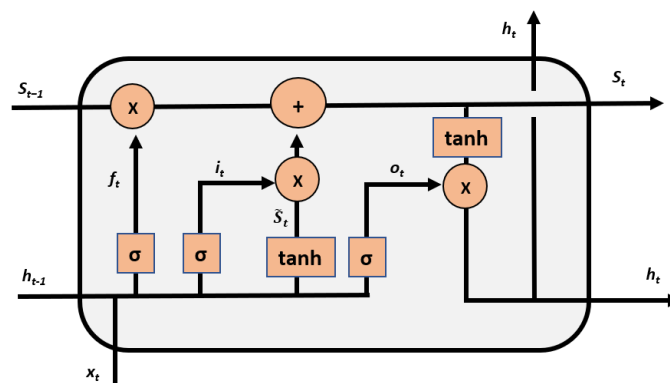


Fig. 6. Architecture of the LSTM's Cell

The cell state, indicated by « S_t », represented the internal final memory that stored short-term and long-term dependencies, and \tilde{S}_t was the new intermediary state of the memory cell. To minimize the prediction error, the weights between the hidden layers were adjusted during the learning step of the LSTM. The outputs h_t were new features that captured the essential information from input features. Here, i_t , o_t , and f_t are the outputs of the cell gates. The LSTM Model hyperparameters are shown in Table 4. All the experimental computations were coded using Python within the TensorFlow framework.

Table 4. Key hyperparameters of the LSTM model

Hyperparameters	Values
Number of layers	3
Number of hidden units per layer	50/100/150
Number of training epochs	300
Training algorithm	Adam optimizer
Learning rate	0.001
Dropout layers	2
Dropout probability	50%
Mini-batch size	50

2.2.3 Testing module

To assess the quality of the generated data and the forecasting results, the testing module used two main criteria: (1) distributional diversity generated samples should be distributed like the real data (even the extreme events) and (2) predictive power generated data should be as informative as the real one and useful in training models for predictive tasks. To assess these criteria, we considered the following metrics:

- Distributional metric [31]

To assess criteria (1), we compared the distributions of synthetic and real data. The two distributions had to be as close as possible. We assumed a binning $S_h = (S_1, \dots, S_k)$ such that about n records of the observed data $X_{1:T}$ were in each bin. The empirical probability density function of the observed series $\hat{f}_h : S_h \rightarrow \mathbb{R} \geq 0$ and the synthetic $\hat{f}_g : S_h \rightarrow \mathbb{R} \geq 0$ could then be defined. The absolute difference between these functions was

$$\Sigma B \in B_h | f_h(S) - f_g(S) | \quad (3)$$

This quantity measured how the two distributions were close to each other. The smaller this quantity, the closer the distributions were.

- Predictive Score [29]

By using additional synthetic samples, we enhanced the predictive power of the real data. So, we first compared the accuracy of the prediction of the same model (LSTM used in the forecasting module) on the real data, then on the synthetic data. To do so, two settings were tested: train on real and test on real (TRTR) and train on synthetic and test on real (TSTR). The TSTR procedure consisted of the addition of the generated synthetic examples to the training dataset already containing real examples.

This procedure enriched the training datasets with more examples. The tests were always performed on real data [28]. Since we were comparing continuous real-valued times series, we needed to use metrics that captured the values of differences between real (y_i) and generated data (\hat{y}_i). Hence, errors were measured by using mean absolute error (MAE), mean absolute percentage error (MAPE), and mean squared error (MSE). MAE was used to measure, on average, how close the predictions were to the outcomes. MAPE was the average percentage of errors. Finally, MSE measured the average squares of the error. It gave more importance to the big errors.

$$MAE = \frac{1}{n} \sum_{i=0}^n | y_i - \hat{y}_i | \quad (4)$$

$$MAPE = \frac{100}{n} \sum_{i=0}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{5}$$

$$MSE = \frac{1}{n} \sum_{i=0}^n (y_i - \hat{y}_i)^2 \tag{6}$$

2.3. Experimental Data

Generally, stock market analysts use a wide range of technical indicators as handcrafted features to predict future evolutions of the stock price. For this framework, we chose the following attributes as input features for the data generation and forecasting modules: $X_t = (X_{t,High}, X_{t,Low}, X_{t,Open}, Y_{t,Close})$, referring to the highest, lowest, opening, and closing prices, respectively, recorded on trading day t . Table 5 shows examples of these data. The forecasting module aimed to accurately predict $\hat{Y}_{T+1}, \hat{Y}_{T+2}, \dots, \hat{Y}_{T+k}$, by training an LSTM using real and generated (synthetic) data, where k is the forecasting horizon. In our study, for the sake of simplification, this parameter was set to 1.

Table 5. Examples of the studied data (Brazil stock index)

Date	Open	High	Low	Close
02-01-2018	76403	77909	76403	77891
03-01-2018	77889	78414	77602	77995
04-01-2018	77998	79135	77998	78647
05-01-2018	78644	79071	78218	79071

2.3.1. Data description

To demonstrate the utility of using synthetic data for stock market forecasting and, more specifically, for extreme variations, we applied our framework to real data from the S&P 500 and five emerging markets: Brazil, India, Malaysia, the Philippines, and Russia. The choice of these data allowed a comparison based on a wide range of market profiles.

Our dataset covered 4 years from 2018 to 2021, corresponding with more than 730 data points for each stock. The historical data were obtained from finance.yahoo.com.

Table 6 presents a summary of some descriptive statistics related to the studied datasets. The standard deviation (SD) measured how the data were spread around the mean Fig. 7 shows examples of the extreme behavior of the studied indices during the spread of the COVID-19 pandemic. The prices of these indices experienced a significant drop during the early days of the pandemic.

Table 6. Descriptive statistics of the studied indices (closing prices)

Statistics	Brazil	India	Malaysia	Philippines	Russia	S&P500
Count	741	735	732	728	740	755
Mean	93705	37213	1640	7333	2629	2958
SD	12397	3302	130	879	287	286
Min	63570	25981	1219	4623	2090	2237
Max	119528	47746	1895	9058	3289	3735

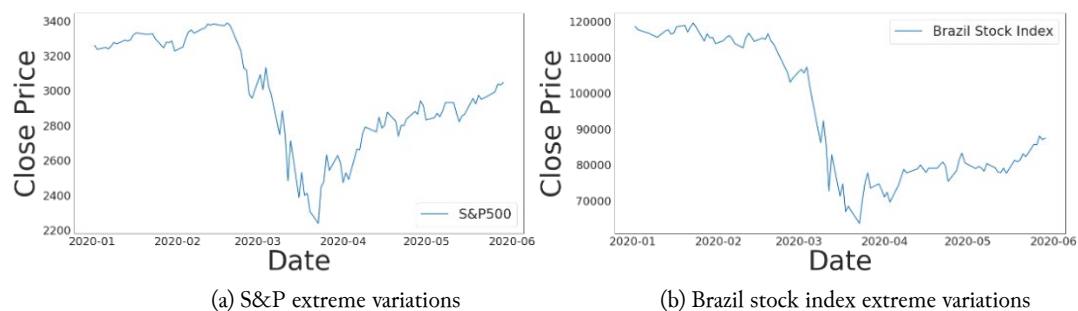


Fig. 7. Examples of indices with extreme variations

Fig. 8 illustrates how the returns are spread around the means, with extreme values being far from the center of the distributions.

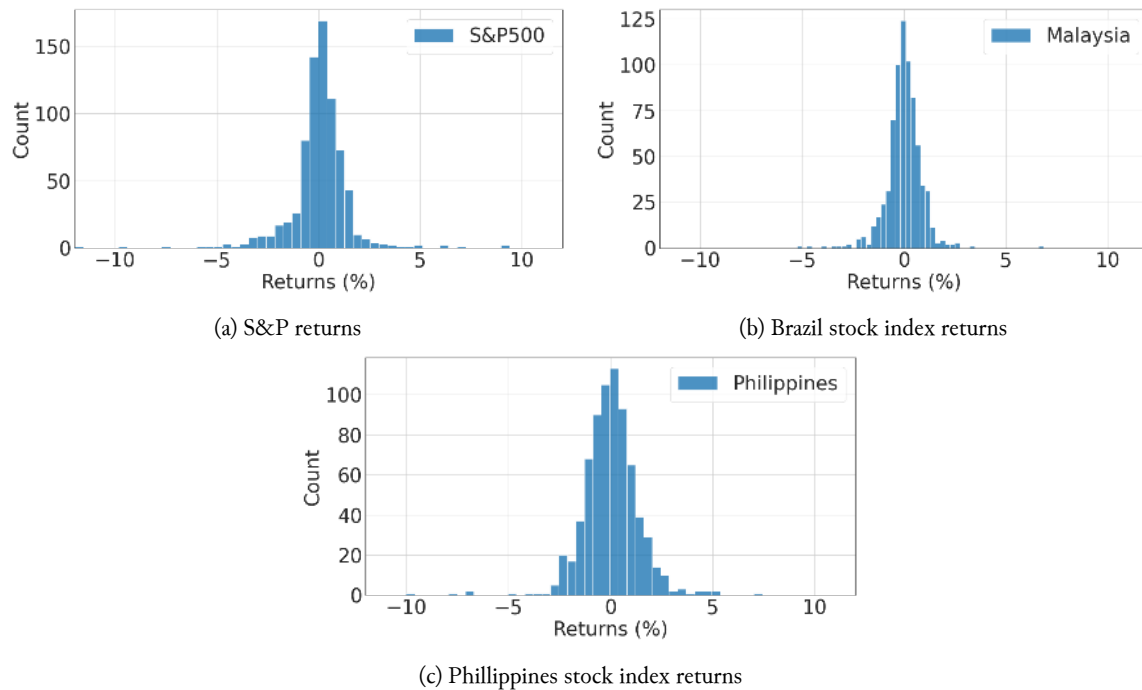


Fig. 8. Examples of returns as histogrammes of the studied indices

2.3.2. Data preparation

Dataset splitting: The dataset was split into two parts: two-thirds of the records were used for the training step and the remaining observations were used for testing (robustness check) and comparing the tested models.

Data standardization: The standardization step was crucial to speed up the learning process. In our study, we standardized the data in a way that their distribution would have a mean value of 0 and an SD of 1, as illustrated in equation (5):

$$x' = x - \mu/\sigma \quad (5)$$

Where μ and σ are the mean value and the SD, respectively, of the training sample prices.

3. Results and Discussion

3.1. Experimental Results

The output of the generation step was sets of synthetic data resembling the real ones with the same entries (features). The visual investigation of the histograms in Fig. 9 shows the high similarity between the distributions of the generated and real data. In addition, the synthetic data show real-looking extreme values between the $\pm 5\%$ and $\pm 10\%$ returns. Since there is a lack of space, each index Fig. 9 shows only the results for the best generation model from the three models tested.

The figure also shows that the DG algorithm produced the best real-looking data, followed by RCGAN. TimeGAN's output was similar to the original data in only the bulk of the distributions, but, in our dataset, it failed to replicate the characteristics of the tails of distributions.

These results can be explained by the conditioning attributes offered by the DG algorithm that better captured the underlying dynamics of training data.

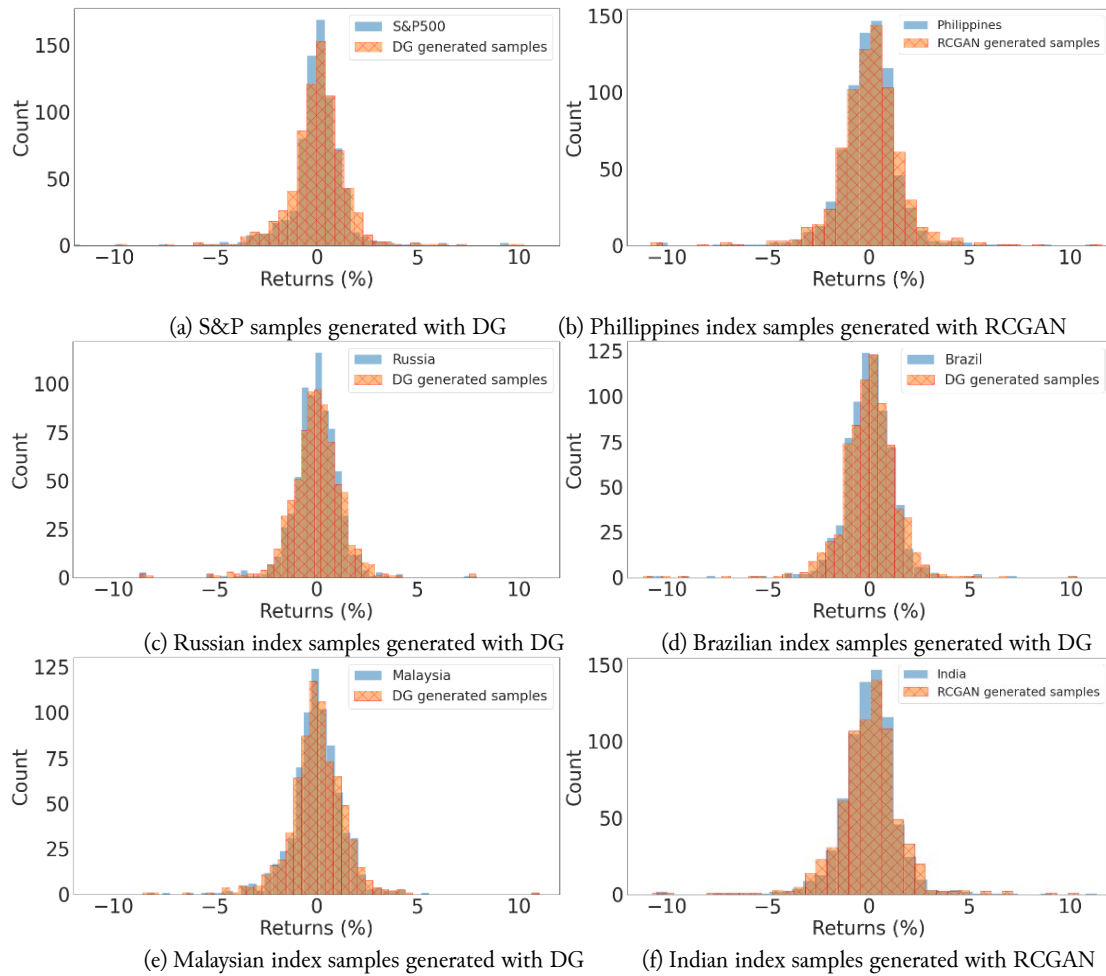


Fig. 9. Histograms of the generated samplesUnits

Quantitatively, the measure of similarity between the distributions, reported in Table 7, shows that DG and RCGAN give the best distributional scores. In other words, the data generated by RCGAN and DG look more like real data in terms of the shape of their distributions. Unlike DG and RCGAN, TimeGAN did not allow any conditioning for the generation step; this can explain the relatively low quality of the produced data.

Table 7. Distributional scores

	S&P 500	Brazil	India	Malaysia	Philippines	Russia
TimeGAN	0.0472 ±0.0003	0.0642 ±0.0002	0.0763 ±0.003	0.0891 ±0.001	0.0303 ±0.003	0.0601 ±0.002
RCGAN	0.0353 ±0.0002	0.0612 ±0.0003	0.0388 ±0.001	0.0799 ±0.002	0.0478 ±0.002	0.0588 ±0.001
DG	0.0287 +0.0001	0.0587 +0.0001	0.0407 +0.002	0.0678 +0.002	0.0509 +0.002	0.0418 +0.002

The predictive power of the generated data was assessed through their usefulness for prediction purposes. LSTMs with the same architecture (see key hyperparameters in Table 4) were trained and tested via the TRTR and TSTR schemes, as described in subsection 2.2.3.

Table 8 reports how synthetic data improved the performance of the prediction algorithm (LSTM) regarding the accuracy of prediction, which was evaluated by MAE, MAPE, and MSE values between the TRTR and TSTR settings. This improvement was due to the richness (more diverse examples) and fidelity of the synthetic data produced by the framework. The outperformance of the RCGAN and DG was confirmed again by all three assessment criteria.

Table 8. Predictive scores

Index	Model	MAPE %		MAE		MSE	
		TSTR	TRTR	TSTR	TRTR	TSTR	TRTR
S&P 500	TimeGAN	1.84	2.07	3.05	4.85	0.0036	0.0059
	RCGAN	1.75	2.07	2.60	4.85	0.0029	0.0059
	DG	1.39	2.07	1.31	4.85	0.0017	0.0059
Brazil	TimeGAN	1.89	2.75	3.90	4.50	0.0026	0.0067
	RCGAN	1.74	2.75	2.99	4.50	0.0019	0.0067
	DG	1.60	2.75	2.06	4.50	0.0012	0.0067
India	TimeGAN	1.67	2.31	3.75	4.64	0.0032	0.0052
	RCGAN	1.35	2.31	2.60	4.64	0.0019	0.0052
	DG	1.59	2.31	1.43	4.64	0.0018	0.0052
Malaysia	TimeGAN	1.64	2.56	3.45	4.45	0.0025	0.0061
	RCGAN	1.51	2.56	2.47	4.45	0.0016	0.0061
	DG	1.39	2.56	1.23	4.45	0.0013	0.0061
Philippines	TimeGAN	1.86	2.87	4.32	4.66	0.0045	0.0088
	RCGAN	1.32	2.87	3.31	4.66	0.0023	0.0088
	DG	1.41	2.87	3.97	4.66	0.0032	0.0088
Russia	TimeGAN	1.87	2.21	3.35	4.34	0.0036	0.0064
	RCGAN	1.55	2.21	2.34	4.34	0.0025	0.0064
	DG	1.18	2.21	1.84	4.34	0.0018	0.0064

3.2. Robustness check

To check the robustness of our results and the generalization of our framework regarding the extreme variations modeling, we performed specific examinations on the independent test dataset (one-third of the original dataset). We considered extreme events as the movements of stock prices smaller or greater than the following quantity:

$$\mu \pm 2 * \sigma \quad (6)$$

where μ and σ are the mean value and the SD, respectively, of the training sample variations. Once these data points were identified, we specifically compared the MAE, MAPE, and MSE metrics of the predictions of these events in terms of the TSTR and TRTR procedures. Table 8 presents the mean values of the three tested algorithms.

Table 9. Predictive-score results for extreme events

Index	MAPE %		MAE		MSE	
	TSTR	TRTR	TSTR	TRTR	TSTR	TRTR
S&P 500	1.55	2.55	2.45	4.45	0.0026	0.0053
Brazil	1.17	2.86	2.95	4.67	0.0028	0.0061
India	1.72	2.16	2.31	5.05	0.0024	0.0057
Malaysia	1.97	2.37	2.74	5.34	0.0021	0.0067
Philippines	1.18	2.77	3.33	5.15	0.0025	0.0082
Russia	1.36	2.48	2.18	4.88	0.0020	0.0073

The results in Table 8 indicate a clear improvement in the forecasting of extreme events by using synthetic data. The quality of extreme event forecasting is almost as good as the prediction of the points from the bulk of the distribution (Table 8).

4. Conclusion

In this work, we developed a framework based on GAN algorithms to simulate extreme stock-market variations by generating synthetic data. This framework can efficiently capture the underlying dynamic of stock data and help tackle the problem of extreme event-data scarcity. Experiments performed on the data from the S&P 500 and five emerging markets show that adding synthetic data to the training

process improves the prediction accuracy of extreme events. The quality of the generated data was evaluated by quantitative and qualitative metrics. The results of this study have significant practical implications for investors, traders, and corporations willing to anticipate the future trends of their financial assets. The proposed framework fills the gap in the usage of GAN for stock market simulation and it is a tool to mimic the stock market's extreme behaviors. For future research, we suggest testing more GAN architectures and using advanced techniques for hyperparameter tuning. We also suggest diversifying the metrics used to assess the generated data. Trading strategies can also be simulated on the generated data to verify to which extent the framework can be profitable.

Declarations

Author contributions: All the authors contributed equally to the writing of this paper. All the authors have read and approve the final paper.

Funding statement: None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest: The authors declare no conflict of interest.

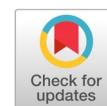
Additional information: No additional information is available for this paper.

References

- [1] S. Ghosh, "COVID-19, stock market, exchange rate, oil prices, unemployment, inflation, geopolitical risk nexus, the case of the BRICS nations: Evidence quantile regression," *Impact Glob. Issues Int. Trade*, pp. 86–105, Jun. 2021, doi: [10.4018/978-1-7998-8314-2.CH005](https://doi.org/10.4018/978-1-7998-8314-2.CH005).
- [2] P. F. Dai, X. Xiong, Z. Liu, T. L. D. Huynh, and J. Sun, "Preventing crash in stock market: The role of economic policy uncertainty during COVID-19," *Financ. Innov.*, vol. 7, no. 1, pp. 1–15, Dec. 2021, doi: [10.1186/S40854-021-00248-Y](https://doi.org/10.1186/S40854-021-00248-Y).
- [3] H. Liu, A. Manzoor, C. Wang, L. Zhang, and Z. Manzoor, "The COVID-19 Outbreak and Affected Countries Stock Markets Response," *Int. J. Environ. Res. Public Heal.* 2020, vol. 17, no. 8, p. 2800, Apr. 2020, doi: [10.3390/IJERPH17082800](https://doi.org/10.3390/IJERPH17082800).
- [4] M. Mazur, M. Dang, and M. Vega, "COVID-19 and the march 2020 stock market crash. Evidence from S&P1500," *Financ. Res. Lett.*, vol. 38, p. 101690, Jan. 2021, doi: [10.1016/J.FRL.2020.101690](https://doi.org/10.1016/J.FRL.2020.101690).
- [5] M. Uddin, A. Chowdhury, K. Anderson, and K. Chaudhuri, "The effect of COVID – 19 pandemic on global stock market volatility: Can economic strength help to manage the uncertainty?," *J. Bus. Res.*, vol. 128, pp. 31–44, May 2021, doi: [10.1016/J.JBUSRES.2021.01.061](https://doi.org/10.1016/J.JBUSRES.2021.01.061).
- [6] I. Shaikh, "Impact of COVID-19 pandemic disease outbreak on the global equity markets," *Econ. Res. Istraživanja*, vol. 34, no. 1, pp. 2317–2336, 2021, doi: [10.1080/1331677X.2020.1863245](https://doi.org/10.1080/1331677X.2020.1863245).
- [7] A. Coletta *et al.*, "Towards Realistic Market Simulations: A Generative Adversarial Networks Approach," *ICAIF 2021 - 2nd ACM Int. Conf. AI Financ.*, pp. 1-9, Nov. 2021, doi: [10.1145/3490354.3494411](https://doi.org/10.1145/3490354.3494411).
- [8] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques – Part II: Soft computing methods," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5932–5941, Apr. 2009, doi: [10.1016/J.ESWA.2008.07.006](https://doi.org/10.1016/J.ESWA.2008.07.006).
- [9] C. L. Chang, M. McAleer, and Y. A. Wang, "Herding behaviour in energy stock markets during the Global Financial Crisis, SARS, and ongoing COVID-19*," *Renew. Sustain. Energy Rev.*, vol. 134, p. 110349, Dec. 2020, doi: [10.1016/J.RSER.2020.110349](https://doi.org/10.1016/J.RSER.2020.110349).
- [10] N. Engelhardt, M. Krause, D. Neukirchen, and P. N. Posch, "Trust and stock market volatility during the COVID-19 crisis," *Financ. Res. Lett.*, vol. 38, p. 101873, Jan. 2021, doi: [10.1016/J.FRL.2020.101873](https://doi.org/10.1016/J.FRL.2020.101873).
- [11] M. Ambros, M. Frenkel, T. L. D. Huynh, and M. Kilinc, "COVID-19 pandemic news and stock market reaction during the onset of the crisis: evidence from high-frequency data," vol. 28, no. 19, pp. 1686–1689, 2020, doi: [10.1080/13504851.2020.1851643](https://doi.org/10.1080/13504851.2020.1851643).
- [12] W. Jiang, "Applications of deep learning in stock market prediction: Recent progress," *Expert Syst. Appl.*, vol. 184, p. 1-97, Dec. 2021, doi: [10.1016/J.ESWA.2021.115537](https://doi.org/10.1016/J.ESWA.2021.115537).
- [13] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nat.* 2015 5217553, vol. 521, no. 7553, pp. 436–444, May 2015, doi: [10.1038/nature14539](https://doi.org/10.1038/nature14539).

- [14] S. C. Huang, C. F. Wu, C. C. Chiou, and M. C. Lin, "Intelligent FinTech Data Mining by Advanced Deep Learning Approaches," *Comput. Econ.*, vol. 59, no. 4, pp. 1407–1422, Apr. 2022, doi: [10.1007/S10614-021-10118-5](https://doi.org/10.1007/S10614-021-10118-5).
- [15] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018, doi: [10.1016/J.EJOR.2017.11.054](https://doi.org/10.1016/J.EJOR.2017.11.054).
- [16] D. Murekachiro, T. Mokoteli, and H. Vadapalli, "Predicting emerging and frontier stock markets using deep neural networks," *Adv. Intell. Syst. Comput.*, vol. 1037, pp. 899–918, 2020, doi: [10.1007/978-3-030-29516-5_68](https://doi.org/10.1007/978-3-030-29516-5_68).
- [17] A. E. de Oliveira Carosia, G. P. Coelho, and A. E. A. da Silva, "Investment strategies applied to the Brazilian stock market: A methodology based on Sentiment Analysis with deep learning," *Expert Syst. Appl.*, vol. 184, pp. 1–12, Dec. 2021, doi: [10.1016/J.ESWA.2021.115470](https://doi.org/10.1016/J.ESWA.2021.115470).
- [18] W. Lu, J. Li, J. Wang, and L. Qin, "A CNN-BiLSTM-AM method for stock price prediction," *Neural Comput. Appl.*, vol. 33, no. 10, pp. 4741–4753, May 2021, doi: [10.1007/S00521-020-05532-Z](https://doi.org/10.1007/S00521-020-05532-Z).
- [19] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," *Expert Syst. Appl.*, vol. 174, p. 114582, Jul. 2021, doi: [10.1016/J.ESWA.2021.114582](https://doi.org/10.1016/J.ESWA.2021.114582).
- [20] K. Armanious *et al.*, "MedGAN: Medical image translation using GANs," *Comput. Med. Imaging Graph.*, vol. 79, p. 101684, Jan. 2020, doi: [10.1016/J.COMPMEIMAG.2019.101684](https://doi.org/10.1016/J.COMPMEIMAG.2019.101684).
- [21] M. K. Baowaly, C. C. Lin, C. L. Liu, and K. T. Chen, "Synthesizing electronic health records using improved generative adversarial networks," *J. Am. Med. Informatics Assoc.*, vol. 26, no. 3, pp. 228–241, Mar. 2019, doi: [10.1093/JAMIA/OCY142](https://doi.org/10.1093/JAMIA/OCY142).
- [22] G. Ramponi, P. Protopapas, M. Brambilla, and R. Janssen, "T-CGAN: Conditional Generative Adversarial Network for Data Augmentation in Noisy Time Series with Irregular Sampling," pp.1–12 Nov. 2018, doi: [10.48550/arXiv.1811.08295](https://doi.org/10.48550/arXiv.1811.08295).
- [23] S. Takahashi, Y. Chen, and K. Tanaka-Ishii, "Modeling financial time-series with generative adversarial networks," *Phys. A Stat. Mech. its Appl.*, vol. 527, pp. 1–12, Aug. 2019, doi: [10.1016/J.PHYSA.2019.121261](https://doi.org/10.1016/J.PHYSA.2019.121261).
- [24] W. Tovar, "Deep Learning Based on Generative Adversarial and Convolutional Neural Networks for Financial Time Series Predictions," Aug. 2020, Accessed: May 02, 2023. doi: [10.48550/arXiv.2008.08041](https://doi.org/10.48550/arXiv.2008.08041).
- [25] K. Zhang, G. Zhong, J. Dong, S. Wang, and Y. Wang, "Stock Market Prediction Based on Generative Adversarial Network," *Procedia Comput. Sci.*, vol. 147, pp. 400–406, Jan. 2019, doi: [10.1016/J.PROCS.2019.01.256](https://doi.org/10.1016/J.PROCS.2019.01.256).
- [26] I. Goodfellow *et al.*, "Generative Adversarial Networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Jun. 2014, doi: [10.1145/3422622](https://doi.org/10.1145/3422622).
- [27] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," *Adv. Neural Inf. Process. Syst.*, vol. 2017–December, pp. 5768–5778, Mar. 2017, Accessed: May 02, 2023. doi: [10.48550/arXiv.1704.00028](https://doi.org/10.48550/arXiv.1704.00028).
- [28] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs," pp. 1–13, Jun. 2017, Accessed: May 02, 2023. doi: [10.48550/arXiv.1706.02633](https://doi.org/10.48550/arXiv.1706.02633).
- [29] J. Yoon, D. Jarrett, and M. van der Schaar, "Time-series Generative Adversarial Networks," *Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 1–11, 2019. Available at: <https://papers.nips.cc/paper/8789-time-series-generative-adversarial-networks>.
- [30] Z. Lin, A. Jain, C. Wang, G. Fanti, and V. Sekar, "Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions," *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, vol. 2020–Oct, pp. 464–483, doi: [10.1145/3419394.3423643](https://doi.org/10.1145/3419394.3423643).
- [31] M. Wiese, L. Bai, B. Wood, and H. Buehler, "Deep Hedging: Learning to Simulate Equity Option Markets," *SSRN Electron. J.*, Oct. 2019, pp.1–13, doi: [10.2139/SSRN.3470756](https://doi.org/10.2139/SSRN.3470756).

Automatic note generator for Javanese gamelan music accompaniment using deep learning



Arik Kurniawati ^{a,1}, Eko Mulyanto Yuniarno ^{a,b,2,*}, Yoyon Kusnendar Suprpto ^{a,b,3},
Aditya Nur Ikhsan Soewidiatmaka ^{c,4}

^a Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

^b Department of Computer Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

^c Soewidiatmaka Gamelan, Bandung, Indonesia

¹ arikkurniawati.19071@mhs.its.ac.id; ² ekomulyanto@ee.its.ac.id; ³ yoyonsuprpto@ee.its.ac.id; ⁴ soewidiatmaka@gmail.com

* corresponding author

ARTICLE INFO

Article history

Received February 27, 2023

Revised April 24, 2023

Accepted May 21, 2023

Available online June 1, 2023

Keywords

Bidirectional LSTM

Deep Learning

Gamelan music

Javanese melody

Musical harmonic

ABSTRACT

Javanese gamelan is a traditional form of music from Indonesia with a variety of styles and patterns. One of these patterns is the harmony music of the Bonang Barung and Bonang Penerus instruments. When playing gamelan, the resulting patterns can vary based on the music's rhythm or dynamics, which can be challenging for novice players unfamiliar with the gamelan rules and notation system, which only provides melodic notes. Unlike in modern music, where harmony notes are often the same for all instruments, harmony music in Javanese gamelan is vital in establishing the character of a song. With technological advancements, musical composition can be generated automatically without human participation, which has become a trend in music generation research. This study proposes a method to generate musical accompaniment notes for harmony music using a bidirectional long-term memory (BiLSTM) network and compares it with recurrent neural network (RNN) and long-term memory (LSTM) models that use numerical notation to represent musical data, making it easier to learn the variations of harmony music in Javanese gamelan. This method replaces the gamelan composer in completing the notation for all the instruments in a song. To evaluate the generated harmonic music, note distance, dynamic time warping (DTW), and cross-correlation techniques were used to measure the distance between the system-generated results and the gamelan composer's creations. In addition, audio features were extracted and used to visualize the audio. The experimental results show that all models produced better accuracy results when using all features of the song, reaching a value of around 90%, compared to using only 2 features (rhythm and note of melody), which reached 65-70%. Furthermore, the BiLSTM model produced musical harmonies that were more similar to the original music (+93%) than those generated by the LSTM (+92%) and RNN (+90%). This study can be applied to performing Javanese gamelan music.



This is an open access article under the [CC-BY-SA](#) license.



1. Introduction

Melody and harmony are important concepts in music. Melody refers to the tone and rhythm [1], while harmony is the use of chords or tones together [2]. They work together to create a great musical composition. Melody is usually noticed first when listening to music, followed by harmony, which completes the tune. Harmony music is used in both Western and Eastern music, including Javanese gamelan, a traditional music from Indonesia that uses harmony music as its accompaniment.

Each musical instrument has a distinct role to perform in the creation of a harmonic song. In Western music, for example, the piano, harmonica, saxophone, and violin create the melody, while the guitar, keyboard, piano, and harp give harmony, and the tambourines and drums form the rhythm. Every instrument is designed to perform a specific function based on the notes, chords, and rhythms contained within the piece. Moreover, the chord notation for all instruments is at the same pitch and is not influenced by the rhythm.

However, Javanese gamelan is unique in its harmony music, where the harmonic tones have different variations based on the rhythm, structure, and dynamics of the song [3]. Each instrument has different tones, with the Bonang instrument (consisting of Bonang Barung and Bonang Penerus) playing harmony music. In contrast, the Balungan group (Saron, Demung, and Peking) plays the melody, and the Gong and Kendhang groups are responsible for organizing the rhythmic patterns.

Song in Javanese gamelan comprises a variety of elements including rhythm, *laya*, scale (*laras*), mode (*pathet*), and dynamics. Rhythm is determined by tempo gradation, while *laya* pertains to the speed of the song. *Laras* refers to the scales used in the song, and *pathet* conveys the emotional feeling conveyed by the song. Dynamics highlight the variation, proportionality, and dynamic nature of the musical elements present in a song [4], [5]. Its purpose is not only to entertain but also to convey social, moral, cultural, and spiritual values [6]. The composition is conveyed through a pattern of playing variations, and if the resulting pattern is not appropriate, the beauty and character of the composition will be lost. In Javanese gamelan music, harmony is achieved through variations of the pattern played, which are tailored to the specific characteristics of the song, thereby showcasing the unique character of each piece. Harmony in Javanese gamelan music is unique. A single song can have several variations of harmony, all based on the specific characteristics of the piece. These variations in musical harmony are usually influenced by including rhythm, *laya*, scale (*laras*), mode (*pathet*), and dynamics of song [4], [5]. Fig. 1 shows an example of a song with the same melody but different harmony notes.



Fig. 1. Javanese gamelan music with the same melody but different note in harmony using (a) Gembyang pattern with Irama Lancar (b) Imbal Sekaran pattern with Irama Lancar (c) Mipil pattern with Irama Tanggung

In Indonesia, modern music uses numeric notation for the melody and letters for chords. This system makes it easier to write and read music, and is used in both classical and modern music, including Javanese gamelan. In modern music, all instruments typically use the same sheet music. However, in traditional Javanese gamelan music, each instrument has its own notation. Fig. 2 illustrates the differences in modern and traditional music notation in Indonesia, where in modern music chord notation is usually the same for all instruments, whereas in Javanese gamelan, the harmony notation can be different for each instrument.

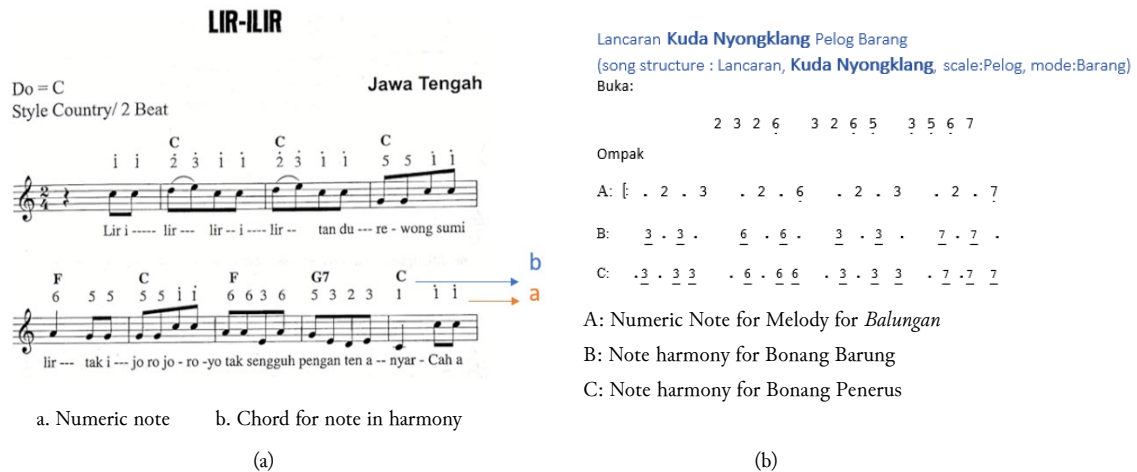


Fig. 2. An Indonesian music notation (a) modern music (b) traditional music in Javanese gamelan

The notation used in Javanese gamelan music is less detailed than shown in Fig. 2b, where only the main melody is written. This means that only experienced gamelan musicians can play all the instruments by reading the information in the song, such as the song structure, scale (*laras*), mode (*pathet*), and dynamics of the music, without knowing the detailed notation for all the instruments. The problem is that novice musicians need complete notation for all instruments to play gamelan music, because the harmony music may have different notation patterns. For example, the song Lancaran Angleng Pelog Barang.

- This song will use Gembyang pattern for harmony music if the dynamic of the song is for music only, and this song will use Imbal Sekaran pattern if the dynamic of the song is for singer accompaniment.
- This song will use Gembyang pattern for harmony music if the rhythm of the song uses Irama Lancar, and this song will use Mipil pattern if this song uses Irama Tanggung.

The selection of the musical harmony pattern is based on the characteristics of a song (such as: rhythm, *laya*, scale (*laras*), mode (*pathet*), and dynamics and the melodic notation of the Balungan instrument). Fig. 3 demonstrates how Javanese gamelan songs are written. Gamelan music uses numeric notes, with the main melody represented by Balungan notes. Fig. 3 provides an example of gamelan music notation and other information contained in songs [3]–[5], [7]–[10], including:

- The song Manyar Sewu is an example of Javanese gamelan music. Its structure is Lancaran, its scale is Slendro, and its mode is Manyura, all of which are included in the song title.
- Javanese gamelan music has various song structures, including Sampak, Srepegan, Ayak-Ayakan, Lancaran, Bubaran, Ketawang, and Ladrang. Each structure is determined by the position of the Kethuk, Kenong, Kempul, Kempyang, and Gong playing, similar to a genre in modern music.
- There are two musical scales in Javanese gamelan: Slendro, which has five notes (1, 2, 3, 5, and 6), and Pelog, which has seven notes (1, 2, 3, 4, 5, 6, and 7). The musical scale is called *laras*.

- *Pathet* is a system of musical modes, and the Slendro is composed of Manyura, Nem, and Sanga, while the Pelog comprises the notes Barang, Lima, and Nem. The *pathet* helps create the atmosphere of the song.
- Buka is the opening of the song, which is represented by the Bonang Barung note.
- Rhythm (*irama*) is usually found at the start of the section.
- Ompak is a song section represented by the Balungan note. Each part is usually organized into multiple lines called *gongan*. Depending on the song structure, there may be multiple *gatra* in a single line, and each *gatra* consists of several notes from the Balungan based on rhythm.
- In gamelan music, the Balungan notes are represented by symbols, such as a circle for Gong Ageng, smiley and frowny faces for Gong Suwuk, a smiley face for Kempul, and a frowny face for Kenong. A dot above a number and the lower octave by a dot below a number indicates the upper octave. Additionally, a dot in place of a number indicates a break or interruption.

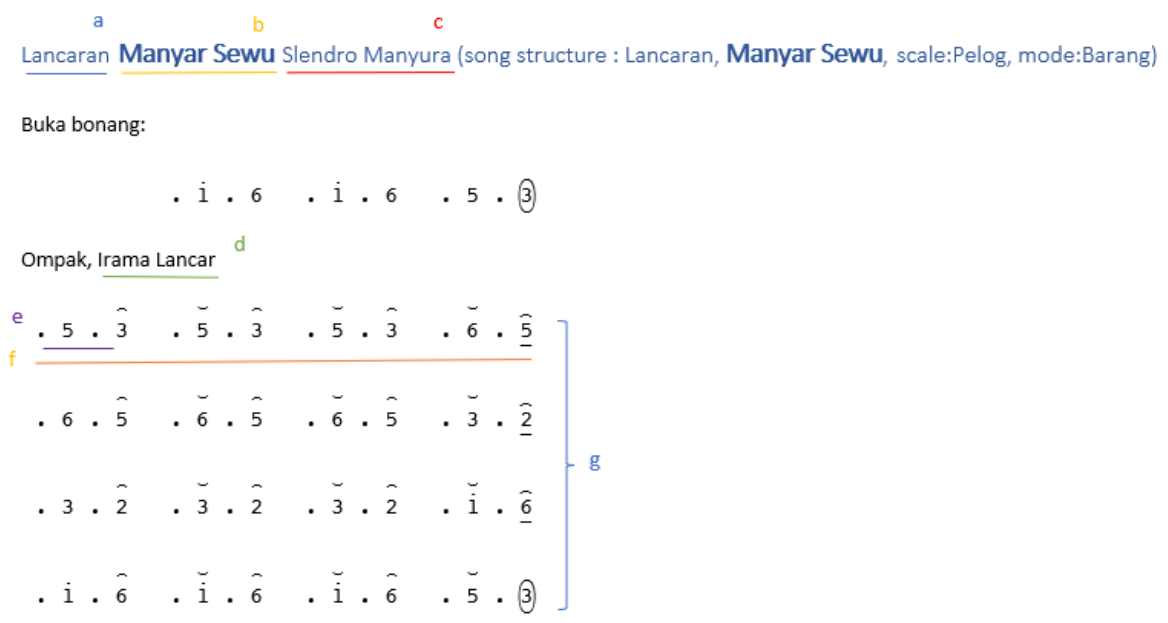


Fig. 3. A song of Javanese *gamelan* (a) song structure (b) name (c) scale and mode of the song (d) rhythm (e) *gatra* (f) *gongan* (g) melody note

All the characteristics of a Javanese *gamelan* song have a significant impact on the resulting variations of harmony music. Even with the same melody, differences in rhythm can result in different harmony music, as demonstrated in Fig. 1a and Fig. 1c. Likewise, variations in song dynamics can lead to different harmonies, as shown in Fig. 1b for a singer's accompaniment and Fig. 1a for music alone. Therefore, the song's characteristics that heavily influence the harmony of Javanese *gamelan* music are song structures, rhythm, scale (*laras*), mode (*pathet*), and song dynamics.

Based on the previous explanation, the problems discussed in this study are:

- Javanese *gamelan* music has various harmony note variations depending on the characteristics of the song, such as song structure, rhythm, scale (*laras*), mode (*pathet*), and dynamic of the song.
- Notation for all instruments in a Javanese *gamelan* song is not fully written, only the melody notation, which becomes a difficulty for novice *gamelan* musicians.

This study proposes a deep learning model for composing musical accompaniment as note harmony in Javanese *gamelan* using detailed information about a song. This study aims to assist novice *gamelan* musicians in learning the harmony music in Javanese *gamelan*. The main contributions of this paper are presented as follows:

- Proposing a simplified method for representing music data as input, utilizing numeric notes.
- Proposing a deep learning model to generate musical accompaniment as note harmony using detailed information about a song.
- Using song characteristic for features such as song structure, *gatra*, and rhythm, this study successfully generates music accompaniment for music harmony, including Bonang Barung and Bonang Penerus.

The structure of this study is as follows: Section 1 introduces the context and significance of the research, and Section 2 discusses earlier studies that have been done on Music Generation, as well as the details of BiLSTM, LSTM, and RNN. Additionally, it includes a discussion of the dataset used for this research and methods for creating an accompaniment Javanese gamelan music generator of the Bonang Barung and Bonang Penerus. Section 3 presents the outcomes of the experiments conducted and their analysis. Finally, Section 4 concludes by summarizing the findings of this study and outlining future work.

2. Method

2.1. Related Work

Deep Learning has rapidly expanded in recent years, solving many complex Artificial Intelligence issues. Deep Learning models are effective at solving various problems, including recognition [11], regression [12], semi-supervised and unsupervised problems [13] for medical diagnosis [14], natural language [15] and image processing [16], and prediction system [17]. These models learn hierarchical features from different data types, including numerical, image, text, and audio. There are different types of Deep Learning neural networks, such as Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN).

This study suggests using Deep Learning to create music with a type of RNN. Typically, only professional composers can create music, but this study aims to help beginners create their own. This is called music generation, and it uses AI to create its own music. This is done by combining AI and art, producing music without human interference.

Research on music generation is conducted to generate melodies [18], polyphonic [19], multitrack [20], and predict musical accompaniment to a given melody [21]. Most of the generators are polyphonic [19]–[23]. Data is typically separated into audio and symbols due to the different forms in which music is presented [22]–[24]. Meanwhile, audio data is used more often, and symbols are also important because melodies are read through notation [22]. Artificial neural networks (ANNs), such as RNNs and their variations, are currently used in music generation [4], [24]–[31]. ANNs can predict musical notes based on previously learned musical structures without the help of a human expert to impart musical knowledge and analysis. RNNs and their variations are particularly useful in music generation because they can remember the musical notes created in the past and use that information to predict future notes.

Recurrent neural networks (RNNs) are unique in their ability to remember past data and use it to predict future data. Therefore, many researchers have used RNNs to create automatic music. For instance, RNNs have been used to predict melodic pitch and generate new music in MIDI format [24], as well as to generate sheet music with a defined structure and style, without imposing the rules of musical composition on the model using ABC Notation [25].

LSTM networks are a type of RNN that can solve long-term dependency problems. Unlike conventional RNNs, which can struggle to link previous and new information, LSTMs store information in memory for an extended period. Although RNNs and LSTMs have similar structures, LSTMs have different memory or repeating modules. Researchers have used LSTM networks for music generation, including creating melodies and harmonies from a MIDI file using a single-layer LSTM [26]. Multiple-layer LSTMs have also been used to generate musical notes, preserving a song's characteristics, and

resulting in better musical variance [27]. In another study, the dataset was preprocessed before entering the LSTM, and the results were reconstructed into sheet music in ABC format, producing pleasant-sounding music [28].

Bidirectional Long Short-Term Memory (BiLSTM) is a variant of LSTM which can process sequence information in two directions, past to future and future to past. This allows the network to preserve both past and future knowledge and generate more relevant output, making it useful in music generation tasks as well as NLP tasks. Researchers have studied the use of BiLSTM with Dynamic Time Wrapping to identify relative content and measure similarity between two temporal sequences [29], generating accurate next music with epoch variations [30], creating note sequences for classical piano [31], and generating unique music with attention [4].

The BiLSTM is a neural network that is an advanced version of the LSTM architecture and belongs to the RNN family [32]. It is capable of better contextual understanding due to its ability to process input sequences in both forward and backward directions. This allows the network to take into account contextual information from both past and future inputs, leading to more precise predictions. Furthermore, the complexity of the BiLSTM architecture makes it less prone to overfitting compared to LSTM and RNN.

This study proposes a BiLSTM model for generating music accompaniment as music harmony from numerical notes and song characteristic features, such as rhythm, melody, scale (*laras*), and mode (*pathet*) in Javanese gamelan. The proposed model aims to overcome the limitations of previous works [4], [29]–[31], which only focused on predicting the next melody to generate new notes.

2.2. Bidirectional LSTM (BiLSTM)

Composers often choose notes based on their context, both before and after, which cannot be easily handled by simple RNNs and LSTMs. To overcome this limitation, a Bidirectional LSTM combines two independent RNNs and takes data from both ends of the input flow to model sequential dependencies in both directions, as illustrated in Fig. 7. In addition, BiLSTM adds another LSTM layer that flows the input sequence backward. The outputs from the two LSTM layers can be combined in various ways, such as averaging, summing, multiplying, or concatenating. BiLSTM is particularly useful in music generation since it can preserve both future and previous knowledge, resulting in more relevant output.

The LSTM network uses activation values other than Candidate values (C). This LSTM network has two outputs, namely the current value of the memory cell ($C^{<t>}$), and the output value or hidden state ($a^{<t>}$), which is defined according to equations (1) and (2).

$$C^{<t>} = \Gamma_u * C^{N<t>} + \Gamma_f * C^{<t-1>} \quad (1)$$

$$a^{<t>} = \Gamma_o * C^{<t>} \quad (2)$$

Thus, the result of calculating the potential value of the memory cell is defined in equation (3).

$$C^{N<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \quad (3)$$

There are separate gate equivalents (3), (4), (5) in the LSTM to control memory cells, i.e.:

$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u) \quad (4)$$

$$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f) \quad (5)$$

$$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o) \quad (6)$$

Where W and b are weight matrices and vectors, t is the current iteration of the recurrent network Γ_u , the update gate, Γ_f the forget gate, Γ_o the output gate. Therefore, if all of these components are combined, the LSTM cell is shown in Fig.7.

2.3. Dataset

The data musical notes for this study used only Lancaran song structure, which was collected from <http://www.gamelanbvg.com/>. The Lancaran is a short Javanese gamelan song with a variety of harmonic patterns, often used in traditional arts such as shadow puppets and traditional dances. Since the dataset used in this study only includes the Lancaran, the song structure is not a considered feature, unless the dataset is expanded to include all types of song structures. The dataset includes notes for Balungan, rhythm, melodic notes at the end of lines and melodic notes at the end of the song, scale, and mode of music were obtained from Javanese gamelan song, and the dynamic of song was determined by gamelan expert. The note of Bonang Barung and Bonang Penerus were created by gamelan expert as the output.

The data used were ten songs as training and validation data, divided between training (80% of data) and validation (20% of data) and one song as test data. Each song had three different harmonic music notation structures: Gembyang pattern with an Irama Lancar, Imbal Sekaran pattern with an Irama Lancar, and Mipil pattern with an Irama Tanggung. Therefore, a total of 33 songs, divided into approximately 400 *gatra*, were used in this study, as illustrated in Fig. 4. While Table 1 lists the songs used in this study.

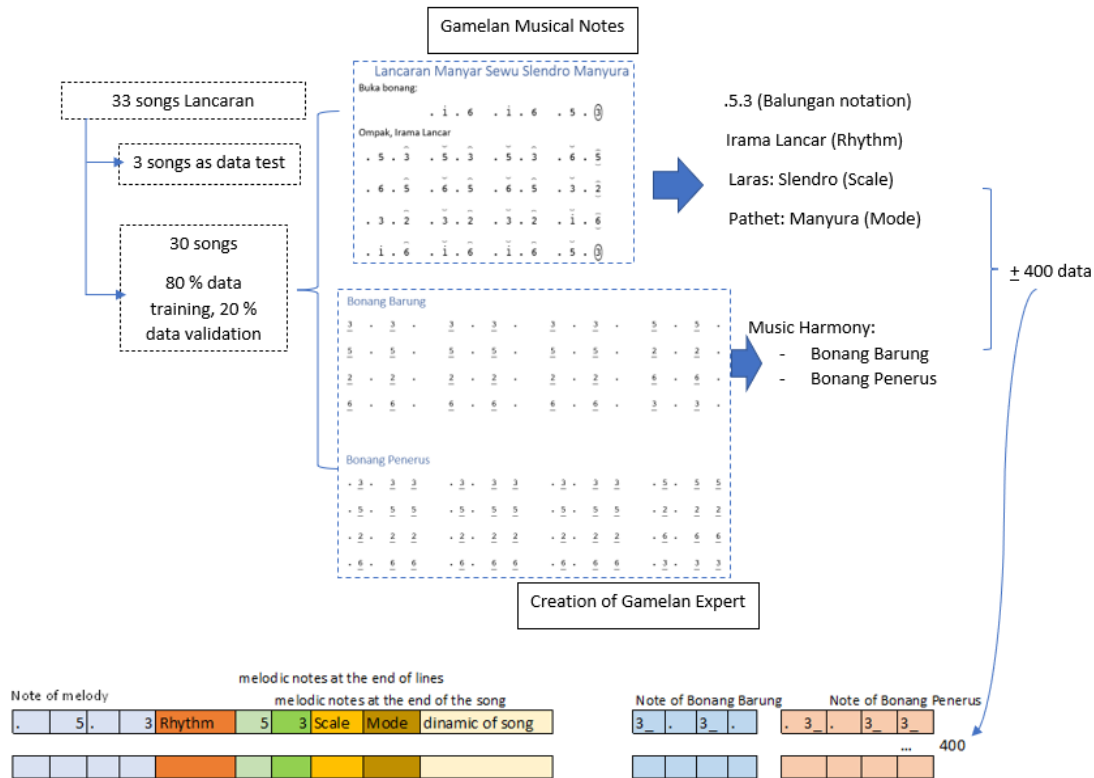


Fig. 4. Dataset Representation

Each song's melody notes, represented by the Balungan, are divided into multiple lines. Each line is composed of several *gatra*, with each *gatra* containing several notes. The number of bonang notations, including Bonang Barung and Bonang Penerus, depends on the rhythm used. For each Balungan note in each *gatra*, there is either one Bonang Barung note or 1-2 Bonang Penerus notes (Irama Lancar), two Bonang Barung notes or 2-4 Bonang Penerus notes (Irama Tanggung), or four Bonang Barung notes or 4-8 Bonang Penerus notes (Irama Dadi), as illustrated in Fig. 5.

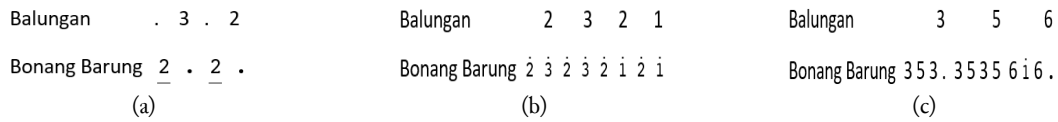


Fig. 5. Structure Notation for (a) *Irama Lancar* (b) *Irama Tanggung* (c) *Irama Dadi*

Table 1. List of songs for dataset

No	Song Titles *	Dataset		
		Training	Validation	Test
1	<i>Lancaran Kuda Nyongklang Pelog Barang</i>	Yes	Yes	No
2	<i>Lancaran Maesa Kurda Slendro Sanga</i>	Yes	Yes	No
3	<i>Lancaran Manyar Sewu Slendro Manyura</i>	Yes	Yes	No
4	<i>Lancaran Rena Rena Slendro Manyura</i>	Yes	Yes	No
5	<i>Lancaran Sarung Jagung Pelog Barang</i>	Yes	Yes	No
6	<i>Lancaran Angleng Pelog Barang</i>	Yes	Yes	No
7	<i>Lancaran Bendrong Pelog Nem</i>	Yes	Yes	No
8	<i>Lancaran Kandhang Bubrab Pelog Nem</i>	Yes	Yes	No
9	<i>Lancaran Kedbu Pelog Barang</i>	Yes	Yes	No
10	<i>Lancaran Lasem Bubrab Pelog Nem</i>	Yes	Yes	No
11	<i>Lancaran Ricik Ricik Slendro Manyura</i>	No	No	Yes

* Lancaran is song structure, Pelog/Slendro is scale, Barang/Sanga/Manyura/Nem is mode

Harmony music in Javanese gamelan is produced by the Bonang group, which includes Bonang Barung and Bonang Penerus. Both instruments have the same layout and design, differing only in their musical scale, Pelog or Slendro, as shown in Fig. 6. The Bonang Barung and Bonang Penerus have a medium melody in the bottom layout and a high melody in the top layout. They can be played using a single or double melody. Writing a note in Bonang Barung can be done with one note if played in a single melody or a double note with medium and high notes played simultaneously. For example, 3 represents playing Bonang 3 in the lower and upper octaves simultaneously.

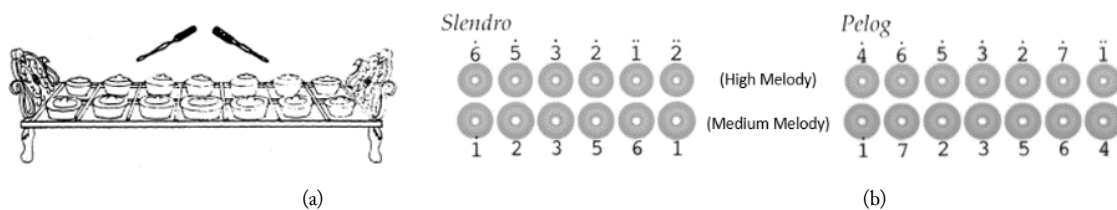


Fig. 6. (a) *Bonang* kettles (b) Layout of *Slendro* and *Pelog* *Bonang*

2.4. Prediction of Notes Harmony as Accompaniment Music

The proposed method in this study is the Bidirectional LSTM (BiLSTM) method for generating musical accompaniment to traditional Javanese gamelan music, as illustrated in Fig. 7. The input and output of the system are as follows.

- Input: notes for *Balungan*, rhythm, a note at the end of lines, a note at the end of the song, scale, mode, dynamic of song.
- Output: sequence of notes harmony for Bonang Barung and Bonang Penerus instruments

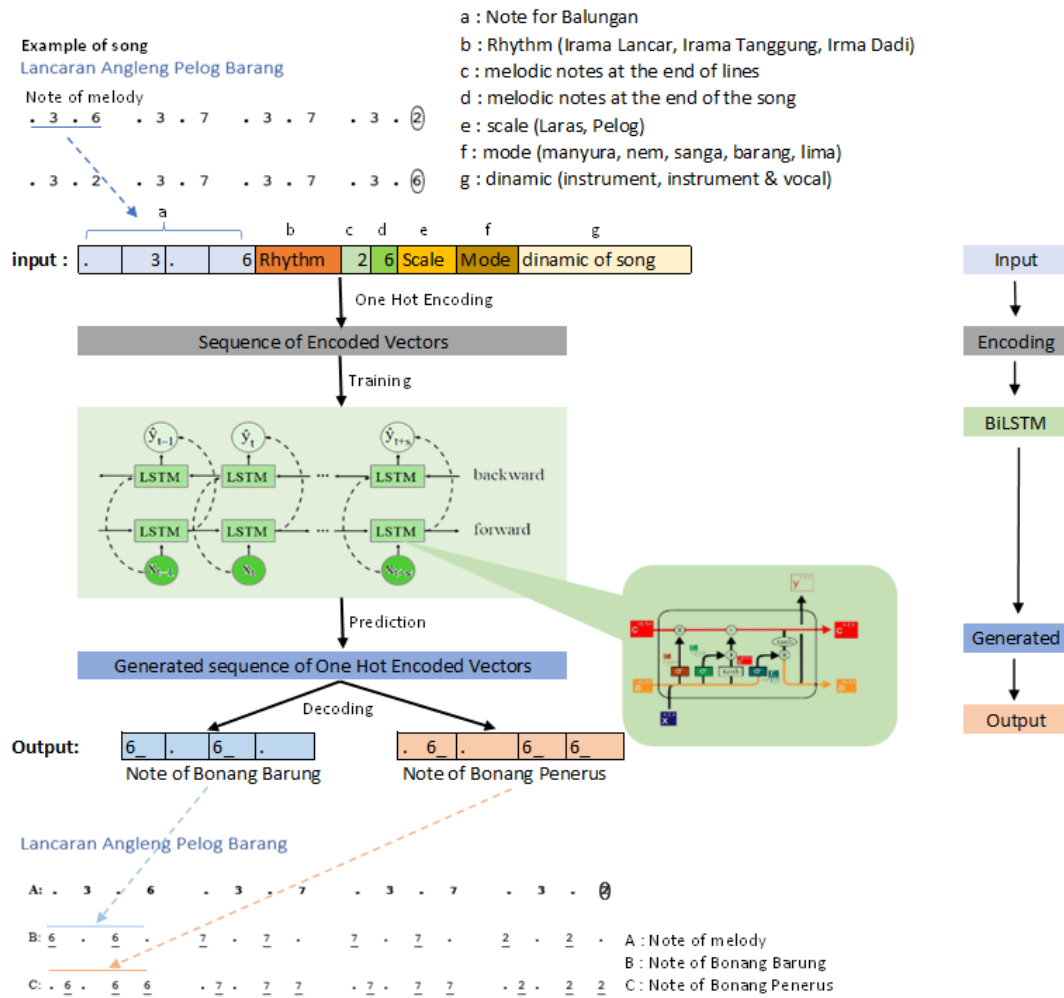


Fig. 7. Diagram of proposed method BiLSTM for prediction of notes harmony in Javanese gamelan

2.4.1. Data Encoding

Before using the input and output data in the BiLSTM, the input and output data need to be encoded into the correct input vector using a one-hot encoding mechanism. Each element of the input/output data is assigned a suitable true/false value. For example, if a particular note exists at a specific time step, the value of the column corresponding to that number "1" will be true, and all other settings will be false. Fig. 8 shows an illustration of a hot coding mechanism, where each column shows the vector code for a different note, rhythm, scale, mode and dynamic in the song for each data. The resulting vector is used as input to the BiLSTM, LSTM, and RNN models.

note in song				rhythm in song		scale in song	
	. 3 . 6			Irama Lancar	Lancar	Laras	Pelog
all available note	1 0 0 0	all available rhythm	1	Irama Tanggung	0	all available scale	0
	1' 0 0 0					Pelog	1
	2 0 0 0			mode in song		dynamic in song	
	2' 0 0 0			all available mode	Barang	all available dynamic	Instrument
	3 0 1 0				0		Instrument
	3' 0 0 0				0		Instrument+Vocal
	5 0 0 0				0		1
	6 0 0 0				1		0
	6. 0 0 0				0		
	7 0 0 0				0		
	7. 0 0 0				0		

Fig. 8. Encoding Technique (a)Note (b)Rhythm (c)Scale (d)Mode (e) Dinamic of song

2.5. BiLSTM Model

This section explains the structure of the BiLSTM model. Fig. 9a shows BiLSTM architecture, which includes an input layer of size 66, 256 LSTM cells, and 256 dropout cells to prevent overfitting. The output layer has 24 dense layers, with 8 for Bonang Barung and 16 for Bonang Penerus. The input includes Balungan note, rhythm, a note at the end of lines, a note at the end of the song, scale, mode, and dynamic of song. The input layer is the first layer of the model, followed by the BiLSTM (LSTM 128) and Dropout (0.2) layers. This is followed by a dense layer and a sigmoid layer as the activation layer for the output of the network. The sigmoid layer can predict accompanying notes in a piece of music. By training the model with an original dataset and testing it with new data, the model can generate accompanying music.

For comparison, this BiLSTM model is also compared with LSTM and RNN models. The structure of the LSTM and RNN architectures is shown in Fig. 9b and Fig. 9c, which are similar to the BiLSTM architecture. The results section discusses the performance of these models.

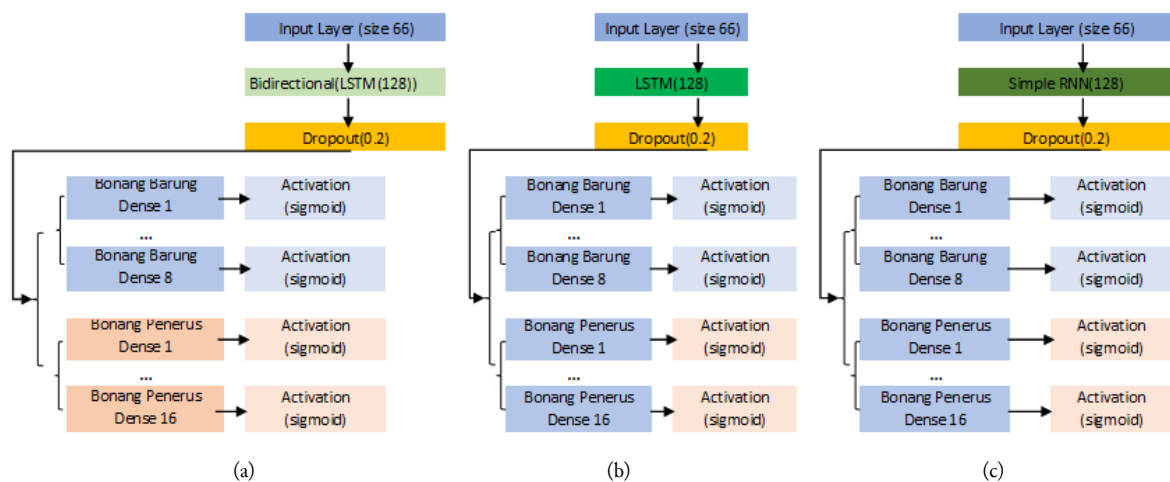


Fig. 9. Diagram architecture (a) BiLSTM (b) LSTM (c) RNN for prediction of notes harmony Javanese *gamelan*

2.6. Generated of sequences of note in harmony

After the BiLSTM network is trained, it can generate a sequence of musical notes for harmonic accompaniment. A dataset with different variations in the composition of musical notes was used to improve prediction and generate diverse sequences with three harmonic musical patterns. The model automatically predicts Bonang Barung and Bonang Penerus notes based on test data containing Balungan note, rhythm, a note at the end of lines, a note at the end of the song, scale, mode, and dynamic of the song.

3. Results and Discussion

In this study, we compared our proposed BiLSTM model with RNN and LSTM models to predict harmonious notes in Javanese *gamelan* music. To analyze its performance and the resulting harmonic notes, we evaluated the model's effectiveness using prediction accuracy and loss values. Then, we selected a sample song (Lancaran Ricik Ricik Slendro Manyura, not included in the training data) to compare with the generated and then evaluated the similarity using musical analysis.

This study presents two experiment scenarios to evaluate the performance of the proposed model. In the first scenario, all features (including notes for Balungan, rhythm, a note at the end of lines, a note at the end of the song, scale, mode, and dynamic of the song) are used as input for the model. In the second scenario, only two features—the note of melody and rhythm—are used. In Javanese *gamelan*, melody and rhythm are typically the most important factors in determining the harmony of a song. However, sometimes other factors must be considered because different songs with the same melody and rhythm may produce different harmonic notes. This is due to various song characteristics, such as

the types of modes and scales, the dynamic of songs (accompaniment of singers or just instruments), and the final note of each line and the song.

3.1. Quantitative Analysis

This section compares the accuracy and loss values of three models: RNN, LSTM, and BiLSTM in two scenarios (with all features and with only two features). Table 2 shows their loss values and prediction accuracies during the training phase. The dataset consisted of 400 data points and was trained with all three models. From Table 2 we can see that BiLSTM has the highest accuracy values compared to LSTM and RNN for all scenarios and all instruments, although the difference is small, around 0.01-0.05 for accuracy values, and the lowest loss value for all scenarios and all instruments, with a difference of around 0.01-0.1. These results were obtained after training the models for 100 epochs with a batch size of 4.

Table 2 also shows that using all song features improved the accuracy of all models. This shows that using all the features of a song can improve the success rate in generating musical accompaniment compared to using only 2 features (Balungan notes as melody notes and rhythm).

Fig. 10 shows the training and validation accuracy and loss values of the 400 data points trained using BiLSTM, LSTM and RNN. The three graphs show that BiLSTM, LSTM, and RNN have similar results, reaching stability at around epoch 40 in the training process.

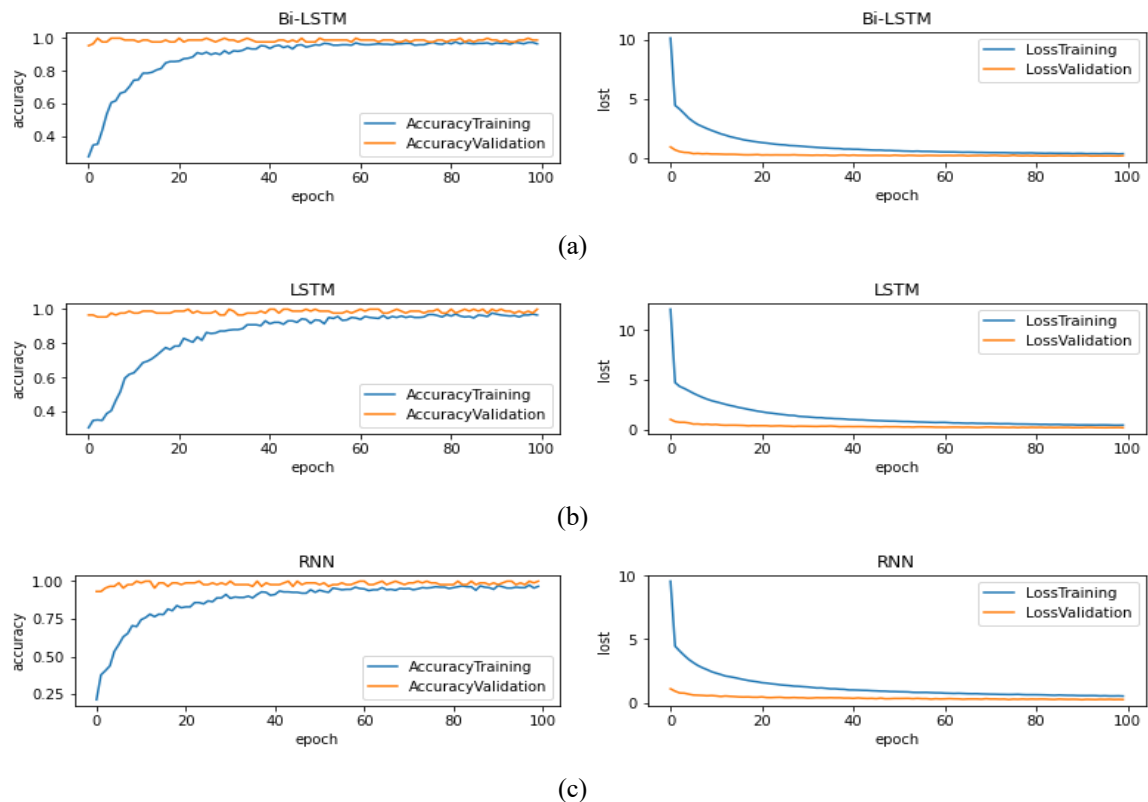


Fig. 10. Graph accuracy and loss value of architecture (a) Bi LSTM (b) LSTM (c) RNN

3.2. Musical Analysis

Musical analysis is used in this study, including note distance, spectrogram [33], dynamic time warping (DTW) [34] and cross-correlation [35]. The aim is to provide additional insight, analyze the effects of musical harmony, and evaluate the similarity between the original music generated by the models, BiLSTM, LSTM, and RNN respectively. This multi-faceted approach ensures a more comprehensive analysis and a more accurate assessment of the similarity of the generated music to the original music.

Based on Table 2, the accompaniment music generator analyzed using note distance, spectrograms, dynamic time warping (DTW), and cross-correlation are scenarios using all features because the accuracy results are better when compared to those using 2 features.

3.2.1. Music Generation Result

After training the three BiLSTM, LSTM, and RNN models, the system can produce harmonic music in numerical notation for both Bonang Barung and Bonang Penerus notations, when all features are used as input. In this study, the Lancaran Ricik Ricik Slendro Manyura song was used as test data for three different harmonic music models: Gembyang pattern for Irama Lancar, Imbal Sekaran pattern for Irama Lancar, and Mipil pattern for Irama Tanggung. Fig. 11, Fig. 12, Fig. 13 show the results of testing the Bonang Barung and Bonang Penerus notes for this song with three patterns.

- In the Gembyang pattern for the Lancaran Ricik Ricik Slendro Manyura song, the system generated identical harmonic music for the Bonang Barung and Bonang Penerus instruments as created by the gamelan composer (Pangrawit), as shown in Fig. 11.
- The Mipil pattern for the Lancaran Ricik Ricik Slendro Manyura song also produces harmony music of the Bonang Barung and Bonang Penerus instruments that are similar to those created by Pangrawit, as shown in Fig. 12.
- The Imbal Sekaran for the Lancaran Ricik Ricik Slendro Manyura song, the system produced harmony music for the Bonang Barung and Bonang Penerus instruments, shown in Fig. 13.

The Imbal Sekaran pattern is unique because it produces different notes despite having the same Balungan notes and rhythm as the Gembyang pattern. This is because the musical harmony notes are influenced by various song features such as the song's use, final note of each line, last note of the song, and its scale and mode. It can be seen in Fig. 13a and Fig. 13b that there is a slight difference between the notation created by the gamelan composer and the result of the music generator produced by BiLSTM, LSTM, and RNN. However, according to the analysis of gamelan experts, this difference does not significantly affect the sound of the gamelan produced later, because the notation that is the Dhing Dhong gamelan (marked with a red box) is still the same as the gamelan composer's creation.

To further analyze the results of the system, the harmonic tones generated by the system are combined with the melodic tones of the Balungan instrument using the Gamelan Synthesis application. This evaluation is done because there is no difference between the instruments used in the Gembyangan and Mipil patterns created by the gamelan composer and the results of the generating system. Therefore, the Imbal Sekaran pattern for the Lancaran Ricik Ricik Slendro Manyura song is evaluated to assess the performance of the system's generator. This study visualizes audio features using Spectrogram and Amplitude vs. Frequency and calculates the correlation between two audio files using Note Distance, DTW, and Cross Correlation to evaluate similarities between the original music and the resulting results from BiLSTM, LSTM, and RNN.

Table 2. Accuracy and lost value of prediction for different models

Model	Loss and Accuracy	Scenario			
		All Features		2 Features	
		Bonang Barung	Bonang Penerus	Bonang Barung	Bonang Penerus
BiLSTM	Loss	0.042	0.034	0.120	0.110
	Accuracy	0.912	0.949	0.656	0.727
LSTM	Loss	0.041	0.033	0.112	0.106
	Accuracy	0.911	0.947	0.663	0.728
RNN	Loss	0.046	0.038	0.110	0.108
	Accuracy	0.906	0.943	0.665	0.732

3.2.2. Note Distances

Note distances are used for calculations to analyze the similarity of musical harmony between the original music and the music produced. This distance is called the exact distance, and its value contains a binary indicator.

$$N_o(Nota_1, Nota_2) = \begin{cases} 0 & \text{if } Nota_1 = Nota_2 \\ 1 & \text{if } Nota_1 \neq Nota_2 \end{cases} \quad (7)$$

As shown in Fig. 13, these values were obtained by calculating the difference between the original notation of the gamelan composer and the generator results of the BiLSTM, LSTM, and RNN models of both instruments, Bonang Barung and Bonang Penerus. Based on equation (7), the note distance value for the test data of the Gembyang and Mipil patterns is 0, as shown in the generator results in Fig. 11 and Fig. 12, where for all models BiLSTM, LSTM, and RNN are similar to the creations of gamelan composers.

In contrast, for the Imbal Sekaran pattern, the value of the note distance varies depending on the generator model. The Bi-LSTM model has a note distance value of 7, the LSTM model has a value of 9 and the RNN model has a value of 10. This means that the result generated by BiLSTM is closer to the original song than LSTM and RNN.

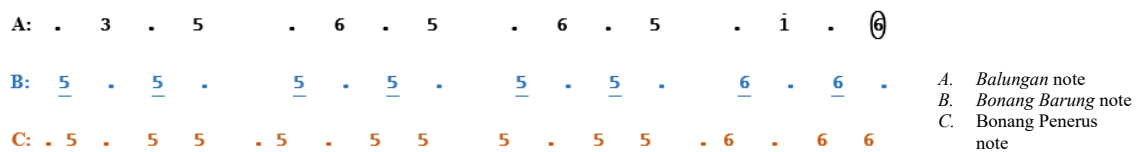


Fig. 11. Part of *Lancaran Ricik Ricik Slendro Manyura* song using *Irama Lancar* and *Gembyang* pattern

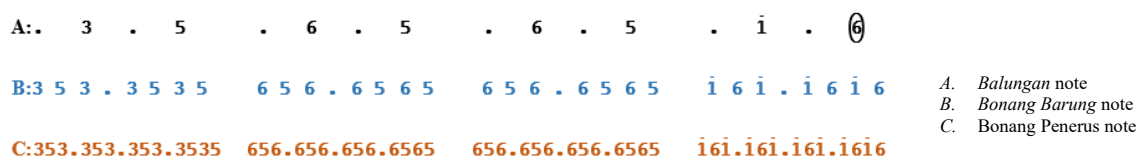


Fig. 12. Part of *Lancaran Ricik Ricik Slendro Manyura* song using *Irama Tanggung* and *Mipil* pattern

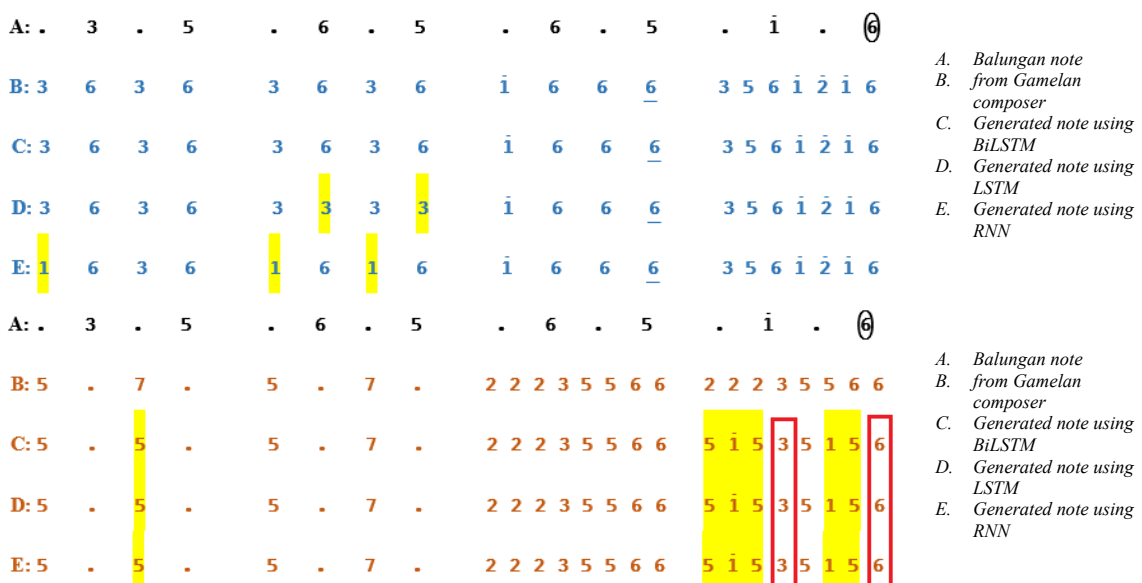


Fig. 13. Part of *Lancaran Ricik Ricik Slendro Manyura* using *Imbal Sekaran* pattern, the yellow sections on parts C, D, and E are the note generator system results that differ from the *gamelan* composer, the red box is the *dbing dbong gamelan* (a) note harmony for Bonang Barung (b) note harmony for Bonang Penerus

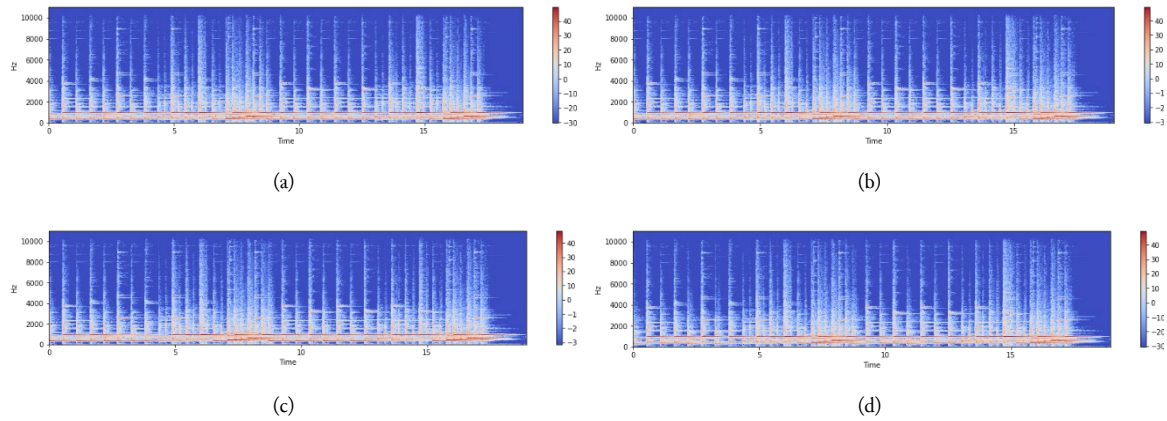


Fig. 14. Spectrogram of the audio data test (a) original (b) generated result using (c) BiLSTM (d) LSTM (e) RNN

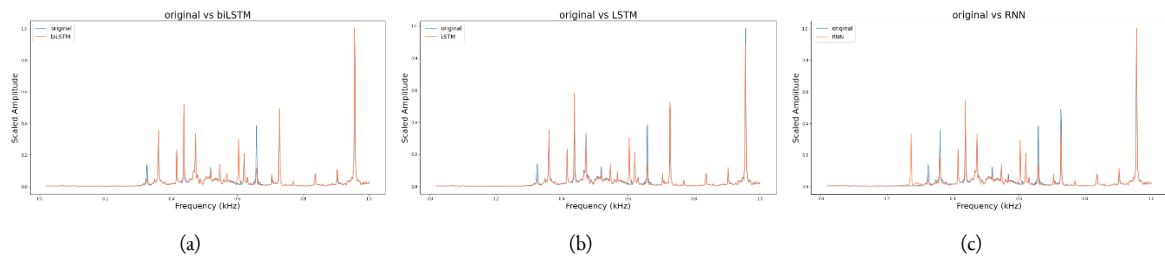


Fig. 15. Visualization of original vs system audio produced by (a) biLSTM (b) LSTM (c) RNN

3.2.3. Audio Visualization

3.2.3.1. Spectrogram

This study uses a spectrogram to extract musical features from audio and measure how similar or different two sounds are. The spectrogram shows the frequency of sound over time, using color to represent signal intensity. Fig. 14 shows the time of the clip on the horizontal axis and the intensity of the original instrument and the generated system on the vertical axis using different colors for the original music from the gamelan composer and the results generated by BiLSTM, LSTM and RNN. The four images show the similarity between the original music and the generator results.

3.2.3.2. Amplitude vs Frequency

The second technique for visualizing audio is to compare the relative amplitudes of the two audios (the original instrument and the generated system) at different frequencies. Fig. 15 shows a plot where the y-axis ranges from 0 to 1. This makes it possible to compare the amplitudes of the original instrument with the generated system, BiLSTM, LSTM, and RNN on the same scale.

From the three plots in Fig. 15, it can be seen that between the original audio (blue) and the generator (red), the results of the BiLSTM generator are more similar to the original shown in Fig. 15a, where the blue plot is less visible compared to the plots of the LSTM and RNN generator results.

3.2.4. Dynamic Time Wrapping (DTW)

DTW is a method for comparing two time series of audio or signals that may have different lengths or distortions [34]. It finds the best alignment or distortion path between the two series, which reduces the distance between corresponding points along the path.

Typically, the DTW is used to calculate the distance between two identical data, with lower values indicating more similarity. In this study, the BiLSTM model has the lowest DTW value of 58198, while

the LSTM and RNN models have DTW values of 77964 and 117480, respectively. These results indicate that the BiLSTM produces music that is closer to the original music than the LSTM and RNN models.

3.2.5. Cross-Correlation

Cross-correlation is used to evaluate the similarity between two audio files by comparing their spectra in the spatial and frequency domains, where a high similarity value indicates similarity with the comparison data [35].

In this study, the analysis of the results shows that the biLSTM model produces audio that is 93.00% similar to the original. Meanwhile, the LSTM model produces audio that is 92.26% similar, and the RNN model produces audio that is 90.09% similar.

4. Conclusion

This study aims to generate harmony notes as accompaniment for Javanese gamelan music using three different recurrent neural networks—BiLSTM, LSTM, and RNN. The models were trained using numerical notation to represent the melody for Balungan and Bonang notation. The use of numerical notation is a simplified way of representing musical data. Because the data used has balanced variations, the three different recurrent neural networks—BiLSTM, LSTM and RNN models—can learn the patterns of musical notes to generate harmonic music. Since the dataset used is only Lancaran type, the song structure is not a feature in this study. Based on the experimental results, it was found that all the models performed better in generating harmonies when using all the features of the song (Balungan notation, rhythm, end of lines and songs, scale, mode and dynamic of the song) than when using only two features (rhythm and Balungan notation). Among the three models, BiLSTM is able to produce harmonies that are more similar to the original music, as shown by the results of note distance analysis, audio visualization, and similarity measurements using DTW and cross-correlation techniques, compared to the LSTM and RNN models. This is because BiLSTM has 2 directions, forward and backward, in learning the training data. However, this study still has limitations in the dataset used, namely having unbalanced variations. Thus, it is necessary to add BiLSTM architecture to increase the output produced, one of which is to use BiLSTM with attention.

Future research can use audio data to produce harmonic music and explore more varied song structures in gamelan music, such as Ketawang, Ladrang, Bubaran, and Ayak-Ayakan, which also have variations in notation for their harmonic patterns. The characteristics of a song greatly affect the success of producing musical accompaniment. Therefore, in future research it is still necessary to study the characteristics of each song according to its song structure, because the diversity of Javanese gamelan styles is not only limited to playing music, but to a more philosophical art. From the results of this study, it appears that the results of the generator system can replace gamelan creators in completing song notation which usually only consists of melodic notation for several instruments. One of them is the harmony notation of the Bonang Barung and Bonang Penerus instruments. Therefore, this study aims to help novice gamelan players learn to play the gamelan, one of which is the harmony music of the Bonang Barung and Bonang Penerus instruments which have various patterns, without the need to understand the rules in Javanese gamelan.

Acknowledgment

This research was funded by the Indonesian Endowment Fund for Education (LPDP) through the BUDI DN Doctoral Scholarship Programme and the Soewidiatmaka Gamelan as gamelan experts in providing data.

Declarations

Author contribution. Arik Kurniawati: conception, methodology, analysis, writing. Eko Mulyanto Yuniarno: Validated, reviewed. Yoyon Kusnendar Suprpto: supervised, reviewed. Aditya Nur Ikhsan Soewidiatmaka: data provision, gamelan theory. All authors read and approved the final version of the manuscript.

Funding statement. This research was supported by LPDP.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

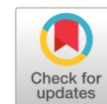
References

- [1] M. Cousineau, S. Carcagno, L. Demany, and D. Pressnitzer, "What is a melody? On the relationship between pitch and brightness of timbre," *Front. Syst. Neurosci.*, vol. 7, no. Jan, p. 127, Jan. 2014, doi: [10.3389/fnsys.2013.00127](https://doi.org/10.3389/fnsys.2013.00127).
- [2] J. D. Lomas and H. Xue, "Harmony in Design: A Synthesis of Literature from Classical Philosophy, the Sciences, Economics, and Design," *She Ji J. Des. Econ. Innov.*, vol. 8, no. 1, pp. 5–64, 2022, doi: [10.1016/j.sheji.2022.01.001](https://doi.org/10.1016/j.sheji.2022.01.001).
- [3] J. Becker, "Traditional Music in Modern Java," *University of Hawaii Press*, p. 253, 2019, doi: [10.2307/j.ctv9zct8](https://doi.org/10.2307/j.ctv9zct8).
- [4] A. Ranjan, V. N. J. Behera, and M. Reza, "Using a Bi-Directional Long Short-Term Memory Model with Attention Mechanism Trained on MIDI Data for Generating Unique Music," in *Studies in Computational Intelligence*, vol. 1006, Springer Science and Business Media Deutschland GmbH, 2022, pp. 219–239, doi: [10.1007/978-3-030-92245-0_10](https://doi.org/10.1007/978-3-030-92245-0_10).
- [5] S. Hastanto, "Konsep *Pathet* Dalam Karawitan Jawa," 2009. [Online]. Available: https://repositori.kemdikbud.go.id/13325/1/Konsep_pathet_dalam_karawitan_jawa.pdf.
- [6] J. Becker, "Earth, Fire, Sakti, and the Javanese Gamelan," *Ethnomusicology*, vol. 32, no. 3, p. 385, 1988, doi: [10.2307/851938](https://doi.org/10.2307/851938).
- [7] A. N. (Andrew N. Weintraub, "Unplayed Melodies: Javanese Gamelan and the Genesis of Music Theory (review)," *Notes*, vol. 63, no. 1, pp. 87–90, 2006, doi: [10.1353/not.2006.0123](https://doi.org/10.1353/not.2006.0123).
- [8] B. Drummond, "Instruments of the Gamelan," *Boston Village Gamelan*, pp. 1–14, 2014. [Online]. Available: <https://www.gamelanbvg.com/gendhing/gamelanGlossary.pdf>.
- [9] J. Hilder, "Central Javanese Gamelan Handbook." p. 51, 1992, [Online]. Available: <https://gamelan.org.nz/wp-content/uploads/2015/02/Jo-Hilder-Central-Javanese-Gamelan.pdf>.
- [10] W. Widodo, B. Susetyo, S. Walton, and W. Appleton, "Implementation of Kupingan Method in Javanese Karawitan Music Training for Foreigners," *Harmon. J. Arts Res. Educ.*, vol. 21, no. 1, pp. 105–114, Jun. 2021, doi: [10.15294/harmonia.v21i1.29993](https://doi.org/10.15294/harmonia.v21i1.29993).
- [11] O. S. Olufunso, A. E. Ewwiekpaefe, and M. E. Irhebhude, "Gender recognition based fingerprints using dynamic horizontal voting ensemble deep learning," *Int. J. Adv. Intell. Informatics*, vol. 8, no. 3, p. 324, Nov. 2022, doi: [10.26555/ijain.v8i3.927](https://doi.org/10.26555/ijain.v8i3.927).
- [12] S. Lathuiliere, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A Comprehensive Analysis of Deep Regression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 9, pp. 2065–2081, Sep. 2020, doi: [10.1109/TPAMI.2019.2910523](https://doi.org/10.1109/TPAMI.2019.2910523).
- [13] B. Kiran, D. Thomas, and R. Parakkal, "An Overview of Deep Learning Based Methods for Unsupervised and Semi-Supervised Anomaly Detection in Videos," *J. Imaging*, vol. 4, no. 2, p. 36, Feb. 2018, doi: [10.3390/jimaging4020036](https://doi.org/10.3390/jimaging4020036).
- [14] M. Bakator and D. Radosav, "Deep Learning and Medical Diagnosis: A Review of Literature," *Multimodal Technol. Interact.*, vol. 2, no. 3, p. 47, Aug. 2018, doi: [10.3390/mti2030047](https://doi.org/10.3390/mti2030047).
- [15] H. Li, "Deep learning for natural language processing: advantages and challenges," *Natl. Sci. Rev.*, vol. 5, no. 1, pp. 24–26, Jan. 2018, doi: [10.1093/nsr/nwx110](https://doi.org/10.1093/nsr/nwx110).
- [16] T. Yampaka, S. Vonganansup, and P. Labcharoenwongs, "Feature selection using regression mutual information deep convolution neuron networks for COVID-19 X-ray image classification," *Int. J. Adv. Intell. Informatics*, vol. 8, no. 2, p. 199, Jul. 2022, doi: [10.26555/ijain.v8i2.809](https://doi.org/10.26555/ijain.v8i2.809).

- [17] S. H. Ing, A. A. Abdullah, M. Y. Mashor, Z.-A. Mohamed-Hussein, Z. Mohamed, and W. C. Ang, "Exploration of hybrid deep learning algorithms for covid-19 mrna vaccine degradation prediction system," *Int. J. Adv. Intell. Informatics*, vol. 8, no. 3, p. 404, Nov. 2022, doi: [10.26555/ijain.v8i3.950](https://doi.org/10.26555/ijain.v8i3.950).
- [18] S. Bhardwaj, S. M. Salim, D. Talha Ali Khan, and S. JavadiMasoudian, "Automated Music Generation using Deep Learning," in *2022 International Conference Automatics and Informatics (ICAI)*, Oct. 2022, pp. 193–198, doi: [10.1109/ICAI55857.2022.9960063](https://doi.org/10.1109/ICAI55857.2022.9960063).
- [19] O. Peracha, "Improving Polyphonic Music Models with Feature-Rich Encoding," in *Proceedings of the 21st International Society for Music Information Retrieval Conference, ISMIR 2020*, Nov. 2019, p. 7. [Online]. Available: <https://arxiv.org/abs/1911.11775>.
- [20] J. Ens and P. Pasquier, "MMM : Exploring Conditional Multi-Track Music Generation with the Transformer," *arxiv Comput. Sci.*, p. 10, Aug. 2020, Accessed: May 27, 2023. [Online]. Available: <https://arxiv.org/abs/2008.06048v2>.
- [21] A. Kurniawati, Y. K. Suprpto, and E. M. Yuniarno, "Multilayer Perceptron for Symbolic Indonesian Music Generation," in *2020 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Jul. 2020, pp. 228–233, doi: [10.1109/ISITIA49792.2020.9163723](https://doi.org/10.1109/ISITIA49792.2020.9163723).
- [22] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, "Deep Learning Techniques for Music Generation," *Cham: Springer International Publishing*, p. 284, 2020, doi: [10.1007/978-3-319-70163-9](https://doi.org/10.1007/978-3-319-70163-9).
- [23] M. Civit, J. Civit-Masot, F. Cuadrado, and M. J. Escalona, "A systematic review of artificial intelligence-based music generation: Scope, applications, and future trends," *Expert Syst. Appl.*, vol. 209, p. 118190, Dec. 2022, doi: [10.1016/j.eswa.2022.118190](https://doi.org/10.1016/j.eswa.2022.118190).
- [24] J. Xie, "A Novel Method of Music Generation Based on Three Different Recurrent Neural Networks," *J. Phys. Conf. Ser.*, vol. 1549, no. 4, p. 042034, Jun. 2020, doi: [10.1088/1742-6596/1549/4/042034](https://doi.org/10.1088/1742-6596/1549/4/042034).
- [25] N. Agarwala, Y. Inoue, and A. Sly, "Music composition using neural network," p. 10, 1992. [Online]. Available: <https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2762076.pdf>.
- [26] S. Mangal, R. Modak, and P. Joshi, "LSTM Based Music Generation System," *IARJSET*, vol. 6, no. 5, pp. 47–54, May 2019, doi: [10.17148/IARJSET.2019.6508](https://doi.org/10.17148/IARJSET.2019.6508).
- [27] F. Shah, T. Naik, and N. Vyas, "LSTM Based Music Generation," in *2019 International Conference on Machine Learning and Data Engineering (iCMLDE)*, Dec. 2019, pp. 48–53, doi: [10.1109/iCMLDE49015.2019.00020](https://doi.org/10.1109/iCMLDE49015.2019.00020).
- [28] S. Agarwal, V. Saxena, V. Singal, and S. Aggarwal, "LSTM based Music Generation with Dataset Preprocessing and Reconstruction Techniques," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, Nov. 2018, pp. 455–462, doi: [10.1109/SSCI.2018.8628712](https://doi.org/10.1109/SSCI.2018.8628712).
- [29] S. Chikkamath and N. S R, "Melody generation using LSTM and BI-LSTM Network," in *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)*, Nov. 2021, pp. 1–6, doi: [10.1109/ICCICA52458.2021.9697286](https://doi.org/10.1109/ICCICA52458.2021.9697286).
- [30] S. S. Azmi, C. S. Shreevara, and S. Baliga, "Music generation using Bidirectional Recurrent Neural Nets," *Int. Res. J. Eng. Technol.*, vol. 7, no. May, pp. 6863–6866, 2020, [Online]. Available: <https://www.academia.edu/download/64615427/IRJET-V7I51292.pdf>.
- [31] T. Jiang, Q. Xiao, and X. Yin, "Music Generation Using Bidirectional Recurrent Network," in *2019 IEEE 2nd International Conference on Electronics Technology (ICET)*, May 2019, pp. 564–569, doi: [10.1109/ELTECH.2019.8839399](https://doi.org/10.1109/ELTECH.2019.8839399).
- [32] H. Elfaik and E. H. Nfaoui, "Deep Bidirectional LSTM Network Learning-Based Sentiment Analysis for Arabic Text," *J. Intell. Syst.*, vol. 30, no. 1, pp. 395–412, Dec. 2020, doi: [10.1515/jisys-2020-0021](https://doi.org/10.1515/jisys-2020-0021).
- [33] N. M R and S. Mohan B S, "Music Genre Classification using Spectrograms," in *2020 International Conference on Power, Instrumentation, Control and Computing (PICC)*, Dec. 2020, pp. 1–5, doi: [10.1109/PICC51425.2020.9362364](https://doi.org/10.1109/PICC51425.2020.9362364).
- [34] L. Lerato and T. Niesler, "Feature trajectory dynamic time warping for clustering of speech segments," *EURASIP J. Audio, Speech, Music Process.*, vol. 2019, no. 1, p. 6, Dec. 2019, doi: [10.1186/s13636-019-0149-9](https://doi.org/10.1186/s13636-019-0149-9).

-
- [35] C. Jiang, D. Yang, and X. Chen, "Similarity Learning For Cover Song Identification Using Cross-Similarity Matrices of Multi-Level Deep Sequences," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, vol. 2020-May, pp. 26–30, doi: [10.1109/ICASSP40776.2020.9053257](https://doi.org/10.1109/ICASSP40776.2020.9053257).

Understanding requirements dependency in requirements prioritization: a systematic literature review



Fiftin Noviyanto ^{a,b,1,*}, Rozilawati Razali ^{a,2}, Mohd Zakree Ahmad Nazri ^{c,3}

^a Center for Software Technology and Management, FTSM, Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia

^b Informatics Department, Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta, Indonesia

^c Center for Artificial Intelligence and Technology, FTSM, Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia

¹ fiftin.noviyanto@tif.uad.ac.id; ² rozilawati@ukm.edu.my; ³ zakree@ukm.edu.my

* corresponding author

ARTICLE INFO

Article history

Received February 24, 2023

Revised May 7, 2023

Accepted May 16, 2023

Available online June 1, 2023

Keywords

Requirement prioritization

Requirements dependencies

Systematic literature review

Machine learning

Optimization technique

ABSTRACT

Requirement prioritization (RP) is a crucial task in managing requirements as it determines the order of implementation and, thus, the delivery of a software system. Improper RP may cause software project failures due to over budget and schedule as well as a low-quality product. Several factors influence RP. One of which is requirements dependency. Handling inappropriate handling of requirements dependencies can lead to software development failures. If a requirement that serves as a prerequisite for other requirements is given low priority, it affects the overall project completion time. Despite its importance, little is known about requirements dependency in RP, particularly its impacts, types, and techniques. This study, therefore, aims to understand the phenomenon by analyzing the existing literature. It addresses three objectives, namely, to investigate the impacts of requirements dependency on RP, to identify different types of requirements dependency, and to discover the techniques used for requirements dependency problems in RP. To fulfill the objectives, this study adopts the Systematic Literature Review (SLR) method. Applying the SLR protocol, this study selected forty primary articles, which comprise 58% journal papers, 32% conference proceedings, and 10% book sections. The results of data synthesis indicate that requirements dependency has significant impacts on RP, and there are a number of requirements dependency types as well as techniques for addressing requirements dependency problems in RP. This research discovered various techniques employed, including the use of Graphs for RD visualization, Machine Learning for handling large-scale RP, decision making for multi-criteria handling, and optimization techniques utilizing evolutionary algorithms. The study also reveals that the existing techniques have encountered serious limitations in terms of scalability, time consumption, interdependencies of requirements, and limited types of requirement dependencies.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

A large-scale software system development normally involves vast amounts of requirements, which contribute significantly to the success of the system. Software projects, on the other hand, do have resource constraints that impede them from realizing all the requirements at once. In essence, implementing massive requirements with limited resources due to budget, schedule [1], and staff constraints is troublesome [2]. One possible management strategy to resolve the issue is through

prioritization. Requirements prioritization (RP) is an essential part of requirements management, aimed at selecting the requirements based on certain predetermined criteria so that they could be implemented in stages [3]–[5]. Several common criteria that determine RP include stakeholders [6], system functionality [7], cost [8], processing time [8], risk considerations [9], [10], and business values [4].

RP activities always involve two parties, namely developers and stakeholders. The two parties have different focuses and thus, priorities [11]. Stakeholders focus on urgency, needs, and business values [12], [13]. Although developers are concerned about project attributes such as effort [12] and cost [14], [13], they are also aware of internal constraints such as dependency between functions or requirements [12]. This is due to the fact that requirements dependency is commonly found on the project software [2], [15]. It is thus risky to conduct RP without considering the dependency between requirements [16], [17]. For instance, giving high priority to requirements that depend on other requirements can increase the waiting time and delay the project [18]. This is because the dependent requirements have to wait for the prerequisite requirements to be completed before they could be implemented. In addition, requirements dependency also implies product complexity [2] and project risk [19]. The higher the dependency, the higher the complexity of the system and thus the higher its risk of failure is [20].

Several studies have investigated RP concerning the criteria and techniques used in the process, such as Hujainah et al. [21], Tan and Mohamed [22], Falak Sher et al. [23], Muhammad Sufian et al. [24], Pitangueira et al. [25], Achimugu et al. [16], and Al Ta'ani and Razali [26]. However, none of the studies examine requirements dependency in depth. In fact, only 4 out of the 65 RP techniques consider requirements dependency [2]. Many studies on RP do not include requirements dependencies as one of the factors influencing priority sequencing. Little is known about requirements dependency in RP. Therefore, this study aims to explore further requirements dependency in RP by conducting a systematic literature review (SLR) in order to improve the understanding of the phenomenon. The objectives are:

- to investigate the impacts of requirements dependency on RP
- to identify different types of requirements dependency
- to discover the techniques used for requirements dependency problems in RP

The structure of the paper is as follows. Section 2 describes the methodology used in the review. Section 3 discusses the threats to validity. Section 4 presents the results and discussions. Section 5 concludes the study.

2. Method

This study adopts the Systematic Literature Review (SLR) method proposed by Kitchenham et al. [27]. Fig. 1 illustrates the review protocol used, which comprises five stages: identification, search strategy, study selection strategy, data retrieval, and result.

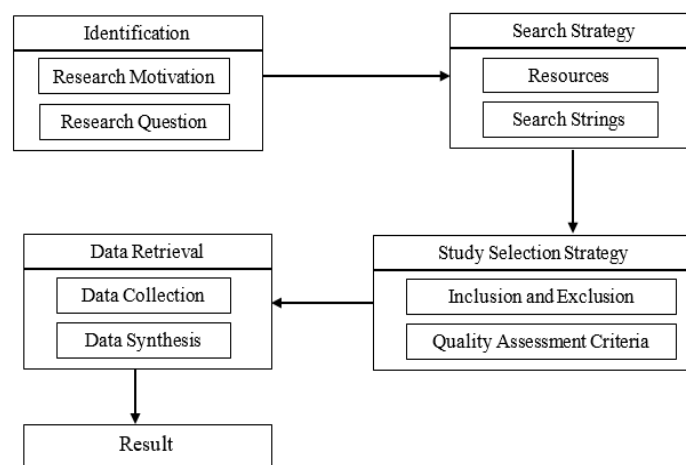


Fig. 1. Review Protocol

In the first stage, the research questions were constructed and aligned with the research motivation and research question. The second stage determined the resources and the search strings based on the research questions. The third stage outlined the inclusion and exclusion criteria for screening the gathered articles together with the Quality Assessment Criteria (QAC) process. The fourth stage finalized the selection of the data collection through which data synthesis was made. In the final step, the results of the synthesis were obtained, which are presented in this paper.

2.1. Research Questions

The study aims to understand the relationships between requirements dependency and RP. Therefore, the following research questions (RQ) were constructed:

- RQ1: Does requirements dependency have impacts on RP?
- RQ2: What are the different types of requirements dependency?
- RQ3: What are the existing techniques used for requirements dependency problems in RP?

2.2. Search Strategy

The search strategy undertaken in this study began with determining the sources of scientific literature. Seven sources were used in the literature search, as listed in Table 1.

Table 1. Selected Literature Database Resources

Resource Name	Resource Link
SpringerLink	http://www.springerlink.com
Google Scholar	https://scholar.google.com
ISI Web of Knowledge	http://www.isiknowledge.com
Elsevier	http://www.elsevier.com
ScienceDirect	http://www.sciencedirect.com/
IEEE Xplore	http://www.ieee.org/web/publications/xplore/
ACM Digital Library	http://portal.acm.org

- Resources: The seven sources were selected because their contents are relevant to the subjects of this study, besides being referred by researchers in the field.
- Search Strings: Specifically, the keywords used for searching research articles in this study were 'requirement prioritization' or 'dependencies'. The search keywords for review papers were 'requirement prioritization' (AND/OR) 'literature review'.

2.3. Study Selection Criteria

The searches in the seven sources using the predefined keywords found 432 articles. These articles were firstly screened in terms of suitability based on their titles and/or abstracts. As a result, only 133 articles were selected. Fig. 2 shows the distribution of the articles. Most articles are journal and conference papers.

To ensure only the most relevant articles would be selected, the 133 articles were further vetted through several subsequent stages, as shown in Fig. 3. Inclusion and Exclusion Criteria: The inclusion criteria used to select the articles are as follows: 1) Articles are written in English; 2) Articles focus on requirements dependency in RP domain; and 3) Articles are able to answer at least one of the research questions. The exclusion criteria include: 1) Articles are not written in English; 2) Duplicate articles - excluding multiple copies of the same study; and 3) Articles are not answering any of the research questions. Each collected article was briefly read through its title, abstract, and content. Studies that did not address the research question were excluded. Similarly, studies that were still in the research process or not published by a publisher were not included. This study aims to gather findings that have been proven empirically. Therefore, review articles were excluded from the selection. If the same article was

found from different sources, only one would be chosen. The articles were published within the period of 2012 to 2022.

DISTRIBUTION OF COLLECTED ARTICLES

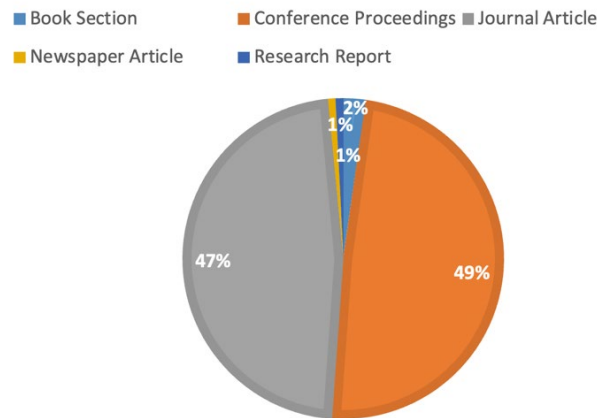


Fig. 2. Percentage of collected articles based on titles and/or abstracts

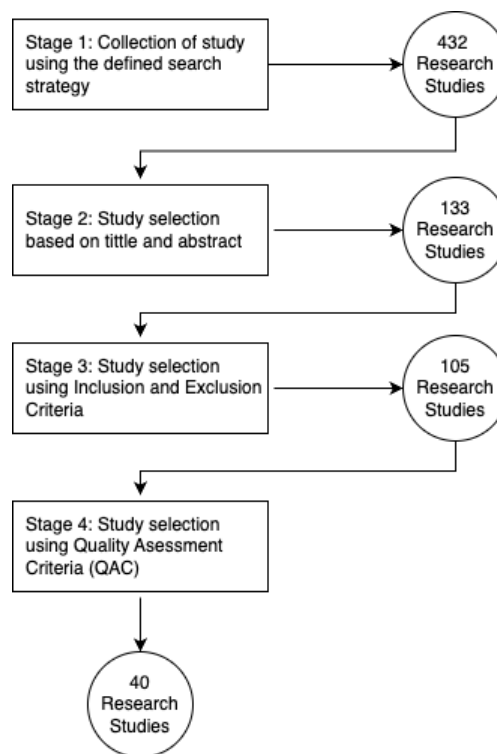


Fig. 3. Percentage of collected articles based on titles and/or abstracts

Quality Assessment Criteria: Quality Assessment Criteria (QAC) was used to measure the quality of the gathered articles with respect to the objectives of the study. First, the articles shall cover requirements dependency and RP. Second, the articles shall be trustworthy. Eight questions were derived to represent the criteria, as listed in Table 2. The possible score for each question was divided into three: Yes (1), Partially (0.5) and No (0). The weighted score for each study was the sum of scores for the eight questions. The assessments were conducted by the authors, through which the scores were consensually determined.

Table 2. Question for Quality Assessment Criteria

ID	Question (Q)	Answer Score
Q1	Is the objective of the research related to requirements prioritization clearly stated?	Yes = 1/ partially= 0.5/no = 0
Q2	Does the study focus on the requirement prioritisation?	Yes = 1/ partially= 0.5/no = 0
Q3	Does the study focus on the dependencies on RP?	Yes = 1/ partially= 0.5/no = 0
Q4	Does the study illustrate the current various/complexity/types of requirement dependencies on RP?	Yes = 1/ partially= 0.5/no = 0
Q5	Does the study explain the proposed techniques to handle requirements dependencies in RP?	Yes = 1/ partially= 0.5/no = 0
Q6	Are the measures used in the study the most relevant ones for answering the research questions (this study)?	Yes = 1/ partially= 0.5/no = 0
Q7	Does the research contribute to requirements prioritization considering requirements dependencies?	Yes = 1/ partially= 0.5/no = 0
Q8	Is the result of the study clearly stated?	Yes = 1/ partially= 0.5/no = 0

After the assessment, only forty articles were selected based on the weighted score > 4.5. A score of 4.5 was used as the baseline as it designates that the article has achieved more than 56% of the best score (4.5 out of 8). Table 3 presents the weighted scores for the forty selected articles. The highest score is 8 (five articles) and the lowest score is 5 (five article), whereas the median score is 6 (twelve articles). This implies that the selected articles are well-documented, and thus contain well-conducted studies. This claim is particularly demonstrated by the scores attained for Q1, Q7 and Q8, which are mainly 1 (Yes). The selected articles are however moderately covering the focus of this study, as the scores for Q2 to Q6 are mostly 0.5 (Partly). This indicates that the emphasis of the current available studies on requirements dependency and RP are still lacking, albeit relevant. Only five articles (Score = 8) fulfill the quality criteria of the study entirely.

Table 3. Quality Assessment criteria results

Reference of the selected study	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Score
[7]	1	1	0.5	0.5	0.5	0.5	1	1	6
[18]	1	1	1	1	1	1	1	1	8
[28]	1	1	1	1	1	1	1	1	8
[29]	1	1	1	0.5	1	1	1	1	7.5
[30]	1	1	0.5	0.5	0.5	0.5	1	1	6
[31]	0.5	1	1	0.5	1	1	1	0.5	6.5
[32]	1	1	0.5	0	0.5	0.5	1	1	5.5
[12]	1	1	0.5	1	0.5	0.5	1	1	6.5
[33]	1	1	0.5	0.5	0.5	1	1	1	6.5
[34]	1	0.5	0.5	0.5	0.5	0.5	1	1	5.5
[35]	1	1	1	0.5	1	1	1	1	7.5
[36]	0.5	1	0.5	0.5	0.5	0.5	1	1	5.5
[26]	1	1	0.5	0.5	0.5	0.5	1	1	6
[37]	1	1	0.5	0.5	0.5	0.5	1	1	6
[38]	1	1	0.5	0.5	0.5	0.5	1	1	6
[39]	1	0.5	0.5	0.5	0.5	0.5	1	1	5.5
[40]	1	1	1	1	1	0.5	0.5	0.5	6.5
[41]	1	1	0.5	0.5	0.5	0.5	1	1	6
[42]	1	0.5	0.5	0.5	1	0.5	1	1	6
[43]	1	1	1	0.5	1	1	1	1	7.5
[44]	1	1	1	0.5	0.5	0.5	1	1	6.5
[45]	1	1	1	1	1	0.5	1	1	7.5
[46]	1	0.5	1	0.5	0.5	0.5	1	1	6
[47]	1	1	0.5	0.5	0.5	0.5	1	1	6
[48]	1	1	1	1	1	1	1	0.5	7.5
[49]	1	1	1	1	1	1	1	1	8
[50]	1	1	0.5	0.5	0.5	0.5	1	1	6
[51]	1	1	0.5	0.5	1	1	1	1	7
[52]	1	1	0.5	0.5	0.5	0.5	1	1	6

Reference of the selected study	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Score
[53]	1	0.5	0.5	0.5	0	0.5	1	1	5
[54]	1	0,5	0,5	0	0	1	1	1	5
[55]	1	0,5	1	1	1	1	1	1	7,5
[56]	1	1	0	0	1	1	1	1	6
[57]	1	1	1	1	1	1	1	1	8
[58]	1	1	1	1	1	1	1	1	8
[59]	1	1	0	0	0	1	1	1	5
[60]	1	0,5	1	1	1	1	1	1	7,5
[61]	1	1	0	0	0	1	1	1	5
[62]	1	1	0	0	0	1	1	1	5
[63]	1	0,5	1	1	1	1	1	1	7,5

2.4. Data Retrieval

The data retrieval consists of two activities, namely data collection and data synthesis. Data collection is the process of bringing together the selected articles, whereas data synthesis is a purposeful activity that extracts facts from the selected studies for answering the stated research questions [27].

- **Data Collection:** This stage gathered and consolidated the selected thirty articles. The articles were then classified into three groups based on the three RQs: RQ1, RQ2 and RQ3. For example, the articles that discuss the impact of requirements dependency on RP were placed under RQ1 group. Same goes to the articles that belong to RQ2 and RQ3. The articles that address more than one RQ were placed accordingly into the respective RQ groups.
- **Data Synthesis:** This stage extracted facts from the grouped articles in order to find the answers for the research questions. The facts were then analyzed and visualized. For example, the findings for RQ1 are presented as a chart that shows the frequency distribution of articles across RP factors. Similarly, the requirements dependency types for RQ2 are illustrated as a taxonomy graph, whereas the techniques for RQ3 are demonstrated as chart and table. The visualization helps in explaining the results, thus providing a better understanding of the phenomenon.

3. Threats of validity

The main challenge in SLR is the validity of the study, which includes the completeness, publication bias and data synthesis [27]. This study adopted the review protocol to overcome the completeness threat. The searches were conducted on various databases and the articles were screened using the predetermined quality criteria. Nevertheless, the searches were limited to publications from year 2012-2022 and articles in other languages were excluded. The consideration for choosing English is due to its status as an international language widely used in reputable journals. To avoid publication bias, only articles that contain empirically proven data were considered. Therefore, gray studies that are still in progress were not included. The consideration to exclude gray literature is the ease of literature search for future researchers. To mitigate the data synthesis threat, QAC process was conducted. QAC identified and filtered reliable studies that could answer the research questions. Moreover, manual checks were carried out on the extracted facts repetitively. The assessments were carried out objectively and consensually by the authors to avoid inconsistencies. The authors read the entire collected papers and provided scores based on the QAC questions.

4. Results and Discussion

This section describes the results of the analysis based on the forty selected articles.

4.1. Overview of Selected Primary Research Studies

Fig. 2 shows that the most selected articles are journal and conference papers (96%), whereas the rest are book sections, newspaper articles and reports (4%). After QAC process, the distribution changes

slightly as shown in Fig. 4. Most articles still constitute journal and conference papers (92.5%), while the rest are book sections only (7.5%). As newspaper articles and reports generally lack scientific evidence and arguments, they could not be selected in this round.

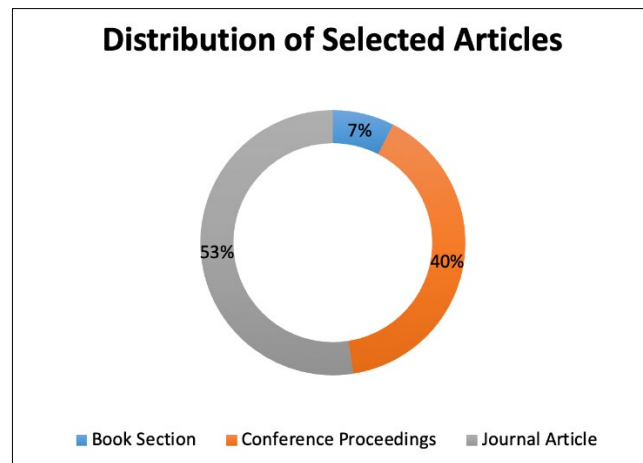


Fig. 4. Percentage of selected articles

Fig. 5 displays the distribution of the selected articles within the period of 2012 until 2022. The chart shows the topics which are consistently studied every year for the last eleven years, with at least three articles per year (median). The number is not high, this indicates that requirements dependency and RP are two topics investigated by the research community in recent years. Since it is not as many as other topics in requirements engineering field, this may suggest that more studies are required to investigate the topics.

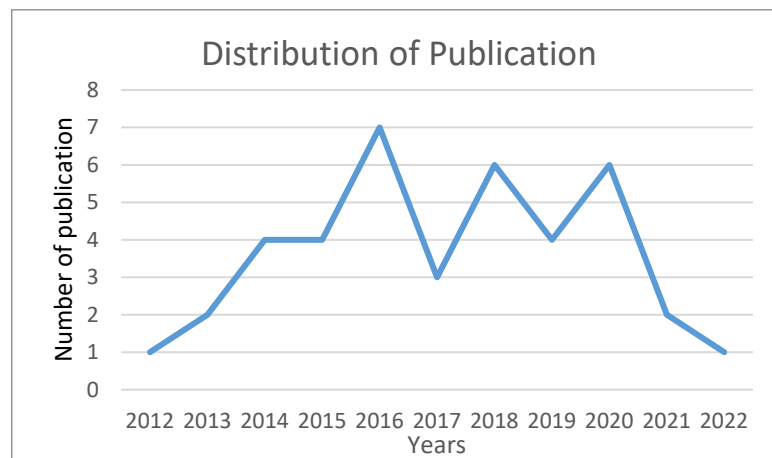


Fig. 5. Number of selected articles by publication year

4.2. Does requirements dependency have impacts on RP (RQ1)?

Fig. 6 indicates that requirements dependency is a critical factor that is of concern to many studies in regards to RP. The articles emphasize that requirements dependency becomes more challenging, particularly for large-scale systems [28], [35], [48]. Improper handling of requirements dependency may cause inefficiency [29], project delays [54], redesign and rework [52], as dependencies among requirements are commonly found in software projects [64]. If a requirement that becomes a prerequisite to other requirements is given a low priority, it affects the completion time of the whole project [18], [40]. The prerequisite requirements, therefore, need to be given a higher priority. This implies that requirements dependency determine the complexity of relationships between requirements and thus contributes to erroneous or redundant results [30] and also implies a higher requirement implementation risk [44], [55]. In the requirement prioritization process, there are two different perspectives from the stakeholders and developers. On the client-side, priorities depend on urgency, needs, and business value.

On the developer side, the priority is influenced by something more technical in the system development process, which is the requirements dependencies. [29], [36]. The results of the qualitative research conducted by Al Ta’ani [26] obtained the same result, indicating that analysts and system developers considered dependency as an important factor in requirement prioritization.

Fig. 6 shows that cost and risk are also relatively significant in RP. In general, cost and risk are implicitly influenced by the complexity of requirements, among others. The higher the complexity, the higher the cost and the risk of implementing the requirements are [20]. As discussed earlier, requirements dependency causes requirements complexity [2]. This fact indirectly highlights further the impacts of requirements dependency on RP.

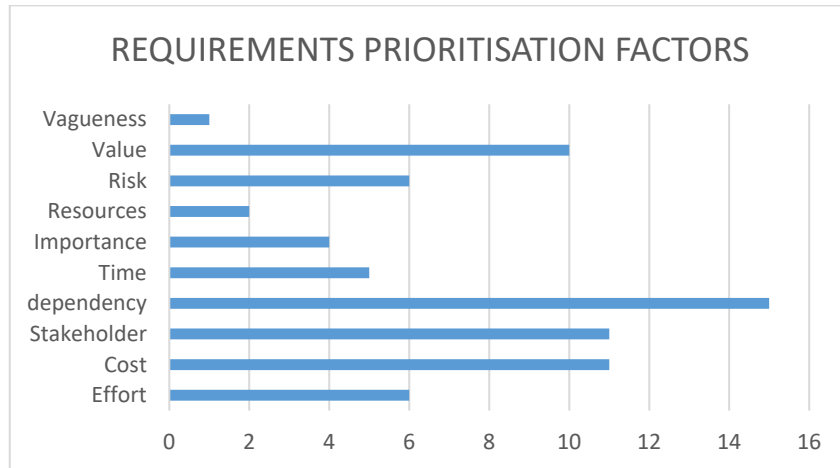


Fig. 6. Frequency of requirements prioritisation factors

4.3. What are the different types of requirements dependency?(RQ2)

RQ2 focuses on extracting the types of requirements dependency. In general, there are two main classifications of requirements dependency proposed by [65] and [66]. As illustrated in Table 4, the former classifies dependency into three groups [65]: Functional; Value-related, and Time-related. The Functional consists of Combination, Implication, and Exclusion. Combination refers to the requirements to be implemented together and Implication is the requirements that must wait for other requirements to complete. Exclusion is the opposite of Combination, comprising the requirements that cannot be applied together as they are conflicting with each other. On the other hand, Value-related consists of Revenue-based and Cost-based; Revenue-based are requirements that can affect income, whereas Cost-based are requirements that can affect costs. The last group is Time-related, which is requirements that need to be implemented based on the time stated in the project schedule.

Table 4. Dependency Classification Based On C. Li

Dependency group	Dependency Type
Functional dependency	Combination
	Implication
	Exclusion
Value-related dependency	Revenue-based
	Cost-based
Time-related dependency	Time-related

The latter classifies dependency into three types, as shown in Table 5, namely Structural Interdependencies, Constrain Interdependencies, and Cost/Value Interdependencies [66]. Structural Interdependencies consist of four: Refined to; Change to; Similar to; and Requires. Constrain Dependencies consist of Requires and Conflicts with. The Requires are included in both Structural and

Constrain Interdependencies. Another classification is Cost/Value Interdependencies, which comprise Increase/Decrease Cost of and Increase/Decrease Value of.

Table 5. Dependency Classification Based On Dahlstedt’s Model

Dependency group	Dependency Type
Structural Interdependencies	Refined_to
	Change_to
	Similar_to
Constrain Interdependencies	Requires
	Cost-based
	Conflicts_with
Cost/Value Interdependencies	Increases/ Decreases_ cost_of
	Increases/ Decreases_ value_of
Structural Interdependencies	Refined_to

In addition to the above classifications, there are also articles that mention indirectly and solely other types of requirements dependency. The articles mostly use different terms, even though they refer to the same kinds of dependency. In order to avoid redundancy and inconsistency, this study joins similar types together and assigns coherent terms that represent the classifications best. Fig. 7 illustrates the taxonomy view of requirements dependency categories synthesized from various classifications proposed in the selected articles.

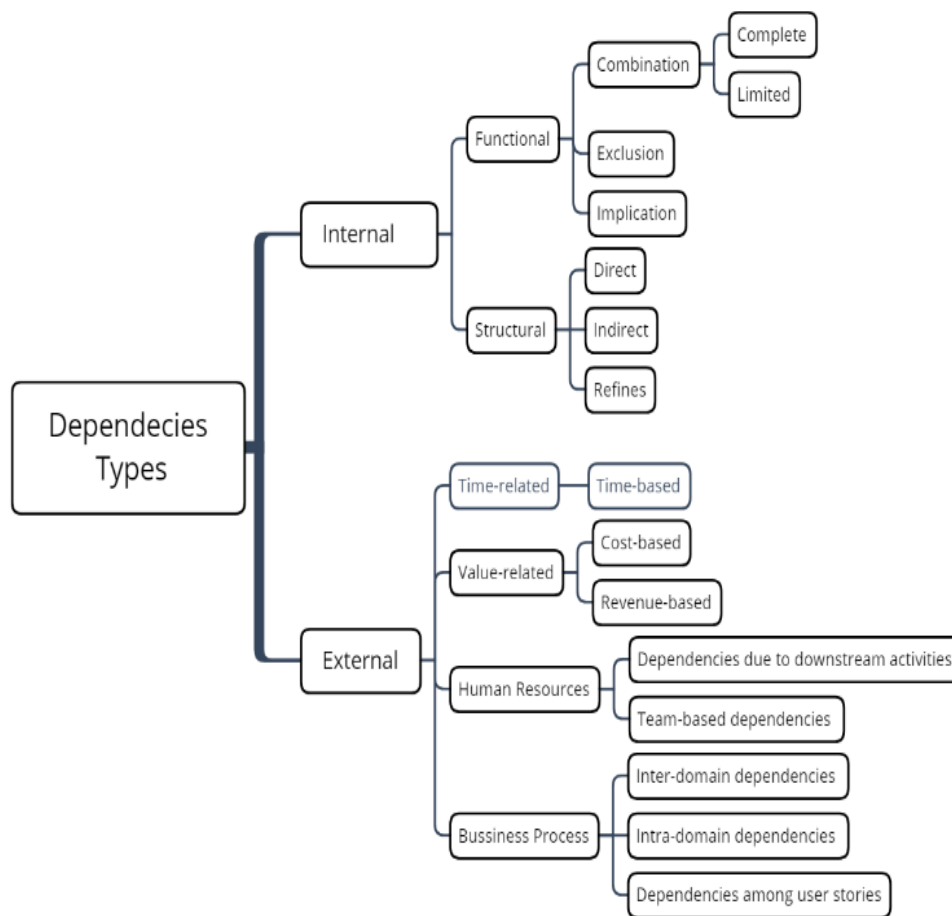


Fig. 7. Types of requirements dependency

Overall, there are two requirements dependency categories: Internal and External. Internal dependency means interior attributes of the system that cause its requirements interdependencies. External dependency means exterior attributes that affect or influence the requirements of the system. Internal dependency has two subcategories, namely, Functional and Structural. External dependency is divided into four subcategories which consist of Time-related, Value-related, Human Resource, and Business Process.

For the Functional subcategory in Internal category, there are three types of dependency including:

- *Combination* [42], [65], [66] is a pair of requirements that must be applied together. Other similar terms used are Coupling [7], Concurrency [35], Requires [54], [55], and Constrain [18], [49]. This type has two subtypes, namely Complete [29], and Limited [29]. Complete is dependent on another requirement completely while Limited is partly dependent on another requirement.
- *Exclusion* [42] is a pair of conflicting requirements, which cannot be applied together. Other similar terms used are Conflicts [35], Contradict [43].
- *Implication* [42] is a requirement that requires other requirements to function. Other terms used are Precedence [7], [51], [58], Time-related [42], [65] and Support [43].

Likewise, there are three types for the Structural [66] subcategory, namely:

- *Direct* [29], [41] means that requirements depend directly on other requirements. For example, X depends on Y directly.
- *Indirect* [29] means that requirements depend on other requirements indirectly. For example, X depends on Z, while Z depends on Y. This shows X depends on Y but through Z.
- *Refines* [35], [67] means that requirements of higher levels are explained by a number of requirements of lower levels. Another term used for this type is hierarchy [31].

On the other hand, the External category consists of Time-related, Value-related, Human resources, and Business processes. There is only one type of Time-related sub-category, namely *Time-based* [42], [65]. This means a requirement that needs to be implemented based on the time stated in the project schedule. Meanwhile, in the Value-related subcategory, the two types comprise:

- *Cost-based* [28], [42], [65], [58], [57], means a requirement that can affect cost. Other terms found are Contribution [35] and Cost-related [58].
- *Revenue-based* [42], [47], [65] means a requirement that can affect income.

There are two types of Human resource subcategory, namely:

- Dependencies due to Downstream Activities [52] imply requirements whose implementation considers optimizing existing human resources.
- Team-based Dependencies [52] concern about avoiding multiple teams having to work on the same or on dependent requirements.
- The last sub-category of External category is the business process, which has three types as follows:
- Inter-domain Dependencies [52] indicate requirements whose implementation depends on requirements across business sectors.
- Intra-domain Dependencies [52] indicate requirements whose implementation depends on certain business processes.
- Dependencies among user stories [52] indicate dependencies between non-functional requirements (e.g. usability, maintainability) and architecture choices.

4.4. What are the existing techniques used for requirements dependency problems in RP? (RQ3)

There are various techniques proposed in the selected articles for solving requirements dependency problems in RP. All the techniques used in the selected studies (based on QAC) were analyzed, clustered, and studied in their process. In general, the discovered techniques have specific problem criteria. Decision Making is used to address RP with multiple criteria: Evolutionary Algorithm for computational optimization, Fuzzy logic to handle uncertainty factors, NLP for automated identification of RP and RD based on human language, Machine Learning for automatic determination of RP based on datasets, and Graph-based approaches for mapping RD within groups of requirements.

The most commonly used techniques found in the selected articles are Decision Making, including Collaborative requirement prioritization method [12], Utility-based prioritization [68], Majority Voting Goal-Based (MVGB) [37], Analytic Hierarchy Process (AHP) [18], [50] and Hierarchical Dependencies [31]. One of the Multi-Criteria-Decision-Making techniques is AHP. AHP has excellent accuracy since pairwise comparison is able to provide decisions that are accurate and worth considering [69]. However, pairwise comparison is time-consuming for large scale projects [37].

The second-highest technique is Evolutionary Algorithm (EA), which comprises the Least-Squares-Based Random Genetic Algorithm [30], Hybrid Enriched Genetic Revamped Integer Linear Programming [42], Multi-objective Evolutionary Algorithms (MOEAs) [15], MOSAs [58], Interactive Genetic Algorithm (IGA) [51] and Early Mutation Testing [32]. The most widely used EA technique is the Genetic Algorithm (GA). This technique aims to reduce computation time. It can be combined with other techniques that are able to provide better accuracy. In the selected articles, EA is only used in simulation cases. Thus, it needs to be proven in industry settings.

The next category is Fuzzy Logic. There are three techniques in this category, namely the Hierarchical Fuzzy Inference System (HFIS) [7], Fuzzy Inference System (FIS) [45], Fuzzy Clustering [62], Rough Set Theory [63], and Tensor and Fuzzy Graphs [28]. Fuzzy is used to help in the decision-making process. Each stakeholder's perception of the value of a requirement is different, which is mainly based on interests and knowledge. Fuzzy Logic can be used to solve uncertainty problems due to human judgment.

Previous studies also use Neuro-Linguistic Programming (NLP) for requirements dependency in RP, such as Satisfiability Modulo Theories (SMT) [48] and SNIPR [70]. SNIPR completes SMT. NLP is used as the input for both techniques. Requirements are clustered using NLP and combined with weighting dependencies. The ranking process is combined with GA [48] and AHP [70]. NLP is quite helpful in filtering requirements, thus minimizing redundancy and similarity. Nevertheless, NLP still needs to be explored more in detecting dependencies between requirements, so that costs and time can be further optimized, especially for large scale projects.

Other existing techniques are Graph and Matrix. They are used to visualize and calculate relation weights. The Matrix can be applied separately [34], [71], [44], [38] or in conjunction with Graph [43], [57]. Graphs are composed of nodes, which represent requirements, and edges as relations. Matrix, on the other hand, consists of rows and columns, with cells showing relations between requirements. Because of the visual representation, both techniques make it easy to view dependencies among requirements. However, the techniques would consume time and cost for large-scale requirements.

Machine learning (ML) has been introduced to automate the process of RP. There are five ML methods for requirements dependency in RP, namely CDBR [29], DRank [35], Active Learning [60], Supervised Classification Technique [55], and Interactive Next Release Problem (iNRP) [39]. First, CDBR exploits the Particle Swarm Optimization (PSO) method [29]. The technique minimizes conflicts between stakeholders and developers using a variety of population sizes, between 10 to 50 set requirements. On the computational time and complexity side, CDBR shows excellent results compared to AHP. Second, DRank uses the RankBoost algorithm for learning and calculating requirements dependencies [35]. The graph is used to show or represent dependencies between requirements. There are two types of a graph generated by the DRank method—the first graph is for representing contributions, and the second graph is to represent business rules. Third, iNRP uses Least Median

Square (LMS) and Multilayer Perceptron (MLP) techniques [39]. Time-consuming testing, placing DRANK, is superior to the AHP and CBRank methods [35]. In general, ML could be used to reduce interactions with practitioners. It provides better computational efficiency at significant scalability. However, the challenge is the availability of datasets and the selection of techniques that fit the project's characteristics.

Fig. 8 shows the distribution of techniques used to address requirements dependency problems in RP, based on their technical bases. Most techniques seem to employ Decision Making and Evolutionary Algorithm technical bases.

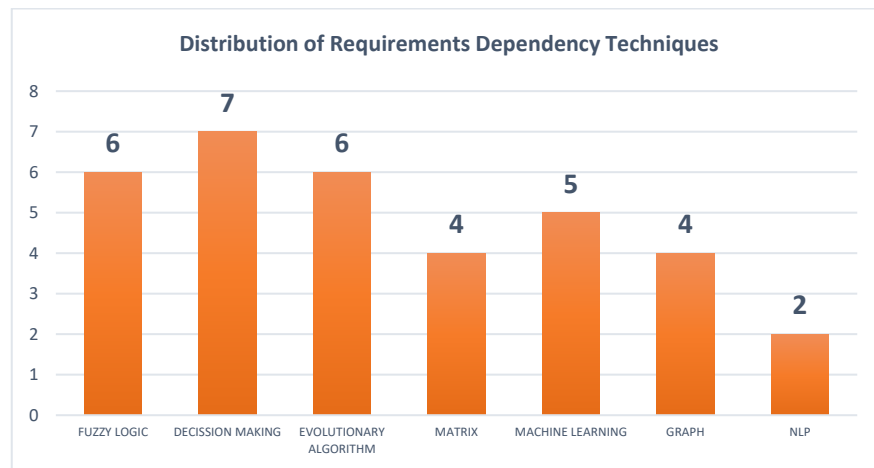


Fig. 8. Techniques of handling requirements dependency in requirements prioritisation

The brief explanation of each technique in terms of process and limitations is presented in Table 6. Based on the synthesis presented in the table, each technique has its own designated approaches in solving requirements dependency problems in RP. It can be seen that most techniques to date only handle small numbers of requirements and cover limited types of dependency. They in fact have yet to be tested using real cases with large data sets. Some are incomplete and shallow, covering trivial aspects of the matter. Stakeholders have different perceptions in assessing the importance of requirements due to diverse backgrounds and knowledge. The Fuzzy Logic techniques are widely used to solve the problem of uncertainties among stakeholders in determining priorities [7], [28], [45].

Decision Making techniques help RP by comparing requirements [12], [18], [31], [33], [37], [50], and [54]. The techniques are accurate, but the comparisons become complex when they involve a large number of requirements. The techniques therefore are not suitable for large-scale projects. ML can help resolving the issue of large data. Prior to that, ML however requires training data that are based on the generated knowledge base containing patterns or rules. Graph [38], [40], [43] and Matrix [34], [43], [44], [71] can aid to visualize the dependencies between requirements. As these techniques only focus on visualization, they usually have to be combined with other techniques because RP needs to consider the values of interest between requirements.

NLP works by reading and recognizing patterns. Although promising, a common obstacle of applying this technique is the inconsistency in the written requirements [48], [70]. As such, reading the patterns is difficult. Recognizing discrete patterns is also not so straightforward, as dependencies vary. EA is a metaheuristics-based technique [30], [42], [47], [51], [72]. Its advantage is the gene selection, which means only the best requirements can survive. EA can be combined with another method that exploits a genetic algorithm to reduce the number of pairs of elicited requirements [51]. Only pairs that allow the disambiguation of equally ranked or differently ranked requirements are elicited. However, this technique has yet to be applied in real projects to truly prove its practicality.

Table 6. Process and Limitation of Requirements Dependencies Techniques

Technique	Description	Process	Limitation
Hierarchical Fuzzy Inference System (HFIS) [7]	HFIS is built hierarchically and deals with uncertainty about human judgment.	<ul style="list-style-type: none"> Data collection Preprocessing (include Dependencies) Ranking uses HFIS Release plan generation 	It only handles two types of dependency: Coupling (Combination) and Precedence (Implication).
Technical base: FUZZY LOGIC AHP [18]	AHP-based prioritisation is performed pairwise by comparing every requirement against each other.	<ul style="list-style-type: none"> Matrix and start comparing all requirements Pairwise comparison of requirements Obtain values for each requirement Sum the row for each column Divide each value by sum the rows Averaging and normalization Priority out of 1 or 100 	Cannot be applied efficiently for large size requirements
Technical base: DECISION MAKING			
Tensor and Fuzzy Graphs [28]	Fuzzy-Graph is defined: its nodes are set of nonempty identified requirements $R = \{r1, \dots, rn\}$ and the edges show the explicit cost relation among the requirements as $C = R \times R$	<ul style="list-style-type: none"> Eliciting importance values of functional requirements (FRs) regarding nonfunctional requirements (NFRs) Generating a primary prioritisation list using the tensor concept Generating a fuzzy graph of "increase/decrease cost of" dependency Generating order of cost dependency Integrating prioritization Final prioritisation 	It handles small numbers of requirements - need to be tested in large-scale requirements
Technical base: FUZZY LOGIC	Tensor Algebra is a multidimensional array that generalizes the representation of the matrix, and each dimension of the tensor is called a mode.		
Collaborative requirement prioritisation approach (CDBR) [29]	An iterative hybrid approach called the Collaborative Dependency-Based Ranking approach follows a priori and a posterior perspective for requirement prioritisation. CDBR exploits the use of Particle Swarm Optimization (PSO)	<ul style="list-style-type: none"> The cause of Initial Priority Assignment Initial Priority Assignment by stakeholder Initial Priority Assignment by developer Final priority estimation using Particle Swarm Optimization 	It only handles two types of dependency: Direct and Indirect.
Technical base: MACHINE LEARNING			
Least-Squares-Based Random Genetic Algorithm [30]	Improve method from the interactive genetic algorithm. This method begins from select the initial population by analyzing some partial input orders, or it can be done randomly.	<ul style="list-style-type: none"> The population is initialized Maintain crossover mutation The threshold value should be null The main loop is entered in the algorithm Start the while loop in which disagreement is higher than the threshold value The process which is to be performed first The result of the comparison is stored The survey is to be utilized 	It handles small numbers of requirements - need to be tested in large-scale requirements
Technical base: EVOLUTIONARY ALGORITHM			
Hierarchical Dependencies [31]	Modified AHP that consider the relationships between the stakeholders' needs and the derived requirements in the form of use cases and non-functional requirements	<ul style="list-style-type: none"> Elicitation of New Requirements Requirements Integration Relativeness Computation Requirements Prioritisation Rearrangement of Requirements 	It handles small numbers of requirements - need to be tested in large-scale requirements
Technical base: DECISION MAKING			

Technique	Description	Process	Limitation
Early mutation testing [32] Technical base: EVOLUTIONARY ALGORITHM	The mutation process is involved in the modification of software artifact (e.g., CS) by injecting artificial faults. Each mutated version is called a mutant.	<ul style="list-style-type: none"> • Mutant generation using MutML • Evaluation of the test suite adequacy • Ranking of the test suites • Adequacy score selection • Scenarios identification • Mapping test cases to requirements • Dependencies analysis • Prioritisation of requirements 	It handles simple and small numbers of requirements - need to be tested in large-scale requirements Manual process
Collaborative requirement prioritisation method [12] Technical base: DECISION MAKING	A collaborative method where both developers and stakeholders are equally involved in assigning final priority.	<ul style="list-style-type: none"> • Stakeholder's perspective and their priority inputs • Developer's perspective and their priority inputs • Dependency computation • Dependency classification • Requirement weight computation classification • Priority assignment rules to calculate the developer's priority • Prioritisation process 	It only handles three types of dependency: Complete, Limited and Inferred
Utility-based prioritisation [68] Technical base: DECISION MAKING	Utility-based prioritisation allows stakeholders to prioritise a requirement with regard to different interest dimensions.	<ul style="list-style-type: none"> • Contribution of requirements to the interest dimensions • Predefined weights for the interest dimensions • Ranking of requirements with static weights 	Need to analyse which features (interest dimensions) are useful to improve prediction quality.
Requirements Change Analysis [34] Technical base: MATRIX DRank [35]	A method based on the changes that change themselves, which are initiated at higher levels.	<ul style="list-style-type: none"> • Analyzing the change using functions • Identifying the difficult changing and • Identifying the dependencies using a matrix 	Need to identify the effort to implement a requirement change and to apply the method to a more complex case study.
DRank [35] Technical base: MACHINE LEARNING	An automated method by combining machine learning with the link analysis technique	<ul style="list-style-type: none"> • Select ranking criteria • Select a scale value for each requirement • Prioritise sampled requirements pairs • Generate subjective requirements prioritisation • Generate requirement dependency graphs (RDGs) • Analyze contribution order • Integrate the prioritisation 	It only handles two types of dependency: Contribution and Business
Enhancing the Process of Requirements Prioritization in Agile Software Development - A Proposed Model [36] Technical base: Technical Scale	A new RP models conducted based on the exiting RP models.	<ul style="list-style-type: none"> • Requirements on project • Select requirement based on Business Value and Risk • Prioritised project backlog based on Effort Estimation and Dependency • Sprint backlog 	A future research is required to validate the proposed improvements for both the model and the technique.
Majority Voting Goal-Based (MVGB) [37] Technical base: DECISION MAKING	Prioritising the requirements with the specific concern of stakeholders.	<ul style="list-style-type: none"> • Defining evaluation function • Finding dependency level for each requirement • Finding the Requirement Prioritisation Value (RPV) for each requirement • Selecting the requirements by highest RPV 	It causes high cost/effort as compared to other techniques.

Technique	Description	Process	Limitation
Software Features Prioritisation [38] Technical base: GRAPH	A model prioritisation based on the node centrality in the probability network	<ul style="list-style-type: none"> • FPN is generated from an FM according to the dependencies between features. • The centrality values of all nodes in the generated FPN are calculated and regarded as metrics for feature prioritisation. 	It supports a small number of features. The technical performance has not been evaluated.
Interactive Next Release Problem (iNRP) [39] Technical base: MACHINE LEARNING	The model composed of three different components with distinct responsibilities: (a) interactive genetic algorithm, (b) interactive module and (c) learning model	<ul style="list-style-type: none"> • DM to specify two architectural settings • The weight of the implicit preferences in comparison to the explicit ones for the fitness calculation. • A minimum number of interactions in which the DM is willing to take part in. • The learning process is performed using the set of samples collected in the previous stage as a training dataset 	The learning model performance has not been evaluated.
Graphs and Integer Programming [40] Technical base: GRAPH & MATH	The integer programming model for requirement selection which maximizes the overall value of selected requirements while mitigating the adverse impact of selection deficiency problem (SDP). Graph-based for capturing the requirements dependencies.	<ul style="list-style-type: none"> • Input requirement set • Identify requirement dependencies • Model requirement dependencies • Specify the budget range • Run the selection model and run the propose selection model • Compare the result 	This study does not use a technique to handle dependencies specifically.
Hidden Structure Method [71] Technical base: MATRIX	A method focused on analyzing a Design Structure Matrix (DSM) based on coupling and modularity theory, and it has been used in a number of software architecture and software portfolio cases.	<ul style="list-style-type: none"> • Identify the direct dependencies and compute the visibility matrix • Identify and rank cyclic groups 	The requirements dependencies have not been weighted.
Hybrid Enriched Genetic Revamped Integer Linear Programming [42] Technical base: EVOLUTIONARY ALGORITHM	Hybrid EGRILP is a combination of Enriched Genetic Algorithm (EGA) with Revamped Integer Linear Programming (RILP) model	<ul style="list-style-type: none"> • Initial requirements • Group requirements based on dependencies • Fitness value calculation • Algorithm (EM) Algorithm • Aging factor • Revamped Integer Linear Programming (RILP) 	It supports a small number of features and time dependency only.
Component-Based Software Development (CBSD) [43] Technical base: MATRIX & GRAPH	CBSD is an approach to software development that relies on the reuse of software components to reduce the development costs and production cycle while increasing the final product's quality.	<ul style="list-style-type: none"> • Define the type of relation between each pair of functional requirements • Create a two-dimensional matrix where each row represent a functional requirement, and the rows represent the same series of requirements • Generate a directed graph from the support relationships • Apply the preceding relationship • Apply the conflicted relationship 	The proposed algorithm has not been tested.
Traceability Metrics [44] Technical base: MATRIX	Methods to provide support for understanding relations between requirements	<ul style="list-style-type: none"> • Parse artifacts and trace links • Generate traceability graph • Calculate traceability metrics 	It focuses on Agile process and risk factor only.

Technique	Description	Process	Limitation
Fuzzy Inference System (FIS) [45] Technical base: FUZZY LOGIC	A mathematical interpretation to model and deal with the uncertainty in human estimation and their limited knowledge.	<ul style="list-style-type: none"> • Preprocessing • Ranking • Generating the release plan 	It handles small numbers of requirements and only supports two types of dependency: Coupling (Combination) and Precedence (Implication). It uses synthetic dataset and supports value-based dependency only.
Multi-objective evolutionary algorithms (MOEAs) [15] Technical base: EVOLUTIONARY ALGORITHM	A framework which is a Java-based for general-purpose multi-objective optimization algorithms.	<ul style="list-style-type: none"> • Generate dataset • Applying four evolutionary algorithms: NSGA-II, MOEA, GDE3, and MOEA/D • Evaluation 	
Satisfiability Modulo Theories (SMT) [48] Technical base: NLP	This method is aimed to obtain a proper initial population.	<ul style="list-style-type: none"> • Requirement elicitation • Obtain requirements dependencies • Formalization • SMT solver • Genetic algorithm • Requirement prioritised documents 	The proposed method has not been tested in industrial setting.
SNIPR [70] Technical base: NLP	Methods for prioritising requirements that exploit NLP in assisting the user in identifying interdependencies and constraints between requirements	<ul style="list-style-type: none"> • Requirements elicitation • Identify requirement dependencies and priority (NLP) • Rank requirements and identify disagreements (SMT solver) • Rerank the subset of requirements for improved accuracy (AHP) • Ranked and Selected Requirements 	Testing is limited to 100 requirements – need to be tested in large-scale requirements
AHP [50] Technical base: DECISION MAKING	A decision-making method that compares all pairs to find a higher priority	<ul style="list-style-type: none"> • The relationships between the different requirements (FR and NFR) identified • Assign a priority value for FR • Assign a priority value for NFR • Pairwise comparison for FR and NFR • Requirement prioritised 	Time-consuming for large numbers of requirements
Interactive Genetic Algorithm (IGA) [51] Technical base: EVOLUTIONARY ALGORITHM	The requirement prioritisation technique is a pairwise comparison method that exploits a genetic algorithm.	<ul style="list-style-type: none"> • Define a set of requirements • Input orders or priorities • Initialize the population of individuals with a set of totally ordered requirements • Set a few important parameters of the algorithm • Execute IGA • Determine the fitness measure (disagreement) to be used during the next selection of the best individuals 	The proposed algorithm has not been tested.
Architecture-Driven Quality Requirements Prioritization [53] Technical base: DECISION MAKING	A method employs an automated design space exploration technique based on quantitative evaluation of quality at- tributes of software architecture models.	<ul style="list-style-type: none"> • Modelling • Architecture model • Select applicable quality degrees of freedom • Operationalize quality requirements • Model effect • Automated Exploration; Automated design space exploration • Analysis • Conflict and dependencies detection • Manual analysis with decision support 	This study does not use a technique to handle dependencies specifically. The proposed method has not been tested in industrial setting.

Technique	Description	Process	Limitation
Liquid-Democracy-based Requirements Prioritization [54] Technical base: DECISION MAKING	In terms of liquid democracy, stakeholders can either evaluate the interest dimensions directly or delegate their vote for a specific interest dimension (or requirement) to a stakeholder who is more qualified to evaluate this dimension/ requirement.	<ul style="list-style-type: none"> Group members had to provide a single rating (1-5 stars) for every requirement. a group-based multi-attribute utility (MAUT) based approach was used to determine a prioritization The members were asked to comment on issues for every requirement. Every requirement was discussed individually by the group 	It supports a small number of features and The proposed method has not been tested in industrial setting.
Supervised Classification Techniques [55] Technical base: MACHINE LEARNING	Introduce an intelligent system in order to tackle the open issues regarding dependencies between requirements by using supervised learning techniques based on text-mining.	<ul style="list-style-type: none"> First, showed 30 different requirements regarding a sports watch to each participant. the second step, the set of randomly ordered requirements was shown to participants. The participants were asked to manually find all correct dependencies of type requires between two requirements based on the shown title and description. After collecting all the dependencies, Natural Language Processing techniques have been exploited to support the automated detection of dependencies 	Limited to support type of dependency: requires.
Dependency-aware software release planning (DA-SRP) [57] Technical base: GRAPH	Dependency aware software release planning (DA-SRP) maximizes the overall value of an optimal subset of features while considering the influences of value-related dependencies extracted from user preferences.	<ul style="list-style-type: none"> The process starts with identification of value-related dependencies from collected user preferences. Identified value-related dependencies will be modeled using the algebraic structure of fuzzy graphs the resulting model is referred to as the Feature Dependency Graph (FDG) of the system Finally, perform dependency-aware release planning to find an optimal configuration of the features using the proposed integer programming model. 	Limited to support type of dependency: value-related dependencies
MOSAs [58] Technical base: EVOLUTIONARY ALGORITHM Integrating Active Learning with Ontology-Based Retrieval [60] Technical base: MACHINE LEARNING	MOSAs were designed for generic optimization problems, and therefore they do not make any domain/problem-specific assumptions when applied AL is a form of ML in which a learning algorithm interactively queries an oracle (typically a human expert) to obtain the desired label for new data points. It has been effective in reducing human efforts in the data analysis process	<ul style="list-style-type: none"> Collect requirements Set specific cost overrun probability distribution Apply URP Requirements permutation Requirements Review Requirements Dependency Extraction by Active Learning (RD-AL) Requirements Dependency Extraction by Ontologies Natural Language Pre-processor Pipeline 	Handle the attribute values of requirements are static The ontology depend on context engineering

Table 7 maps the techniques across the dependency types. It can be seen that most techniques emphasize on Internal dependency, particularly on Functional. Among the three Functional variations,

Combination is the most explored. Across the dependency types, DRank (ML) is the most applied technique. However, DRank is used so far to address Internal-Functional dependency only.

Table 7. Mapping of Techniques and Types of Requirements Dependency

Technical-base	Technique	INTERNAL						EXTERNAL								
		Functional		Structural				Related	Time-Related	Value-Related	Resources	Human	Business Process			
		Combination		C	D	E	F	G	H	I	J	K	L	M	N	O
		A*	B													
Decision Making	AHP															
	Hierarchical Dependencies															
	Collaborative requirement prioritisation method															
	Utility-based prioritisation															
	Majority Voting Goal-Based (MVGB)															
Evolutionary Algorithm	Architecture-Driven Quality Requirements Prioritization															
	Liquid-Democracy-based															
	Least-Squares-Based Random															
	Early mutation testing															
	Hybrid Enriched Genetic Revamped Integer Linear Programming															
Fuzzy Logic	Multi-objective evolutionary algorithms (MOEAs)															
	Interactive Genetic Algorithm (IGA)															
	MOSAs															
	Hierarchical Fuzzy Inference System (HFIS)															
Graph	Tensor and Fuzzy Graphs															
	Fuzzy Inference System (FIS)															
	Software Features Prioritisation															
Graph & Math	Dependency-Aware Software Release Planning (DA-SRP)															
	Graphs and Integer Programming															
Machine Learning	Collaborative requirement prioritization approach (CDBR)															
	DRank															
	Interactive Next Release Problem (iNRP)															
	Integrating Active Learning with Ontology-Based Retrieval															
Matrix	Supervised Classification Techniques															
	Requirements Change Analysis															
	Hidden Structure Method															
Matrix & Graph	Traceability Metrics															
	Component-Based Software Development (CBSD)															
NLP	Satisfiability Modulo Theories (SMT)															
	SNIPR															
Technical Scale	Enhancing the Process of Requirements Prioritization in Agile Software Development															

^a A: Complete, B: Limited, C: Exclusion, D: Implication, E: Direct, F: Indirect, G: Refines, H: Time-Based, I: Cost-Based, J: Revenue-Based, K: Dependencies due to downstream activities, L: Team-based dependencies, M: Inter-domain dependencies, N: Intra-domain dependencies, O: Dependencies among user stories.

The least explored techniques are AHP Modified (Decision Making), Fuzzy Graph and Tensor Algebra (Fuzzy Logic), Hidden Structure (Matrix), Interactive GA (EA) and Hybrid Enriched Genetic Revamped Integer Linear Programming (EA). The least investigated Internal dependency is Structural-Refines. In comparison to Internal dependency, External dependency receives lesser attention. In fact, only Time-related and Value-related dependencies are being addressed involving merely three techniques, namely Fuzzy Graph and Tensor Algebra (Fuzzy Logic), Genetic Algorithm (EA) and Hybrid Enriched Genetic Revamped Integer Linear Programming (EA).

Most of the techniques proposed by the authors are still at the research stage and have not been applied to solve real-world industry problems. One technique that has been applied in real-world cases is DA-SRP [57]. This technique is used in the development of industrial software called PMS-II. The case study involves 23 features considering user preferences within a certain budget. The preference matrix for PMS-II is constructed based on user preferences. The calculation of dependency strength and quality related to values is then performed using fuzzy membership functions. This technique results in optimal feature selection. Another technique is Integrating Active Learning with Ontology-Based Retrieval [60]. This technique is applied to two industrial datasets, namely Siemens Austria and Blackline Safety Corp Canada. Both companies have collected software requirements and manually determined RD. The application of the proposed technique in the research reduces efforts with good accuracy, achieving 86% accuracy in the second company.

Based on these findings, several preliminary interpretations can be made. One possible explanation on why most techniques address Internal dependency is because such dependency is definite and structured. Thus, it is more straightforward to tackle. On the other hand, External dependency involves vague and diverse elements that rely heavily on the nature of project. The elements in fact vary across projects, which are not so apparent to determine. This also helps to explain why among External dependency types, only Cost-based and Time-based are addressed by the techniques. This is due to the fact that cost and time are the most objective variables in projects. Another challenge in handling external dependencies is the conditions that are beyond the control of system developers. External dependencies can be addressed by involving stakeholders as business owners or parties related to the system requirements being developed. Considerations from these stakeholders can be used as an important factor in determining the priority sequence.

5. Conclusion

This study has provided an understanding of requirements dependency in RP in terms of its impacts, types and techniques based on a review made on thirty selected articles. The results show that requirements dependency has significant impacts on RP. Ignoring requirements dependency during RP could delay product release and increase project cost as well as project risk. The different types of requirements dependency have also been identified. There are at least 14 types, which can be clustered into two categories: Internal and External. Each type has different characteristics and thus requires different techniques. There are 28 techniques that are capable of handling requirements dependency problems in RP. These techniques are derived from various technical bases, including Fuzzy Logic, Decision Making, Evolutionary Algorithm, Matrix, Machine Learning, Graph, and Neuro-Linguistic Programming.

Some limitations and gaps are observed in the reviewed articles, which require further research. Most techniques focus on Internal dependency, rather than External. In fact, Functional is more investigated than Structural in Internal dependency. With regards to practicality, most techniques to date are still being tested in laboratory settings with small data sets and covering limited types of dependency. Their scalability and efficiency in handling large-scale requirements are thus arguable. Future studies should be able to apply RP techniques by considering dependency factors with various types in large-scale software development with a set of requirements. As RP plays an important role in ensuring the success of a software project, effective and yet practical solutions are necessary. In prioritizing requirements that involve multiple factors and a large number of requirements, combining Multicriteria Decision Making techniques with Machine Learning can be beneficial. However, it requires adjustments based on business

value. In this study, the QAC process was performed manually without the use of tools. Therefore, future reviews can utilize tools such as the Cochrane Risk of Bias tool to obtain stronger assessment results.

Acknowledgment

The authors would like to thank Universiti Kebangsaan Malaysia (PP-FTSM-2020) and Universitas Ahmad Dahlan for supporting this research.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

- [1] S. Fahmy and J. H. Yahaya, "The Role of Human in Software Configuration Management," in *The 2018 7th International Conference on Software and Computer Applications*, 2018, pp. 56–60, doi: [10.1145/3185089.3185117](https://doi.org/10.1145/3185089.3185117).
- [2] F. Hujainah, R. Binti, A. B. U. Bakar, and M. A. Abdulgaber, "Investigation of Requirements Interdependencies in Existing Techniques of Requirements Prioritization," *Hrčak*, vol. 26, no. 4, pp. 1186–1190, 2019, doi: [10.17559/TV-20171129125407](https://doi.org/10.17559/TV-20171129125407).
- [3] N. Garg, "Recent Advancements in Requirement Elicitation and Prioritization Techniques," in *2015 International Conference on Advances in Computer Engineering and Applications*, Ghaziabad, India: IEEE, 2015, pp. 237–240. doi: [10.1109/ICACEA.2015.7164702](https://doi.org/10.1109/ICACEA.2015.7164702).
- [4] M. Soumya Krishnan, "RFP based Requirement Prioritization - A One-Step Solution," *Mater. Today Proc.*, vol. 5, no. 1, pp. 642–649, 2018, doi: [10.1016/j.matpr.2017.11.128](https://doi.org/10.1016/j.matpr.2017.11.128).
- [5] P. Bajaj and V. Arora, "Multi-Person Decision-Making for Requirements Prioritization using Fuzzy AHP," *ACM SIGSOFT Software Engineering Notes*, vol. 38, no. 5, pp. 1–6, 2013. doi: [10.1145/2507288.2507302](https://doi.org/10.1145/2507288.2507302).
- [6] R. H. Al-Ta'ani and R. Razali, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 6, no. 6, p. 846, 2016, doi: [10.18517/ijaseit.6.6.1375](https://doi.org/10.18517/ijaseit.6.6.1375).
- [7] M. Alrashoud, M. Al-hammadi, A. Ghoneim, E. Hazza, F. Alqahtani, and A. Abhari, "Cognitive and Hierarchical Fuzzy Inference System for Generating Next Release Planning in SaaS Applications," *IEEE Access*, vol. 7, pp. 102966–102974, 2019, doi: [10.1109/ACCESS.2019.2929214](https://doi.org/10.1109/ACCESS.2019.2929214).
- [8] M. Masood, F. Azam, M. W. Anwar, and A. Amjad, "Defining Meta-Model for Value-Oriented Requirement Prioritization Technique," in *Proceedings of the 2019 7th International Conference on Computer and Communications Management*, Bangkok, Thailand: ACM New York, NY, USA, 2019, pp. 72–77. doi: <https://doi.org/10.1145/3348445.3352739>.
- [9] R. K. Chopra, V. Gupta, and D. S. Chauhan, "Experimentation on accuracy of non functional requirement prioritization projects \mathfrak{C} ," *Perspect. Sci.*, vol. 8, pp. 79–82, 2016, doi: [10.1016/j.pisc.2016.04.001](https://doi.org/10.1016/j.pisc.2016.04.001).
- [10] M. A. Rahman, R. Razali, and D. Singh, "A Risk Model of Requirements Change Impact Analysis," *J. Softw.*, vol. 9, no. 1, pp. 76–81, 2014, doi: [10.4304/jsw.9.1.76-81](https://doi.org/10.4304/jsw.9.1.76-81).
- [11] N. C. Pa and A. M. Zain, "A Survey of Communication Content in Software Requirements Elicitation involving Customer and Developer," *J. Softw. Syst. Dev.*, vol. 2011, pp. 1–12, 2011, doi: [10.5171/2011.742200](https://doi.org/10.5171/2011.742200).
- [12] A. Gupta and C. Gupta, "Towards Dependency Based Collaborative Method for Requirement Prioritization," *2018 11th Int. Conf. Contemp. Comput. IC3 2018*, pp. 1–3, 2018, doi: [10.1109/IC3.2018.8530542](https://doi.org/10.1109/IC3.2018.8530542).

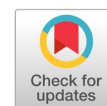
- [13] H. Sheemar, "Enhancing User-Stories Prioritization Process in Agile Environment," in *2017 International Conference on Innovations in Control, Communication and Information Systems (ICICCI)*, Greater Noida, India: IEEE, pp. 1-6, 2017. doi: [10.1109/ICICCI.2017.8660760](https://doi.org/10.1109/ICICCI.2017.8660760).
- [14] F. F. Ismail, R. Razali, and Z. Mansor, "Considerations for Cost Estimation of Software Testing Outsourcing Projects," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 9, no. 1, pp. 142-152, 2019. doi: [10.18517/ijaseit.9.1.6382](https://doi.org/10.18517/ijaseit.9.1.6382).
- [15] A. Sureka, "Requirements Prioritization and Next-Release Problem under Non-Additive Value Conditions," in *2014 23rd Australian Software Engineering Conference*, Milsons Point, NSW, Australia: IEEE, 2014, pp. 120-123. doi: [10.1109/ASWEC.2014.12](https://doi.org/10.1109/ASWEC.2014.12).
- [16] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. R. Mahrin, "A systematic literature review of software requirements prioritization research," *Inf. Softw. Technol.*, vol. 56, no. 6, pp. 568-585, 2014, doi: [10.1016/j.infsof.2014.02.001](https://doi.org/10.1016/j.infsof.2014.02.001).
- [17] N. Kukreja, S. S. Payyavula, B. Boehm, and S. Padmanabhuni, "Value-based requirements prioritization: Usage experiences," *Procedia Comput. Sci.*, vol. 16, pp. 806-813, 2013, doi: [10.1016/j.procs.2013.01.084](https://doi.org/10.1016/j.procs.2013.01.084).
- [18] A. Mustapha and N. Ibrahim, "Minimizing Inter-Dependency Issues Of Requirements In Parallel Developing Software Projects With Ahp," *COMPUSOFT, An Int. J. Adv. Comput. Technol.*, vol. 8, no. Viii, pp. 3317-3323, 2019. Available at: <https://search.proquest.com/openview/be41202e4771ac573941bf7633203a2c/1?pq-origsite=gscholar&cbl=2032622>.
- [19] A. Martakis, "Handling Requirements Dependencies in Agile Projects : A Focus Group with Agile Software Development Practitioners," in *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, Paris, France: IEEE, 2013, pp. 1-11. doi: [10.1109/RCIS.2013.6577679](https://doi.org/10.1109/RCIS.2013.6577679).
- [20] T. S. G. International, "Chaos Report 2015," pp. 1-13, 2015. Available at: https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf
- [21] F. Hujainah, R. B. A. Bakar, M. A. Abdulgaber, and K. Z. Zamli, "Software Requirements Prioritisation: A Systematic Literature Review on Significance, Stakeholders, Techniques and Challenges," *IEEE Access*, vol. 6, pp. 71497-71523, 2018, doi: [10.1109/ACCESS.2018.2881755](https://doi.org/10.1109/ACCESS.2018.2881755).
- [22] T. Amelia and R. B. Mohamed, "Review on Cost-Value Approach for Requirements Prioritization Techniques," *Proc. - 2018 5th Int. Conf. Inf. Technol. Comput. Electr. Eng. ICITACEE 2018*, pp. 310-314, 2018, doi: [10.1109/ICITACEE.2018.8576908](https://doi.org/10.1109/ICITACEE.2018.8576908).
- [23] F. Sher, D. N. A. Jawawi, R. Mohamad, and M. I. Babar, "Requirements prioritization techniques and different aspects for prioritization a systematic literature review protocol," *2014 8th Malaysian Softw. Eng. Conf. MySEC 2014*, no. December, pp. 31-36, 2014, doi: [10.1109/MySec.2014.6985985](https://doi.org/10.1109/MySec.2014.6985985).
- [24] M. Sufian, Z. Khan, S. Rehman, and W. H. Butt, "A Systematic Literature Review : Software Requirements Prioritization Techniques," in *2018 International Conference on Frontiers of Information Technology (FIT)*, Islamabad, Pakistan, Pakistan: IEEE Computer Society, 2018, pp. 35-40. doi: [10.1109/FIT.2018.00014](https://doi.org/10.1109/FIT.2018.00014).
- [25] A. M. Pitangueira, R. S. P. Maciel, M. De Oliveira Barros, and A. S. Andrade, "A systematic review of software requirements selection and prioritization using SBSE approaches," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8084 LNCS, pp. 188-208, 2013, doi: [10.1007/978-3-642-39742-4_15](https://doi.org/10.1007/978-3-642-39742-4_15).
- [26] R. H. Al-Ta'ani and R. Razali, "A Framework for Requirements Prioritisation Process in an Agile Software Development Environment: Empirical Study," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 6, no. 6, p. 846, 2016, doi: [10.18517/ijaseit.6.6.1375](https://doi.org/10.18517/ijaseit.6.6.1375).
- [27] B. Kitchenham *et al.*, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Durham, UK, pp. 1-65, 2007. Available at: https://www.elsevier.com/_data/promis_misc/525444systematicreviewsguide.pdf
- [28] N. Misaghian, H. Motameni, and M. Rabbani, "Prioritizing interdependent software requirements using tensor and fuzzy graphs," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 27, no. 4, pp. 2697-2717, 2019, doi: [10.3906/elk-1806-179](https://doi.org/10.3906/elk-1806-179).

- [29] A. Gupta and C. Gupta, "CDBR: A semi-automated collaborative execute-before-after dependency-based requirement prioritization approach," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 2, pp. 421-432, 2018, doi: [10.1016/j.jksuci.2018.10.004](https://doi.org/10.1016/j.jksuci.2018.10.004).
- [30] H. Ahuja and U. Batra, "Performance Enhancement in Requirement Prioritization by Using Least-Squares-Based Random Genetic Algorithm," in *Innovations in Computational Intelligence*, Singapore: Springer, Singapore, vol. 713, pp. 251-263, 2018. doi: [10.1007/978-981-10-4555-4_17](https://doi.org/10.1007/978-981-10-4555-4_17)
- [31] L. Alawneh, "Requirements Prioritization Using Hierarchical Dependencies," in *Advances in Intelligent Systems and Computing*, AISC, Volu.Cham: Springer, Cham, vol. 558, pp. 459-464, 2018. doi: [10.1007/978-3-319-54978-1](https://doi.org/10.1007/978-3-319-54978-1).
- [32] N. Condori-Fernandez, M. F. Granda, and T. E. J. Vos, "Towards a functional requirements prioritization with early mutation testing," *2018 IEEE/ACM 5th Int. Work. Requir. Eng. Test.*, pp. 21-24, 2018, doi: [10.1145/3195538.3195539](https://doi.org/10.1145/3195538.3195539).
- [33] A. Felfernig and M. Stettinger, "Towards Utility-based Prioritization of Requirements in Open Source Environments," in *2018 IEEE 26th International Requirements Engineering Conference*, IEEE, Ed., Banff, Canada, 2018, pp. 406-411. doi: [10.1109/RE.2018.00-17](https://doi.org/10.1109/RE.2018.00-17).
- [34] S. Jayatilleke, R. Lai, and K. Reed, "A method of requirements change analysis," *Requir. Eng.*, vol. 23, no. 4, pp. 493-508, 2017, doi: [10.1007/s00766-017-0277-7](https://doi.org/10.1007/s00766-017-0277-7).
- [35] F. Shao, R. Peng, H. Lai, and B. Wang, "DRank: A semi-automated requirements prioritization method based on preferences and dependencies," *J. Syst. Softw.*, vol. 126, pp. 141-156, 2017, doi: [10.1016/j.jss.2016.09.043](https://doi.org/10.1016/j.jss.2016.09.043).
- [36] M. Alkandari and A. Al-shammeri, "Enhancing the Process of Requirements Prioritization in Agile Software Development - A Proposed Model," *J. Softw.*, vol. 12, no. 6, pp. 439-453, 2017, doi: [10.17706/jsw.12.6.439-453](https://doi.org/10.17706/jsw.12.6.439-453).
- [37] R. M. Liaqat, M. A. Ahmed, F. Azam, and B. Mehboob, "A Majority Voting Goal Based technique for Requirement Prioritization," *2016 22nd Int. Conf. Autom. Comput. ICAC 2016 Tackling New Challenges Autom. Comput.*, pp. 435-439, 2016, doi: [10.1109/ICAC.2016.7604958](https://doi.org/10.1109/ICAC.2016.7604958).
- [38] Z. Peng, J. W. B, K. He, and H. Li, "An Approach for Prioritizing Software Features Based on Node Centrality in Probability," in *Computer Science*, in vol 9679, vol. 1. Cham: Springer, Cham, 2016, pp. 106-121. doi: [10.1007/978-3-319-35122-3](https://doi.org/10.1007/978-3-319-35122-3).
- [39] A. Allex, A. Matheus, I. Yeltsin, A. Dantas, and J. Souza, "An Architecture based on interactive optimization and machine learning applied to the next release problem," *Autom. Softw. Eng.*, vol. 24, no. 3, pp. 623-671, 2016, doi: [10.1007/s10515-016-0200-3](https://doi.org/10.1007/s10515-016-0200-3).
- [40] D. Mougouei, "Factoring Requirement Dependencies in Software Requirement Selection using Graphs and Integer Programming," in *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Singapore: IEEE International Conference, 2016, pp. 884-887. doi: [10.1145/2970276.2975936](https://doi.org/10.1145/2970276.2975936)
- [41] R. Lagerström, M. Addibpour, and F. Heiser, "Product Feature Prioritization Using the Hidden Structure Method: A Practical Case at Ericsson," in *2016 Proceedings of PICMET '16: Technology Management for Social Innovation*, Honolulu, Hawaii, USA: PICMET, 2016, pp. 2308-2315. doi: [10.1109/PICMET.2016.7806519](https://doi.org/10.1109/PICMET.2016.7806519).
- [42] S. Valsala and A. R. Nair, "Requirement Prioritization and Scheduling in Software Release Planning Using Hybrid Enriched Genetic Revamped Integer Linear Programming Model Karpagam University, Coimbatore, Scientist Department, Bangalore, India," *Res. J. Appl. Sci. Eng. Technol.*, vol. 12, no. 3, pp. 347-354, 2016, doi: [10.19026/rjaset.12.2342](https://doi.org/10.19026/rjaset.12.2342).
- [43] R. Alzyoudi, K. Almakadmeh, and H. Natoureh, "A Probability Algorithm for Requirement Selection In Component-Based Software Development," *IPAC '15 Proc. Int. Conf. Intell. Inf. Process. Secur. Adv. Commun.*, pp. 1-6. doi: [10.1145/2816839.2816871](https://doi.org/10.1145/2816839.2816871).
- [44] P. R. B and M. Patrick, "Estimating the Implementation Risk of Requirements in Agile Software Development Projects with Traceability Metrics," in *International Working Conference on Requirements*

- Engineering: Foundation for Software Quality*, Switzerland: Springer International Publishing, 2015, pp. 81–97. doi: [10.1007/978-3-319-16101-3](https://doi.org/10.1007/978-3-319-16101-3).
- [45] M. Alrashoud and A. Abhari, “Intelligent Automation & Soft Computing Perception-Based Software Release Planning,” *Intell. Autom. Soft Comput.*, vol. 21, no. 2, pp. 175–195, 2015, doi: [10.1080/10798587.2014.960229](https://doi.org/10.1080/10798587.2014.960229).
- [46] J. Cox, E. Bouwers, M. Van Eekelen, and J. Visser, “Measuring Dependency Freshness in Software Systems,” in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Florence, Italy: IEEE, 2015, pp. 109–118. doi: [10.1109/ICSE.2015.140](https://doi.org/10.1109/ICSE.2015.140).
- [47] A. Sureka, “Requirements Prioritization and Next-Release Problem under Non-Additive Value Conditions,” in *2014 23rd Australasian Software Engineering Conference*, Sydney: IEEE, 2014, pp. 120–123. doi: [10.1109/ASWEC.2014.12](https://doi.org/10.1109/ASWEC.2014.12).
- [48] Z. Tong, Q. Zhuang, Q. Guo, and P. Ma, “Research on Technologies of Software Requirements Prioritization,” in *International Conference on Trustworthy Computing and Services ISCTCS 2013*, Beijing, China: Springer, Berlin, Heidelberg, 2014, pp. 9–21. doi: [10.1007/978-3-662-43908-1](https://doi.org/10.1007/978-3-662-43908-1).
- [49] J. Mczara, S. Sarkani, T. Holzer, and T. Eveleigh, “Software requirements prioritization and selection using linguistic tools and constraint solvers — a controlled experiment,” *Empir. Softw. Eng.*, vol. 20, no. 6, pp. 1721–1761, 2014, doi: [10.1007/s10664-014-9334-8](https://doi.org/10.1007/s10664-014-9334-8).
- [50] F. Fellir, K. Nafil, and Rajaa Touahni, “System requirements prioritization based on AHP,” in *2014 Third IEEE International Colloquium in Information Science and Technology (CIST)*, Tetouan, Morocco: IEEE, 2014, pp. 163–167. doi: [10.1109/CIST.2014.7016612](https://doi.org/10.1109/CIST.2014.7016612).
- [51] P. Tonella, A. Susi, and F. Palma, “Interactive requirements prioritization using a genetic algorithm,” *Inf. Softw. Technol.*, vol. 55, no. 1, pp. 173–187, 2013, doi: [10.1016/j.infsof.2012.07.003](https://doi.org/10.1016/j.infsof.2012.07.003).
- [52] M. Daneva *et al.*, “Agile requirements prioritization in large-scale outsourced system projects : An empirical study,” *J. Syst. Softw.*, vol. 86, no. 5, pp. 1333–1353, 2013, doi: [10.1016/j.jss.2012.12.046](https://doi.org/10.1016/j.jss.2012.12.046).
- [53] A. Koziolok, “Architecture-Driven Quality Requirements Prioritization,” in *2012 First IEEE International Workshop on the Twin Peaks of Requirements and Architecture (TwinPeaks)*, Chicago, Illinois, USA: IEEE, 2012, pp. 15–19. doi: [10.1109/TwinPeaks.2012.6344554](https://doi.org/10.1109/TwinPeaks.2012.6344554).
- [54] R. Samer, M. Stettinger, and A. Felfernig, “Group Recommender User Interfaces for Improving Requirements Prioritization,” *UMAP 2020 - Proc. 28th ACM Conf. User Model. Adapt. Pers.*, pp. 221–229, 2020, doi: [10.1145/3340631.3394851](https://doi.org/10.1145/3340631.3394851).
- [55] M. Atas, R. Samer, and A. Felfernig, “Automated Identification of Type-Specific Dependencies between Requirements,” *Proc. - 2018 IEEE/WIC/ACM Int. Conf. Web Intell. WI 2018*, pp. 688–695, 2019, doi: [10.1109/WI.2018.00-10](https://doi.org/10.1109/WI.2018.00-10).
- [56] H. Saeeda, J. Dong, Y. Wang, and M. A. Abid, “A proposed framework for improved software requirements elicitation process in SCRUM: Implementation by a real-life Norway-based IT project,” *J. Softw. Evol. Process*, vol. 32, no. 7, pp. 1–24, 2020, doi: [10.1002/smr.2247](https://doi.org/10.1002/smr.2247).
- [57] D. Mougouei and D. M. W. Powers, “Dependency-aware software release planning through mining user preferences,” *Soft Comput.*, vol. 24, no. 15, pp. 11673–11693, 2020, doi: [10.1007/s00500-019-04630-y](https://doi.org/10.1007/s00500-019-04630-y).
- [58] H. Zhang, M. Zhang, T. Yue, S. Ali, and Y. Li, “Uncertainty-wise Requirements Prioritization with Search,” *ACM Trans. Softw. Eng. Methodol.*, vol. 30, no. 1, pp. 1–54, 2021, doi: [10.1145/3408301](https://doi.org/10.1145/3408301).
- [59] F. Hujainah, R. Binti Abu Bakar, A. B. Nasser, B. Al-haimi, and K. Z. Zamli, “SRPTackle: A semi-automated requirements prioritisation technique for scalable requirements of software system projects,” *Inf. Softw. Technol.*, vol. 131, no. November 2020, p. 106501, 2021, doi: [10.1016/j.infsof.2020.106501](https://doi.org/10.1016/j.infsof.2020.106501).
- [60] G. Deshpande *et al.*, “Requirements Dependency Extraction by Integrating Active Learning with Ontology-Based Retrieval,” *Proc. IEEE Int. Conf. Requir. Eng.*, vol. 2020-Augus, pp. 78–89, 2020, doi: [10.1109/RE48521.2020.00020](https://doi.org/10.1109/RE48521.2020.00020).
- [61] A. Ali, Y. Hafeez, S. Hussain, and S. Yang, “Role of Requirement Prioritization Technique to Improve the Quality of Highly-Configurable Systems,” *IEEE Access*, vol. 8, pp. 27549–27573, 2020, doi: [10.1109/ACCESS.2020.2971382](https://doi.org/10.1109/ACCESS.2020.2971382).

- [62] A. Vescan and C. Serban, "Towards a new Test Case Prioritization Approach based on Fuzzy Clustering Analysis," *Proc. - 2020 IEEE Int. Conf. Softw. Maint. Evol. ICSME 2020*, pp. 786–788, 2020, doi: [10.1109/ICSME46990.2020.00091](https://doi.org/10.1109/ICSME46990.2020.00091).
- [63] M. Sadiq and V. S. Devi, "A rough-set based approach for the prioritization of software requirements," *Int. J. Inf. Technol.*, vol. 14, no. 1, pp. 447–457, 2022, doi: [10.1007/s41870-021-00749-0](https://doi.org/10.1007/s41870-021-00749-0).
- [64] P. Carlshamre, K. Sandahl, M. Lindvall, B. Regnell, and J. Natt och Dag, "An industrial survey of requirements interdependencies in software product release planning," *Proc. IEEE Int. Conf. Requir. Eng.*, pp. 84–91, 2001, doi: [10.1109/ISRE.2001.948547](https://doi.org/10.1109/ISRE.2001.948547).
- [65] C. Li, M. van den Akker, S. Brinkkemper, and G. Diepen, "An integrated approach for requirement selection and scheduling in software release planning," *Requir. Eng.*, vol. 15, no. 4, pp. 375–396, May 2010, doi: [10.1007/S00766-010-0104-X](https://doi.org/10.1007/S00766-010-0104-X).
- [66] Å. G. Dahlstedt and A. Persson, "Requirements Interdependencies : State of the Art and Future Challenges," in *Engineering and Managing Software Requirements*, Springer, pp. 95-116, 2005. doi: [10.1007/3-540-28244-0_5](https://doi.org/10.1007/3-540-28244-0_5)
- [67] G. Deshpande, "SReYantra: Automated software requirement inter-dependencies elicitation, analysis and learning," *Proc. - 2019 IEEE/ACM 41st Int. Conf. Softw. Eng. Companion, ICSE-Companion 2019*, pp. 186–187, May 2019, doi: [10.1109/ICSE-COMPANION.2019.00076](https://doi.org/10.1109/ICSE-COMPANION.2019.00076).
- [68] A. Felfernig *et al.*, "Towards utility-based prioritization of requirements in open source environments," *Proc. - 2018 IEEE 26th Int. Requir. Eng. Conf. RE 2018*, pp. 406–411, 2018, doi: [10.1109/RE.2018.00-17](https://doi.org/10.1109/RE.2018.00-17).
- [69] T. L. Saaty, "Decision Making – The Analytic Hierarchy And Network Processes (AHP / ANP)," *J. Syst. Sci. Syst. Eng.*, vol. 13, no. 1, pp. 1–35, 2004. doi: [10.1007/s11518-006-0151-5](https://doi.org/10.1007/s11518-006-0151-5).
- [70] J. McZara, S. Sarkani, T. Holzer, and T. Eveleigh, "Software requirements prioritization and selection using linguistic tools and constraint solvers—a controlled experiment," *Empir. Softw. Eng.*, vol. 20, no. 6, pp. 1721–1761, 2015, doi: [10.1007/s10664-014-9334-8](https://doi.org/10.1007/s10664-014-9334-8).
- [71] R. Lagerström, M. Addibpour, and F. Heiser, "Product feature prioritization using the Hidden Structure method: A practical case at Ericsson," *PICMET 2016 - Portl. Int. Conf. Manag. Eng. Technol. Technol. Manag. Soc. Innov. Proc.*, pp. 2308–2315, 2017, doi: [10.1109/PICMET.2016.7806519](https://doi.org/10.1109/PICMET.2016.7806519).
- [72] N. Condori-Fernandez and P. Lago, "Characterizing the contribution of quality requirements to software sustainability," *J. Syst. Softw.*, vol. 137, pp. 289–305, 2018, doi: [10.1016/j.jss.2017.12.005](https://doi.org/10.1016/j.jss.2017.12.005).

A hybrid model for aspect-based sentiment analysis on customer feedback: research on the mobile commerce sector in Vietnam



Thanh Ho ^{a,b,1,*}, Hien Minh Bui ^{c,2}, Thai Kim Phung ^{c,3}

^a University of Economics and Law, Ho Chi Minh City, Vietnam

^b Viet Nam National University Ho Chi Minh City, Vietnam

^c UEH College of Technology and Design (UEH-CTD), University of Economics Ho Chi Minh City (UEH), Vietnam

¹ thanhht@uel.edu.vn; ² hienbui.192118003@st.ueh.edu.vn; ³ phungthk@ueh.edu.vn

* corresponding author

ARTICLE INFO

Article history

Received December 28, 2022

Revised April 8, 2023

Accepted April 23, 2023

Available online June 2, 2023

Keywords

Customer feedback

Sentiment analysis

Mobile commerce

Machine and deep learning

POS tagging

ABSTRACT

Feedback and comments on mobile commerce applications are extremely useful and valuable information sources that reflect the quality of products or services to determine whether data is positive or negative and help businesses monitor brand and product sentiment in customers' feedback and understand customers' needs. However, the increasing number of comments makes it increasingly difficult to understand customers using manual methods. To solve this problem, this study builds a hybrid research model based on aspect mining and comment classification for aspect-based sentiment analysis (ABSA) to deeply comprehend the customer and their experiences. Based on previous classification results, we first construct a dictionary of positive and negative words in the e-commerce field. Then, the POS tagging technique is applied for word classification in Vietnamese to extract aspects of model commerce related to positive or negative words. The model is implemented with machine and deep learning methods on a corpus comprising more than 1,000,000 customer opinions collected from Vietnam's four largest mobile commerce applications. Experimental results show that the Bi-LSTM method has the highest accuracy with 92.01%; it is selected for the proposed model to analyze the viewpoint of words on real data. The findings are that the proposed hybrid model can be applied to monitor online customer experience in real time, enable administrators to make timely and accurate decisions, and improve the quality of products and services to take a competitive advantage.



This is an open access article under the [CC-BY-SA](#) license.



1. Introduction

With today's Internet and e-commerce platforms explosion, online shopping has become easier and more convenient than ever. Mobile commerce applications have been rapidly developing. In Vietnam, four mobile commerce apps (i.e., Shopee, Lazada, Sendo, and Tiki) have the most visits on Google Play Store, with total monthly traffic of 143 million in Q4, 2020 [1]. In addition, a large amount of data from users represents online responses in the form of daily texts on mobile commerce applications. These reviews are a valuable resource for businesses to understand the users' experiences and opinions about products and services, which is helpful for both users and manufacturers [2]. However, it is becoming more difficult to identify the main patterns with the increasing number of comments every day. Therefore, an automated approach to extracting and summarizing the main patterns of online commentary is essential, with opinion mining through sentiment analysis (SA), specifically aspect-based

sentiment analysis (ABSA), posing a significant challenge. In particular, the Vietnamese language is a complex language that consists of a 29-character alphabet including Latin characters, using additional tones, such as accents (´), hypotenuses (ˆ), question marks (?), tildes (~), and heavy accents (˙). Additionally, many borrowed words are derived from Chinese, French, and English [3].

Various studies with machine learning approaches in sentiment analysis have been published. Yanuar Nurdiansyah *et al.* [4] suggested a system that utilizes the Naïve Bayes Classifier method to classify sentiment in Bahasa Indonesia movie reviews into two categories (positive and negative), with an average classification accuracy of 88.37%. Al Amrani *et al.* [5] proposed a hybrid approach to identify product reviews offered by Amazon using Random Forest (RF) and Support Vector Machine (SVM). Bolbol & Maghari *et al.* [6] conducted an experiment with various machine learning classifiers and found that Logistic Regression (LR) achieved the highest accuracy of 93% on the Arabic Tweets dataset. Other comparative studies used machine learning techniques (Naïve Bayes, SVM, Decision Tree, K-Nearest Neighbor, and Artificial Neural Network) to classify customer opinions [7]–[11]. Recently, deep learning approaches produced better performance than traditional machine learning. Peng *et al.* [12] employed Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) [13], [14], and Convolutional Neural Networks (CNN) and the results have shown that CNN has reported the accuracy of 88.22%, RNN and LSTM have reported accuracy of 68.64% and 85.32% respectively. Li *et al.* [15] combined BERT+BiLSTM+CNN to classify sentiment on the Weibo text dataset and achieved the accuracy of 92.4%.

In Vietnamese, Nguyen *et al.* [16] used the PhoBERT model to classify sentiment-based stock article titles, with an accuracy of 93%. In another research, Nguyen *et al.* [17] proposed fine-tuning BERT method for sentiment analysis of Vietnamese Reviews, the results show that the BERT-RCNN model achieves the best result with the F1-score is 91.15%. In addition, Truong *et al.* [18] used the pre-trained model PhoBERT, with other fine-tuning techniques was achieved with an accuracy of 94.28% on the UIT-VSFC dataset.

ABSA is an interesting research topic in the opinion mining field [19]. People tend to talk about many aspects in their comments about a product or service and provide many words with positive or negative connotations. Specifically, each comment will include the following four categories: (1) an aspect of a word that has a positive or negative meaning, (2) an aspect of many words with a positive or negative meaning, (3) many aspects of one word that have positive or negative connotations, and (4) many aspects of many words that have positive or negative connotations [20]. For example, “Lazada has a lot of discount codes, but app still many bugs to fix”. Here, users gave positive reviews of the “discount” aspect but negative reviews of the “app” aspect. Dealing with the comments that have many facets in sentences is an extremely challenging task.

One of the early studies in the field of user opinion aspect mining was introduced by Hu & Liu [21], which is based on the frequency of occurrence of nouns and noun phrases to exploit aspects. Subsequently, various approaches have been studied, focusing on the aspect mining problem, with the most widely used approach being the rule-based approach [22], which extracts aspects based on the grammatical relationships of words in a sentence. An aspect extraction system [23] was introduced from products that considered nouns to be aspect terms and extracted them based on POS tagging and term frequency-inverse document frequency (TF-IDF) methods. Classification algorithms, such as Naïve Bayes and SVMs which have also been applied for aspect extraction. Mai & Le [24] proposed a sequence-labeling approach that combines BiRNN and Conditional Random Fields (CRF) to concurrently extract opinion targets and detect their associated sentiments in smartphone-related datasets. Luc Phan *et al.* [25] give a suggested method for the Vietnamese aspect-based sentiment task is based on the Bi-LSTM architecture, using fastText word embeddings, their experiments demonstrate that this approach achieves the highest F1-scores of 84.48% for the aspect task and 63.06% for the sentiment task in Vietnamese Smartphone Feedback Dataset (UIT-ViSFD).

In this study, a hybrid research model that combines aspect mining and comment classification is developed. We propose an experimental method for the model that consists of five main tasks: first, we

provide a Vietnamese customer reviews dataset in the e-commerce industry; Second, we offer dictionaries and techniques to address issues in pre-processing Vietnamese text; Third, we propose the Bi-SLTM model for text sentiment classification; Fourth, we identify aspects of the product and service that users comment on, such as positive or negative comments, using POS tagging based on the result of the proposed model's output; Finally, we build a the dashboard to help managers gain deeper understanding of customers' needs and preferences, thereby improving the level of customer satisfaction and expectations about products and services. Fig. 1 provides an overview of the hybrid research model.

The remainder of this paper is structured as follows: Section 2 presents the methodology used in this study. Section 3 outlines the results and associated discussion. In this section, we detail our experiments and provide corresponding discussions. Finally, in Section 4, we summarize our work and discuss future directions for research.

2. Method

To build the hybrid model and research method, the secondary data from previous studies which are related to the research objectives of the article on topics of aspects extraction and sentiment analysis in the field of mobile commerce and others are surveyed and studied. Specifically, we surveyed the theoretical basis, analyzed business models, and proposed appropriate research models. And then, experimental research method will be applied to implement and evaluate the proposed model through the following stages: identifying business problems and collecting online comments from customers, cleaning data, integrating, and applying statistical methods, and employing machine and deep learning techniques for natural language processing to uncover hidden knowledge and insights. The implicit knowledge in the data is represented by the nuances in customer comments, specific aspects, and issues related to the nuances of words. Based on the results, recommendations related to business and management will be proposed. The proposed hybrid model for aspect-based sentiment analysis in Fig. 1 includes 5 stages.

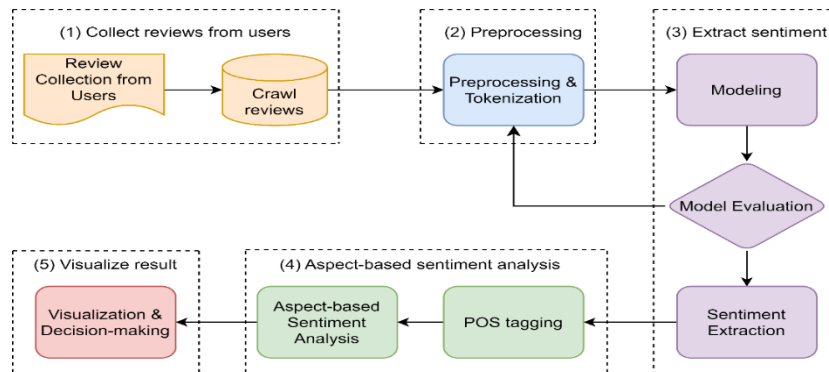


Fig. 1. A hybrid model for aspect-based sentiment analysis (Source: Authors)

2.1. Data Collection

Python language with google_play_scraper library is used to collect data from Google Play Store, which includes over 1,2 million customer feedback on four mobile commerce apps (Shopee, Lazada, Sendo, and Tiki) from 2013 to 2022. The dataset is named VN_E-commerce_Review and is published at <https://www.kaggle.com/datasets/hienbm/vietnamese-ecommerce-review>.

2.2. Data Pre-processing

After observing the original data set, there are so much noisy data such as other words not in the Vietnamese alphabet, emoji, emoticons, teen code, missing accents, and abbreviations. Therefore, the study highly focuses on pre-processing. As a result, the accuracy of proposed model has improved efficiently. The results after pre-processing will be shown in Table 1, including: (1) Converting all the words to lowercase; (2) Removing words with hashtag (#), html (< >), url (http://), mention (@); (3) Removing characters number, punctuation, and strange characters that are not in Vietnamese alphabet;

(4) Removing elongated characters, duplicate and continuous words or phrases (e.g., “ok ok okkkkkk” -> “ok”); (5) Removing duplicate emojis; (6) Adding Vietnamese accents with dictionary mapping utilizing Vietnamese dictionary words with "key" as the original word and "value" as the word without the accents. (7) Mapping teen code and abbreviation words with a dictionary (e.g. “thjk” -> “thích like”); (8) Removing extra white space; (9) Applying word tokenizer with pyvi library [26] (the library was published by Viet -Trung Tran) (eg: “tẩy chay” -> “tẩy chay boycott”). Table 1 shows the examples of the dataset before and after pre-processing.

Table 1. Dataset examples after pre-processing (Source: Authors)

Original text (in Vietnamese)	After pre-processing (in Vietnamese and English)
dịch vụ tốt và nhiệt tình 🤗🤗 tuyệt vời :)))	dịch_vụ_tốt <i>good service</i> và <i>and</i> nhiệt_tình <i>enthusiastic</i> tuyệt_vời <i>wonderful</i>
rất tiện và chất lượng lắm...	rất <i>very</i> tiện <i>convenient</i> và <i>and</i> chất_lượng <i>quality</i> lắm <i>much</i>
đề nghị xử lý các shop lừa đảo	đề_nghị <i>suggest</i> xử_lý <i>dealing with</i> các shop <i>shops</i> lừa_đảo <i>scam</i>

2.3. Modeling

This study not only intends to build a novel technique to improve sentiment classification on comments using the Bi-LSTM model, but it also aims to conduct a sentiment dictionary and extract aspects based on sentiment. The structure of the suggested model is explored in detail in this section. Fig. 2 depicts the general structure of the Bi-LSTM model.

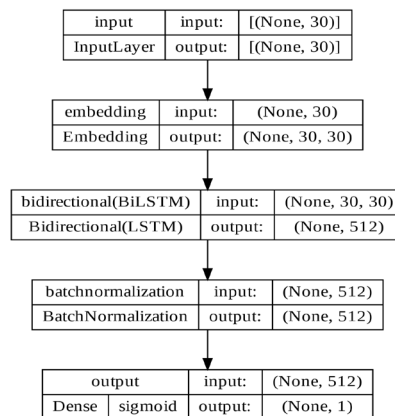


Fig. 2. The architecture of Bi-LSTM model (Source: Authors)

After pre-processing, the dataset is divided into 3 sets as input data: a training set with 1,039,240 reviews (-80% of random data), a validation test set with 129,905 reviews (-10% of random data remaining), and a test set with 129,905 reviews (-10% of random data remaining). The label results have scores from 1 to 5, and then we divide them into 3 groups corresponding to scores 1-3 as Negative and scores 4-5 as Positive [26]. Fig. 3 shows the number of text reviews for each class for the training, validation, and test sets.

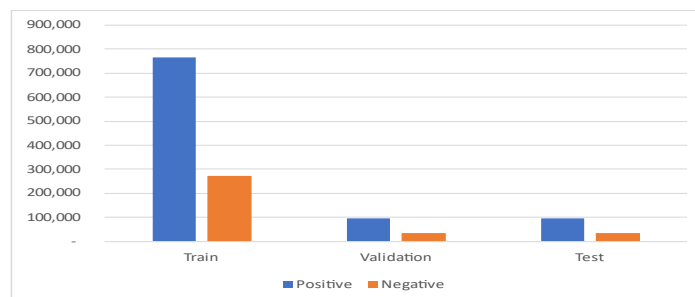


Fig. 3. The number of text reviews for training, validation, and test sets

Embedding layer takes the integer-encoded vocabulary and looks up the embedding vector for each word index. The embedding layer using training samples from the VN_E-commerce_Review dataset is trained rather than a pre-trained embedding word model such as word2vec [27] or GloVe [28]. In this study, we used an embedding layer given by Keras with a set of 32,526 unique vocabularies, with each word is embedded in a 30-dimension vector space.

Bi-LSTM layer is an extension of the LSTM models that combine Bi-RNN models and LSTM units to capture the context information. In the first round, an LSTM is applied on the input sequence (i.e., forward layer). In the second round (i.e., backward layer) of the LSTM model, the reverse form of the input sequence is fed into the LSTM model [29]. Using the LSTM twice improves the learning of long-term dependencies, which allows it to learn the context more efficiently. The architecture of Bi-LSTM is illustrated in Fig. 4.

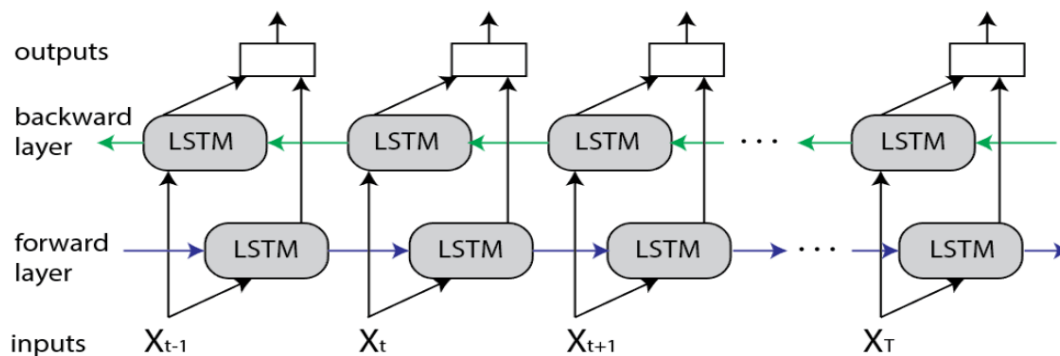


Fig. 4. The architecture of Bi-LSTM [30]

Sigmoid function is equivalent to a 2-element Softmax, with the second element being assumed to be zero. The output of the sigmoid function is always within a range between 0 to 1:

$$\text{sigmoid}(x) = \frac{1}{1+e^{-x}} \quad (1)$$

2.4. Hyper-parameter setting

Hyperparameter tuning is a crucial step in improving model performance. This method involves modeling the relationship between hyperparameters and the model's performance using a probabilistic model. The model is iteratively updated based on the performance of the evaluated hyperparameters, resulting in a more efficient search of the hyperparameter space. It can prevent overfitting or underfitting and selects the best model as the final model. In this study, we tune four different hyperparameters below and optimize the accuracy. After that, the Bayesian optimization function from Keras Tuner [30] for hyperparameter tuning is used. The objective of parameter is set to "val_accuracy" and the "max_trials" parameter is set to 50, which means that the optimization function will attempt to optimize the validation accuracy by trying a maximum of 50 different combinations of hyperparameters. Table 2 describes the best hyperparameter values in the proposed model.

2.5. Metrics

Confusion matrix is applied to evaluate the model. It is a matrix with the number of True Positives (TP), True Negatives (TN), False Negatives (FN), and False Positives (FP). The metrics that are most widely used for evaluation are described as follows:

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

$$\text{precision} = \frac{TP}{TP + FP} \quad (3)$$

$$\text{recall} = \frac{TP}{TP + FN} \quad (4)$$

$$F1 - score = \frac{2 * precision * recall}{precision + recall} \quad (5)$$

Table 2. Hyper-parameters setting

Parameters	Values	Search space
Embedding dimension	30	-
Filter	256	Values between 32 and 512 (inclusive) in steps of 32. These represent the number of units in a neural network layer
(LSTM) dropout	0.5	Values between 0.1 and 0.5 (inclusive) in steps of 0. These represent the fraction of input units to drop in a neural network layer.
Bi-LSTM output size	512	-
Activation	tanh	One of two values "tanh" or "relu". It represents the activation function to use in a neural network layer.
Batch size	256	-
Number of epoch	100	-
Batch normalization	Yes	-
Loss function	binary_crossentropy	-
Optimizer	Adam	-
Learning rate	0.002	Values between 1e-5 and 1e-1 (inclusive). We use logarithmic sampling to explore the search space that represents the learning rate of a neural network optimizer.

2.6. Building sentiment dictionary and extracting aspects by using POS tagging method

POS tagging stands for "Part-of-Speech tagging". It is a process in natural language processing that involves assigning a part of speech (such as noun, verb, adjective, etc.) to each word in a sentence. The process involves analyzing the grammatical structure of the sentence and identifying the role of each word in the sentence. In this study, to apply the technique of labeling from type, we used the pyvi library [31], which is considered one of the libraries with the best results for natural language processing in the Vietnamese language, with a F1 - score of 0.925.

2.7. Building sentiment dictionary

Based on using the POS tagging method and the results of model classification, we built a dictionary to classify positive and negative nuances. The dictionary was built by filtering adjective words in the dataset and determining their frequency of occurrence with respect to the rating score to assign them to either the positive or negative group. If a word's frequency of occurrence in one group is more than the other, it will be classified into that respective group. Table 3 shows some examples of words classified as positive or negative.

Table 3. Example of nuance words (Source: Authors)

word (in Vietnamese and English)	pos_tag	negative	positive
tốt <i>good</i>	A	4.68%	95.32%
nhiều <i>multiple</i>	A	41.37%	58.63%
tuyệt_vời <i>excellent</i>	A	1.41%	98.59%
nhanh <i>fast</i>	A	12.43%	87.57%
rẻ <i>cheap</i>	A	16.69%	83.31%
ok <i>okay</i>	A	6.43%	93.57%
tuyệt <i>great</i>	A	1.87%	98.13%
cao <i>high</i>	A	58.91%	41.09%
đúng <i>true</i>	A	46.85%	53.15%
tê <i>bad</i>	A	96.08%	3.92%
tiện_lợi <i>convenient</i>	A	2.94%	97.06%
hài_lòng <i>happy</i>	A	5.96%	94.04%

2.7.1. Aspects Extraction

The aspects by filtering the most frequently occurring noun words in the dataset were extracted that are related to segments such as goods, price, customer care, and customer experience UX/UI, etc. The aspect-based sentiment extraction results are presented in Table 4.

Table 4. Example of aspect based on nuance words (Source: Authors)

Word (in Vietnamese and English)	pos_tag
hàng <i>goods</i>	N
người <i>people</i>	N
sản_phẩm <i>product</i>	N
cửa_hàng <i>shop</i>	N
tiền <i>money</i>	N
ứng_dụng <i>app</i>	N
chất_lượng <i>quality</i>	N
hàng <i>order</i>	N
phí <i>cost</i>	N
giá <i>price</i>	N

3. Results and Discussion

3.1. Experimental setup

Colaboratory Pro which is released by Google for all data pre-processing, training, and testing: 24GB of Reading Access Memory (RAM), GPU Nvidia Tesla P100-PCIe-16GB, and Intel(R) Xeon(R) CPU @ 2.30GHz. To train the model, the binary cross-entropy [32] for the loss function is applied:

$$\Lambda\sigma\sigma = -\sum_{c=1}^M y_{o,c} \log(p_{o,c}), \quad (6)$$

In the given equation, M represents the total number of classes, \log refers to the natural logarithm, $y_{o,c}$ is a binary indicator taking values of either 0 or 1, representing whether class label c is the correct classification for observation o , and $p_{o,c}$ is the predicted probability that observation o belongs to class c . For the optimizers, the optimal Adam method [33] with learning rate equal 0.002 is applied. The number of epochs is set to 100 with EarlyStopping to monitor the metric `val_accuracy`, and patience equals 15. This means stopping training when the monitored metric has stopped improving after 15 epochs in a row. Fig. 5 displays the accuracy and loss training histories.

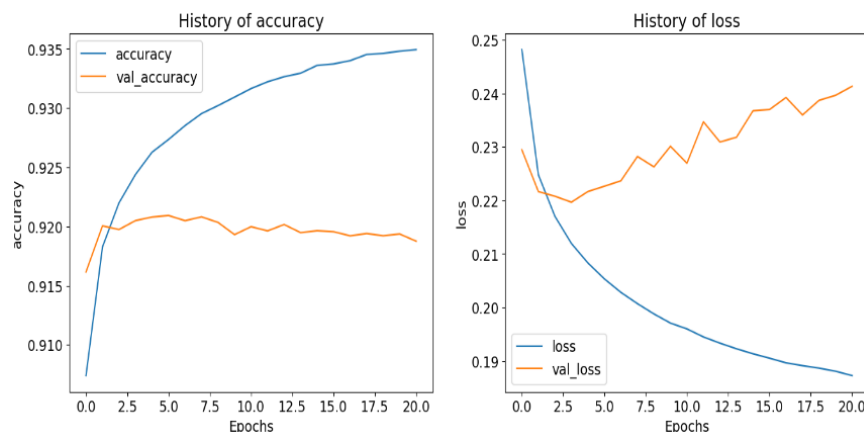


Fig. 5. The training history of the accuracy and the loss (Source: Authors)

3.2. Experimental results

After training, the test data set is used to assess the performance of the proposed model. Fig. 6 shows the result in confusion matrix.

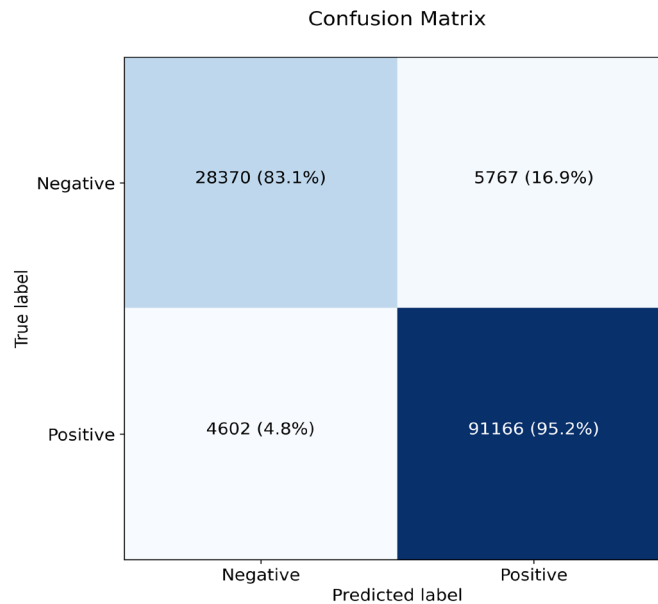


Fig. 6. The confusion matrix (Source: Authors)

Table 5 illustrates the accuracy, precision, recall, and F1-score values for the proposed technique in comparison to conventional machine learning, as well as the assessment findings for both before and after pre-processing.

Table 5. Performance Comparison of Bi-LSTM Model (Source: Authors)

Model	Original				After pre-processing			
	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Naïve Bayes	88.78	92.59	92.15	92.37	89.15	92.65	92.63	92.64
SVM	90.28	92.3	94.72	93.49	90.53	92.55	94.78	93.65
Logistic regression	90.49	92.37	94.95	93.64	90.75	92.61	95.04	93.81
LSTM	91.81	93.69	95.31	94.49	91.95	93.94	95.22	94.58
Bi-LSTM	91.73	93.99	94.85	94.42	92.01	94.07	95.17	94.61

After obtaining the results, the Bi-LSTM method is the most suitable for the dataset used in this study, with an accuracy of 92.01% and F-score of 94.61% after pre-processing step, which are higher than the remaining methods and before pre-processing.

3.3. Business application

After implementing the labeling model from categories using the POS tagging method, Fig. 7 show the analysis of positive and negative sentiment trends in detail, including the total number of comments in the nine years from 2013 to 2022 covered by the corpus, with the Lazada application having the highest total number of comments (530,055), of which there were 410,021 (accounting for 77.35%) positive comments (sentiment = 1.0) and 120,034 (22.65%) negative comments (sentiment = 0.0). The Shopee application ranked second with a total of 518,716 comments, with 352,094 positive (67.88%) and 166,622 negative comments (32.12%). In third place, the Sendo application had a total of 179,321 comments, with 143,170 positive (79.74%) and 36,368 (20.26%) negative comments. Lastly, Tiki

application had a total of 70,602 comments, with 52,310 positive (74.09%) and 18,292 negative comments (25.91%).

Fig. 7 shows that most applications have a decreasing trend of a positive proportion and have an increasing negative proportion of comments from 2019 to 08/2022.

Appid	Target	2013	2014	2015	2016	2017	At 2018	2019	2020	2021	2022	Grand Total
lazada	0	23	187	1,706	3,404	5,227	12,297	17,585	18,811	42,110	18,684	120,034
	1	29.87%	19.50%	18.64%	24.97%	24.70%	27.89%	27.89%	31.80%	31.68%	37.64%	110,021
shopee	0	54	772	7,446	10,231	15,933	31,802	124,374	91,683	90,083	37,643	77,359
	1	70.13%	80.50%	81.36%	75.03%	75.30%	72.12%	87.61%	82.98%	68.15%	66.83%	166,622
sendo	0			339	1,577	4,478	17,037	26,394	25,387	48,081	43,329	32,129
	1			18.72%	39.60%	40.73%	51.60%	30.33%	20.92%	31.17%	40.77%	352,094
tiki	0			1,472	2,405	6,516	15,982	60,617	95,980	106,170	62,953	67,889
	1			81.28%	60.40%	59.27%	48.40%	69.67%	79.08%	68.83%	59.23%	36,368
lazada	0			405	1,742	2,862	6,650	14,416	5,779	3,520	994	20,269
	1			15.79%	26.12%	23.05%	22.03%	17.63%	21.81%	22.85%	24.76%	143,170
shopee	0			2,160	4,926	9,555	23,537	67,372	20,714	11,885	3,021	79,749
	1			84.21%	73.88%	76.95%	77.97%	82.37%	78.19%	77.15%	75.24%	18,292
sendo	0			105	445	767	2,074	5,318	4,454	3,807	1,322	25,919
	1			38.46%	27.28%	50.26%	20.18%	17.47%	31.50%	38.71%	53.18%	52,310
tiki	0			168	1,186	759	8,203	25,115	9,687	6,028	1,164	74,099
	1			61.54%	72.72%	49.74%	79.82%	82.53%	68.50%	61.29%	46.82%	

Fig. 7. The total numbers of positive and negative comments on the four applications from 06/2013 to 08/2022

Dive into more detail, Fig. 8 presents the top 10 words with positive nuances in the period (01/2018 - 08/2022). "Good" was the word most appreciated by users on all four applications, with Lazada having the most comments from 2018-2021. Followed by words such as "multiple", "excellent", "fast", "cheap", "okay", "great", "true", "better", and "happy".

Word	2018				2019				At / Appid 2020				2021				2022			
	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki
tốt (good)	8,543	7,261	3,516	2,115	35,475	21,172	13,899	6,792	24,432	6,272	22,009	2,731	23,391	3,587	23,268	1,681	9,771	928	14,099	275
nhieu (multiple)	1,939	1,337	2,100	445	4,301	3,146	5,820	1,155	4,091	1,222	8,217	734	7,417	693	10,322	395	3,636	171	6,927	148
tuyệt_vời (excellent)	909	630	556	283	5,022	2,989	3,958	1,192	4,262	945	6,395	335	3,397	557	6,670	166	1,502	165	4,113	39
nhANH (fast)	1,376	1,118	970	682	4,464	2,428	2,591	2,109	3,057	968	4,626	1,114	3,521	410	4,547	739	2,457	115	2,858	169
rẻ (cheap)	1,260	873	1,125	202	2,729	1,642	2,217	526	1,904	529	3,137	267	2,661	283	3,482	167	1,033	63	2,088	49
ok (okay)	661	495	340	109	2,597	1,456	1,415	486	2,294	472	2,950	269	2,623	331	3,391	206	1,495	82	1,708	24
tuyệt (great)	626	476	343	175	3,031	1,572	1,868	667	2,228	392	3,023	157	1,534	168	3,124	66	491	47	1,897	19
đúng (true)	755	656	722	178	2,028	1,385	1,147	551	1,896	536	1,778	397	2,329	313	2,056	288	1,463	64	1,333	68
hơn (better)	978	535	1,073	184	1,147	934	1,547	533	1,014	319	1,930	343	1,525	235	2,436	313	771	66	1,468	57
hài_lòng (happy)	766	447	341	260	1,942	1,079	962	1,054	1,497	409	1,758	334	2,072	211	1,711	204	1,141	62	802	15

Fig. 8. Top 10 words with positive nuances in the period (01/2018 - 08/2022)

The template is designed so that author affiliations are not repeated each time for multiple authors of the same affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization). This template was designed for two affiliations.

Fig. 9 shows the top 10 negative words in the period (01/2018 - 08/2022). In 2022, for the Shopee app, the negative words to care about were "bad", "long", "different" and "slow" with more than 1000 occurrences more than the rest of the top 10 words. For the Lazada, words with an occurrence number above 1000 were "bad". The negative words that Tiki needs to pay attention to include "bad", "long".

Word	2018				2019				At / Appid 2020				2021				2022			
	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki
cao (high)	3,144	696	2,017	59	2,321	2,039	1,739	196	1,015	403	1,936	203	945	285	1,959	156	573	56	926	35
tệ (bad)	460	229	904	103	622	620	1,633	421	1,386	329	1,697	361	2,505	247	3,721	490	1,150	74	3,039	153
khác (different)	574	362	729	154	862	730	1,375	525	923	332	1,718	362	1,832	205	2,723	276	933	66	1,671	78
lâu (long)	469	373	761	84	878	632	1,082	457	824	224	1,301	261	1,420	207	3,376	309	883	75	2,443	110
chậm (slow)	347	418	1,325	124	643	585	1,915	360	612	207	1,973	237	819	145	2,608	305	496	75	2,380	58
kém (less)	363	219	420	36	746	516	722	185	987	216	831	178	1,250	139	1,201	150	406	25	678	42
đắt (expensive)	1,100	200	1,174	57	663	451	902	155	329	92	727	137	380	75	1,141	124	255	28	596	56
ít (little)	186	150	184	94	301	280	392	187	283	102	518	99	609	77	892	64	377	30	624	43
gần (near)	260	79	362	45	279	174	476	171	259	63	555	106	451	41	971	137	226	13	580	29
khó (difficult)	167	155	195	31	285	233	420	85	392	141	569	73	744	105	588	65	350	21	315	32

Fig. 9. Top 10 negative words in the period (01/2018 - 08/2022)

In terms of words with a nuance-related aspect, Fig. 10 shows the top 10 words in the period (01/2018 - 08/2022) for the four applications including "app", "order", "cost", "customer", "error", "voucher", "advertising", "phone", "time", and "pr*ck". Based on the 2022 data, the positive and negative related aspects that all four applications need to care about are as follows "app", which includes feedback regarding the use of the application, ease of use, access speed, and the response on applications. This aspect group had the highest occurrence frequency in the top 10 aspect-related words.

Word	2018				2019				At / Appid 2020				2021				2022			
	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki	lazada	sendo	shopee	tiki
ủng_dụng (app)	883	754	1,433	381	1,927	1,856	3,576	831	2,319	745	4,271	595	4,304	505	5,737	483	2,021	129	3,895	136
đơn (order)	924	700	1,690	245	1,165	1,235	1,856	846	1,373	644	2,300	492	2,862	498	5,220	755	1,733	142	3,987	198
phi (cost)	3,670	808	2,967	83	2,618	1,853	2,345	249	1,210	360	2,469	211	1,154	223	2,245	152	666	45	1,064	29
khách_hàng (customer)	651	492	623	121	1,266	1,052	1,385	458	1,468	583	2,221	379	2,572	371	3,178	485	1,117	100	1,732	132
lỗi (error)	450	185	901	225	447	367	1,712	329	502	207	1,908	264	1,184	94	3,409	191	669	26	5,976	70
mã (voucher)	190	108	630	86	296	529	1,382	140	397	224	2,003	153	1,025	153	2,887	125	648	29	3,137	54
quảng_cáo (advertising)	202	182	437	51	495	450	780	130	742	178	919	151	3,969	111	2,193	75	1,690	35	1,352	20
điện_thoại (phone)	271	357	1,319	104	401	699	1,449	223	475	267	1,624	186	986	161	2,379	138	469	26	1,897	46
giờ (time)	682	188	575	64	512	610	930	220	516	209	1,042	197	1,012	97	1,713	205	524	25	1,171	53
c* (pr*ck)	246	100	335	49	521	282	738	115	637	109	651	111	1,556	64	1,774	80	977	21	1,909	34

Fig. 10. Top 10 aspects related to positive and negative nuances in the period (01/2018 – 08/2022)

3.4. Discussion

The experimental results have clarified the research model in Fig. 1 for the analysis of positive and negative aspects and nuances of customers toward mobile applications of commercial shops through feedback, comments, and reviews. In particular, when analyzing aspects related to negative nuances, the results show that the majority of opinions focus on six main issues related to the shops' products and services: (1) product quality is not as advertised; (2) the process of returning products is difficult and time consuming such that customers do not know when they will be refunded; (3) the mobile application is difficult to use; (4) when there is a response, the customer calls the switchboard and waits for a long time; (5) service attitude errors of staff, as there are promises that are not fulfilled; (6) the transportation of goods causes damage and long waiting times.

Based on customer feedback, the study also analyzed and found six solutions that can be recommended for shops to improve product and service quality based on the positive aspects and nuances deduced from the customers' opinions: (1) reminding about and taking proactive measures against stalls in stores; (2) updating customers on a regular basis about the product return process, what stage the return is in, and when to refund and through which channels; (3) providing specific instructions for customers via video, images, or directly on the mobile application when the customer makes a return or complaint; (4) preparing staffing plans during peak hours or promotional campaign days to always have sufficient numbers of customer support staff. In the future, stores should develop a system of call centers that automatically receive and answer calls or build AI chatbots to support customers in a timely manner; (5) provide continuous reviews of quality monitoring, listening to call recordings until the end, and viewing replies to emails and messages. Moreover, shops should identify common mistakes and provide training sessions for employees to follow the correct process, learn how to control emotions, and help them better understand customer wishes; (6) continuous checks of the shipping process, stowage, and the shipper's performance through customer reviews. In addition, there should be detailed instructions on how to pack items so that stores selling on mobile applications can properly pack products.

4. Conclusion

Based on the research and analysis results from our dataset of customer opinions about e-commerce products and services from the four largest mobile commerce applications in Vietnam (i.e., Lazada, Shopee, Sendo, and Tiki) on the Google Play Store from 01/2013 to 08/2022, the experimental results demonstrate that the proposed Bi-LSTM model achieved the best model among all existing models, with a high F1 score of 94.61%. The result of the Bi-LSTM model can be attributed to its ability to learn long-term dependencies and understand the context of the text, which is crucial in identifying sentiment. Additionally, the study obtained research results of scientific and practical significance with the following contributions: (1) providing a dataset of customer reviews specific to the Vietnamese e-commerce industry; (2) developing dictionaries and techniques to address pre-processing issues with Vietnamese text; (3) proposing a hybrid model for classifying sentiment and extracting aspects of products and services from reviews. These results have significant implications for both researchers and businesses. For instance, the model can be employed to build dashboards to monitor online customer

experiences in real-time, assist administrators in making timely decisions, enhance the quality of products and services, and improve customer satisfaction.

The study has some limitations that could be addressed in future research. The model focuses on considering the positive and negative nuances and did not analyze neutral nuances. To expand the model's capabilities: (1) planning to incorporate more neutral nuances in the Vietnamese language; (2) improving the feature extraction for the network. An integrated word embedding approach may produce better results; (3) using pre-trained models and applying them to large datasets.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

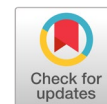
References

- [1] "The Map of E-commerce in Vietnam," *iprice*, 2019. Accessed Apr. 19. 2021. [Online]. Available at: <https://iprice.vn/insights/mapofecommerce/en/>.
- [2] B. Liu, M. Hu, and J. Cheng, "Opinion observer," in *Proceedings of the 14th international conference on World Wide Web - WWW '05*, 2005, p. 342, doi: [10.1145/1060745.1060797](https://doi.org/10.1145/1060745.1060797).
- [3] "Vietnamese language," *Britannica*. [Online]. Available at: <https://www.britannica.com/topic/Vietnamese-language>.
- [4] Y. Nurdiansyah, S. Bukhori, and R. Hidayat, "Sentiment analysis system for movie review in Bahasa Indonesia using naive bayes classifier method," *J. Phys. Conf. Ser.*, vol. 1008, no. 1, p. 012011, Apr. 2018, doi: [10.1088/1742-6596/1008/1/012011](https://doi.org/10.1088/1742-6596/1008/1/012011).
- [5] Y. Al Amrani, M. Lazaar, and K. E. El Kadiri, "Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis," *Procedia Comput. Sci.*, vol. 127, pp. 511-520, Jan. 2018, doi: [10.1016/j.procs.2018.01.150](https://doi.org/10.1016/j.procs.2018.01.150).
- [6] N. K. Bolbol and A. Y. Maghari, "Sentiment Analysis of Arabic Tweets Using Supervised Machine Learning," in *2020 International Conference on Promising Electronic Technologies (ICPET)*, Dec. 2020, pp. 89-93, doi: [10.1109/ICPET51420.2020.00025](https://doi.org/10.1109/ICPET51420.2020.00025).
- [7] B. Noori, "Classification of Customer Reviews Using Machine Learning Algorithms," *Appl. Artif. Intell.*, vol. 35, no. 8, pp. 567-588, Jul. 2021, doi: [10.1080/08839514.2021.1922843](https://doi.org/10.1080/08839514.2021.1922843).
- [8] M. Khan and K. Malik, "Sentiment Classification of Customer's Reviews About Automobiles in Roman Urdu," in *Advances in Intelligent Systems and Computing*, vol. 887, Springer Verlag, 2019, pp. 630-640, doi: [10.1007/978-3-030-03405-4_44](https://doi.org/10.1007/978-3-030-03405-4_44).
- [9] M. A. Qureshi *et al.*, "Sentiment Analysis of Reviews in Natural Language: Roman Urdu as a Case Study," *IEEE Access*, vol. 10, pp. 24945-24954, 2022, doi: [10.1109/ACCESS.2022.3150172](https://doi.org/10.1109/ACCESS.2022.3150172).
- [10] S. Goyal, "Review Paper on Sentiment Analysis of Twitter Data Using Text Mining and Hybrid Classification Approach," *Internatio nal J. Eng. Dev. Res.*, vol. 5, no. 2, pp. 2321-9939, 2017, [Online]. Available: <https://www.ijedr.org/papers/IJEDR1702032.pdf>.
- [11] A. Bayhaqy, S. Sfenrianto, K. Nainggolan, and E. R. Kaburuan, "Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes," in *2018 International Conference on Orange Technologies (ICOT)*, Oct. 2018, pp. 1-6, doi: [10.1109/ICOT.2018.8705796](https://doi.org/10.1109/ICOT.2018.8705796).
- [12] P. Cen, K. Zhang, and D. Zheng, "Sentiment Analysis Using Deep Learning Approach," *J. Artif. Intell.*, vol. 2, no. 1, pp. 17-27, Jul. 2020, doi: [10.32604/jai.2020.010132](https://doi.org/10.32604/jai.2020.010132).
- [13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735-1780, Nov. 1997, doi: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735).

- [14] Q. Lu, "Neural Network based Approaches for Aspect-Based Sentiment Analysis," *Highlights Sci. Eng. Technol.*, vol. 12, pp. 222–229, Aug. 2022, doi: [10.54097/hset.v12i.1457](https://doi.org/10.54097/hset.v12i.1457).
- [15] H. Li, Y. Ma, Z. Ma, and H. Zhu, "Weibo Text Sentiment Analysis Based on BERT and Deep Learning," *Appl. Sci.*, vol. 11, no. 22, p. 10774, Nov. 2021, doi: [10.3390/app112210774](https://doi.org/10.3390/app112210774).
- [16] N. S. Tun, N. N. Long, T. Tran, N. T. Thao, D. T. Thu Phuong, and T. Nguyen, "Stock article title sentiment-based classification using PhoBERT," in *CEUR Workshop Proceedings*, 2021, vol. 3026, pp. 225–233. [Online]. Available at: <https://ceur-ws.org/Vol-3026/paper25.pdf>.
- [17] Q. T. Nguyen, T. L. Nguyen, N. H. Luong, and Q. H. Ngo, "Fine-Tuning BERT for Sentiment Analysis of Vietnamese Reviews," in *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*, Nov. 2020, pp. 302–307, doi: [10.1109/NICS51282.2020.9335899](https://doi.org/10.1109/NICS51282.2020.9335899).
- [18] T.-L. Truong, H.-L. Le, and T.-P. Le-Dang, "Sentiment Analysis Implementing BERT-based Pre-trained Language Model for Vietnamese," in *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*, Nov. 2020, pp. 362–367, doi: [10.1109/NICS51282.2020.9335912](https://doi.org/10.1109/NICS51282.2020.9335912).
- [19] W. Zhang, X. Li, Y. Deng, L. Bing, and W. Lam, "A Survey on Aspect-Based Sentiment Analysis: Tasks, Methods, and Challenges," in *IEEE Transactions on Knowledge and Data Engineering*, 2022, pp. 1–20, doi: [10.1109/TKDE.2022.3230975](https://doi.org/10.1109/TKDE.2022.3230975).
- [20] A. S. Shafie, N. M. Sharef, M. A. Azmi Murad, and A. Azman, "Aspect Extraction Performance with POS Tag Pattern of Dependency Relation in Aspect-based Sentiment Analysis," in *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*, Mar. 2018, pp. 1–6, doi: [10.1109/INFRKM.2018.8464692](https://doi.org/10.1109/INFRKM.2018.8464692).
- [21] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Aug. 2004, pp. 168–177, doi: [10.1145/1014052.1014073](https://doi.org/10.1145/1014052.1014073).
- [22] T. A. Rana and Y.-N. Cheah, "Exploiting sequential patterns to detect objective aspects from online reviews," in *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, Aug. 2016, pp. 1–5, doi: [10.1109/ICAICTA.2016.7803101](https://doi.org/10.1109/ICAICTA.2016.7803101).
- [23] K. Srividya and A. M. S. Sowjanya, "Aspect Based Sentiment Analysis using POS Tagging and TFIDF," *Int. J. Eng. Adv. Technol.*, vol. 8, no. 6, pp. 1960–1963, Aug. 2019, doi: [10.35940/ijeat.F7935.088619](https://doi.org/10.35940/ijeat.F7935.088619).
- [24] L. Mai and B. Le, "Aspect-Based Sentiment Analysis of Vietnamese Texts with Deep Learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10751 LNAI, Springer Verlag, 2018, pp. 149–158, doi: [10.1007/978-3-319-75417-8_14](https://doi.org/10.1007/978-3-319-75417-8_14).
- [25] L. Luc Phan *et al.*, "SA2SL: From Aspect-Based Sentiment Analysis to Social Listening System for Business Intelligence," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12816 LNAI, Springer Science and Business Media Deutschland GmbH, 2021, pp. 647–658, doi: [10.1007/978-3-030-82147-0_53](https://doi.org/10.1007/978-3-030-82147-0_53).
- [26] B. Liu, "Many Facets of Sentiment Analysis," in *A Partical Guide to Sentiment Analysis*, Springer, Cham, 2017, pp. 11–39, doi: [10.1007/978-3-319-55394-8_2](https://doi.org/10.1007/978-3-319-55394-8_2).
- [27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, Jan. 2013, pp. 1–12, doi: [10.48550/arXiv.1301.3781](https://doi.org/10.48550/arXiv.1301.3781).
- [28] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543, doi: [10.3115/v1/D14-1162](https://doi.org/10.3115/v1/D14-1162).
- [29] M. M. Rahman, Y. Watanobe, and K. Nakamura, "A Bidirectional LSTM Language Model for Code Evaluation and Repair," *Symmetry (Basel)*, vol. 13, no. 2, p. 247, Feb. 2021, doi: [10.3390/sym13020247](https://doi.org/10.3390/sym13020247).
- [30] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, and L. Invernizzi, "A Hyperparameter Tuning Library for Keras," *GitHub*, 2019. Accessed Apr. 30, 2020. [Online]. Available at: <https://github.com/keras-team/keras-tuner>.

-
- [31] B.-T. Nguyen-Thi and H.-T. Duong, "A Vietnamese Sentiment Analysis System Based on Multiple Classifiers with Enhancing Lexicon Features," in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, vol. 293, Springer Verlag, 2019, pp. 240–249, doi: [10.1007/978-3-030-30149-1_20](https://doi.org/10.1007/978-3-030-30149-1_20).
- [32] D. R. Cox, "The Regression Analysis of Binary Sequences," *J. R. Stat. Soc. Ser. B*, vol. 20, no. 2, pp. 215–232, Jul. 1958, doi: [10.1111/j.2517-6161.1958.tb00292.x](https://doi.org/10.1111/j.2517-6161.1958.tb00292.x).
- [33] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, pp. 1–15, Dec. 22, 2014, doi: [10.48550/arXiv.1412.6980](https://doi.org/10.48550/arXiv.1412.6980).

Deep learning approaches for MIMO time-series analysis



Fachrul Kurniawan ^{a,1,*}, Sarina Sulaiman ^{b,2}, Siaka Konate ^{c,3}, Modawy Adam Ali Abdalla ^{d,e,4}

^a Informatics Engineering, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Kota Malang, Jawa Timur 65144, Indonesia

^b UTM Big Data Centre, Ibnu Sina Institute for Scientific and Industrial Research, Soft Computing Research Group, Universiti Teknologi Malaysia, Malaysia.

^c Department of Electronic and Telecommunications, Normal School of Technical and Vocational Education, 91094 Bamako, Mali

^d College of Energy and Electrical Engineering, Hohai University, Nanjing 211100, China

^e Department of Electrical and Electronic Engineering, College of Engineering Science, Nyala University, Nyala 63311, Sudan

¹ fachrulk@ti.uin-malang.ac.id; ² sarina@utm.my; ³ konatesiaka77@gmail.com; ⁴ brojacter88@yahoo.com

* corresponding author

ARTICLE INFO

Article history

Received April 18, 2023

Revised May 31, 2023

Accepted June 6, 2023

Available online June 13, 2023

Keywords

MIMO

Time series

Deep learning

Stock prices

ABSTRACT

This study presents a comparative analysis of various deep learning (DL) methods for multi-input and multi-output (MIMO) time-series forecasting of stock prices. The analysis is conducted on a dataset comprising the stock price of Bitcoin. The dataset consists of 2950 rows from December 2017 to December 2021. This study aims to evaluate the performance of multiple DL methods, including Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), and Gated Recurrent Unit (GRU). The evaluation criteria for selecting the best-performing methods in this research are based on two performance metrics: Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE). These metrics were chosen for specific reasons related to assessing the accuracy and reliability of the forecasting models. MAPE is used to assess accuracy, while RMSE helps detect outliers in the system. Results show that the LSTM method achieves the best performance, outperforming other methods with an average MAPE value of 8.73% and Bi-LSTM has the best average RMSE value of 0.02216. The findings of this study have practical implications for time-series forecasting in the field of stock trading. The superior performance of LSTM highlights its potential as a reliable method for accurately predicting stock prices. The Bi-LSTM model's ability to detect outliers can aid in identifying abnormal stock market behavior. In summary, this research provides insights into the performance of various DL models of MIMO for stock price forecasting. The results contribute to the field of time-series forecasting and offer valuable guidance for decision-making in stock trading by identifying the most effective methods for predicting stock prices accurately and detecting unusual market behavior.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Time series analysis involves the examination and prediction of data that is collected sequentially over time. This field of study is crucial in various domains, including finance [1], economics [2], meteorology [3], and sales forecasting [4]. However, time series analysis poses several challenges that need to be addressed to ensure accurate and reliable predictions. These challenges can be categorized into two main areas: single-output forecasting and multi-output forecasting. In single-output forecasting, common

problems include handling changing trends [5], identifying and modeling seasonal patterns [6], dealing with outliers [7], addressing stationarity assumptions [8], managing autocorrelation, [9] and working with limited data [10]. On the other hand, multi-output forecasting introduces additional complexities such as high dimensionality and the need to account for interactions between multiple output variables [11]. The multi-output forecasting problem has its own challenges because changes in one variable or output can affect other variables or output [12]. Therefore, it is important to consider the interactions and interrelationships between variables when developing an effective multi-output forecasting model.

Multi-Input and Multi-Output (MIMO) time-series forecasting is useful for real-world multivariate data applications. Financial markets use past prices [8], trade volumes [13], market sentiment [14], and macroeconomic data [15] to anticipate stock prices. MIMO time-series forecasting also has other uses. Temperature [16], humidity [17], wind speed [18], and precipitation interact [19] in the weather forecast. MIMO forecasting predicts energy consumption [20], resource availability [21], and production outputs [22] in resource management and optimization. Due to its complexity and real-world applications, the problem of MIMO time-series forecasting has gained attention in recent years [23][24][25][26]. The difficulty in this problem lies in modeling the dependencies between the input and output variables, especially when they have different time resolutions and levels of noise. Traditional statistical and machine learning (ML) approaches have limitations in capturing the dynamics and interactions between variables [27][28]. Therefore, developing a robust and efficient solution for MIMO time-series forecasting for developing robust time series models and obtaining accurate predictions, which have significant implications for decision-making and planning in various fields is crucial.

MIMO Time-series forecasting techniques struggle to represent long-term dependencies [29]. Trends, cycles, and seasonality are long-term dependencies. Various solutions have been proposed to address this problem, including vector autoregression moving average (ARIMA) models [30], multilayer perceptron (MLP) [31], and dynamic Bayesian networks (DBNs) [32][33]. These methods have shown promising results in capturing the nonlinear dependencies between variables and handling missing values and noisy data. However, they still have limitations in modeling long-term dependencies and dealing with high-dimensional data [34]. Therefore, there is a need for more advanced and flexible models that can overcome these limitations and provide accurate and interpretable predictions. Deep learning (DL) models, such as Convolutional Neural Networks (CNNs) [35][36][37], Long Short-Term Memory Networks (LSTMs) [38][39][40], and Gated Recurrent Units (GRU) [41], have been proposed as a promising solution for MIMO time-series forecasting. Therefore, this problem remains an active area of research, and further investigations are needed to develop more efficient and interpretable models.

This paper aims to reveal the performance of five different deep learning approaches: CNN, RNN, LSTM, GRU, and Bidirectional LSTM (Bi-LSTM). Recurrent Neural Networks (RNNs), LSTMs, and GRU can simulate long-term dependencies [42]. Recurrent connections and memory cells allow these models to store and learn from prior observations and dependencies. These models can better anticipate long-term trends by adding memory processes. MIMO time-series forecasting uses high-dimensional data to predict multiple outputs from multiple inputs [43]. Computational, model, and dimensionality issues arise with high-dimensional data. CNNs can handle high-dimensional data [44]. CNNs find data patterns by extracting local and global characteristics from input sequences using convolutional algorithms, making suitable for time-series forecasting. These models can capture the complex temporal patterns and interactions between variables and provide accurate and robust predictions. Moreover, they can handle missing values, noisy data, and high-dimensional input and output variables [45][46]. Developing a DL model for MIMO time-series forecasting requires careful consideration of the model architecture, data preprocessing, and hyperparameter tuning. MLP as baseline of deep learning is used to compare the performance of developed models for stock prediction.

2. Method

2.1. Data Collection

The dataset used is 2950 rows of data from <https://www.cryptodatadownload.com/> from December 18, 2017, to December 31, 2021. The dataset consists of 7 attributes whose visualization can be seen in Fig. 1. The description of the dataset can be seen in Table 1.

Table 1. Dataset Describe

index	open	close	high	low	Volume BTC	Volume USDT	tradecount
count	2950	2950	2950	2950	2950	2950	2950
mean	18296.42	18315.04	18836.06	17672.97	54331.53	1126496122.00	817203.89
std	17653.74	17668.82	18172.63	17054.67	35003.00	1426290194.00	756509.22
min	3211.71	3211.72	3276.50	3156.26	1521.53	11770168.04	12417.00
25%	7099.74	7099.735	7295.18	6863.25	31580.60	246861465.70	255846.75
50%	9509.07	9507.64	9709.17	9236.61	45692.96	452651204.90	506273.00
75%	26440.43	26932.90	27477.50	25835.00	67269.47	1561412415.00	1200025.00
max	67525.82	67525.83	69000.00	66222.40	402201.67	13477694935.00	6331062.00

This study uses two attributes as target data: the open attribute and the close attribute. The open attribute is the stock price at the beginning of the trade opening, and the close attribute is the price at the end of the period. The open and close prices are beneficial for analyzing pattern trends in stock prices [47]. Both of these attributes are attributes that affect changing the pattern trend (pattern) that is generated for stock predictions. The difference between the open and close prices can provide insights into the intraday price movement, such as whether the stock experienced a positive or negative trend during the trading session [48]. These trends often indicate investor sentiment, market momentum, and trading strategies market participants employ.

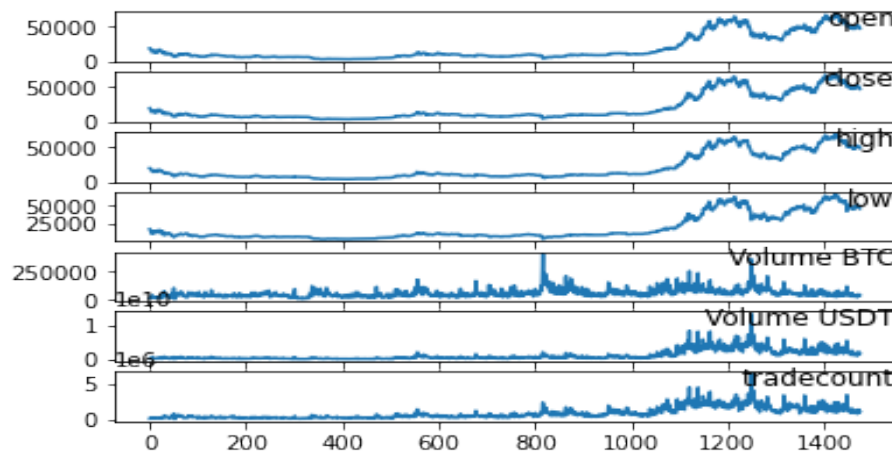


Fig. 1. Visualization of Dataset

Incorporating open and close prices as target variables enables the models to learn and capture these patterns, enhancing forecasting accuracy. Considering that factor, the selection of open and close prices as target variables for the forecasting task allows the models to leverage the intraday dynamics and patterns exhibited by these attributes. By incorporating this information into the training process, the models can learn to capture and utilize the trends and fluctuations in stock prices, leading to more accurate and reliable predictions. The comparison of the values between the close and open attributes can be seen in Fig. 2.

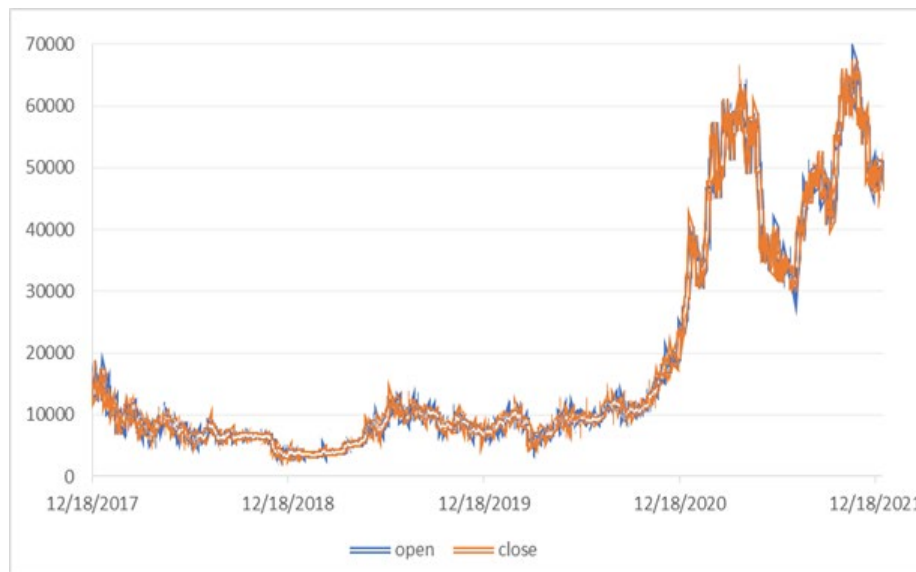


Fig. 2. Graphic Data Open and Close

2.2. Data Preprocessing

Min-max normalization is a common data preprocessing technique used in time-series analysis to scale the data values to a specific range, usually between 0 and 1, is achieved by subtracting the minimum value from each data point and dividing it by the range between the minimum and maximum values. The resulting values are then scaled to the desired range. Min-max normalization is well-suited for time-series analysis as it addresses time-dependent data's specific scaling and range normalization requirements [49]. It preserves temporal relationships, handles seasonal and trend components, mitigates the influence of outliers, provides an interpretable scale, and is compatible with various modeling techniques.

The first step is to identify the dataset's minimum and maximum values and apply min-max normalization to time-series data, can be done by iterating through each time step and keeping track of the minimum and maximum values [50]. Once these values have been identified, the data can be normalized using the equation (1).

$$x_{norm} = (x - min_val) / (max_val - min_val) \quad (1)$$

where x is the original data point, min_val is the minimum value in the dataset, max_val is the maximum value in the dataset, and x_{norm} is the normalized data point.

It is important to note that min-max normalization should be applied separately to the training and test datasets. The minimum and maximum values used for normalization should be based on the training dataset only, then applied to the test dataset using the same formula, ensuring that the test dataset is not used to inform the normalization process and prevents data leakage.

2.3. MIMO Forecasting

The forecasting model framework in the study can be seen in Fig. 3. Fig. 3 shows that the number of inputs is seven, and the number of outputs is 2. The forecasting process uses various types of methods that have been selected, as shown in Table 2. The first method is Convolutional Neural Networks (CNN), CNN are artificial neural networks that can be used for time-series forecasting. While CNNs were initially developed for image recognition tasks, they have also been applied to sequential data, including time-series data [51]. The main idea behind CNNs is to use filters that convolve over the input data to extract relevant features. These filters are typically small and move across the input data, computing dot products at each location. The outputs from the dot products are then passed through a nonlinear activation function, such as ReLU, and pooled to reduce the dimensionality of the data. This process is repeated multiple times in multiple layers, with each layer capturing more complex features of the input data. In the context of time-series forecasting, CNNs can be used to extract temporal features

from sequential data [52]. For example, the filters can convolve over the input time-series data to capture patterns such as trends, cycles, and spikes.

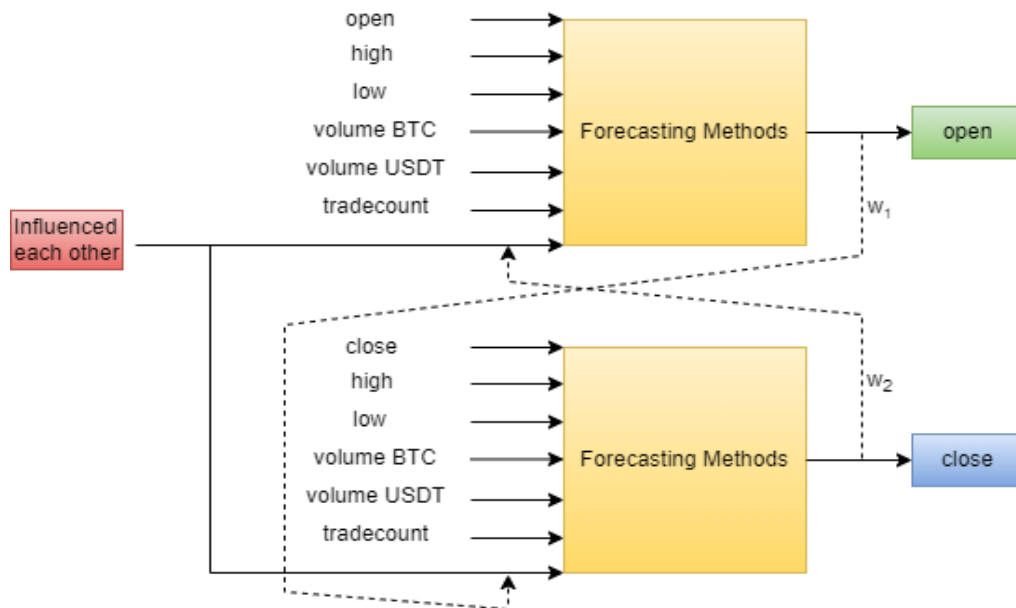


Fig. 3. MIMO Forecasting Scheme

The resulting feature maps can then be fed into fully connected layers to produce a forecast. One advantage of CNNs for time-series forecasting is their ability to capture local and global dependencies in the data. Additionally, CNNs can handle varying input lengths, making them useful for forecasting time-series data with irregular time intervals. The equation of CNN can be seen in equations (2) to (5).

$$H(t) = f(\text{conv}(X(t))) \tag{2}$$

$$F(t) = \text{flatten}(H(t)) \tag{3}$$

$$D(t) = g(W(f) * F(t) + b(f)) \tag{4}$$

$$Y(t) = D(t) \tag{5}$$

where $X(t)$ represents the input at time step t , $H(t)$ is the hidden state at time step t , $F(t)$ is flattened, $D(t)$ is a dense layer, and $Y(t)$ is the output. In the case of MIMO, the output is $Y(t)1$ and $Y(t)2$, which are depend on each other as in (6) and (7).

$$Y(t)1 = D(t) + Y(t)2 \tag{6}$$

$$Y(t)2 = D(t) + Y(t)1 \tag{7}$$

The second is Recurrent Neural Networks (RNN). Unlike traditional feedforward neural networks, which have no memory and process each input independently, RNNs have a memory component that allows them to process sequential data [53]. The key idea behind RNNs is the use of recurrent connections between nodes, which allow information to persist over time. In this way, the network can capture temporal dependencies in the data. RNNs use a hidden state that is updated at each time step, and the output at each time step is a function of the current input and the hidden state. The hidden state is passed from one-time step to the next, allowing the network to learn a representation of the entire sequence. In time-series forecasting, RNNs can predict the next value in a time series based on the previous values. The network is trained using a supervised learning algorithm, such as backpropagation through time, where the loss is minimized between the predicted and actual values of

the target variable [54]. One advantage of RNNs for time-series forecasting is their ability to capture long-term dependencies in the data, making them useful for predicting trends and cycles. The equation of RNN can be seen in equations (8) to (9).

$$H(t) = f(W(xh) * X(t) + W(hh) * H(t - 1) + b(h)) \quad (8)$$

$$Y(t) = g(W(hy) * H(t) + b(y)) \quad (9)$$

The output in the case of MIMO is $Y(t)1$ and $Y(t)2$, which depend on one another as in (10) and (11).

$$Y(t)1 = g(W(hy) * H(t) + b(y)) + Y(t)2 \quad (10)$$

$$Y(t)2 = g(W(hy) * H(t) + b(y)) + Y(t)1 \quad (11)$$

Third, Long Short-Term Memory (LSTM) is a type of RNN commonly used for time-series forecasting. LSTMs were designed to address the limitations of traditional RNNs, such as difficulty in capturing long-term dependencies and vanishing gradients [55]. The key idea behind LSTMs is using a memory cell that can remember information for long periods. Three gates control the memory cell: the input gate, the forget gate, and the output gate. These gates allow the network to update and forget information from the memory cell selectively. In the context of time-series forecasting, LSTMs can predict the next value in a time series based on the previous values [56]. The network is trained using a supervised learning algorithm, such as backpropagation through time, where the loss is minimized between the predicted and actual values of the target variable. One advantage of LSTMs for time-series forecasting is their ability to capture long-term dependencies in the data, making them useful for predicting trends and cycles [57]. Additionally, LSTMs can handle variable-length sequences, making them helpful in forecasting time-series data with irregular time intervals. The equation of LSTM can be seen in equations (12) to (17).

$$C(t) = f(W(xc) * X(t) + W(hc) * H(t - 1) + b(c)) \quad (12)$$

$$F(t) = \sigma(W(xf) * X(t) + W(hf) * H(t - 1) + b(f)) \quad (13)$$

$$I(t) = \sigma(W(xi) * X(t) + W(hi) * H(t - 1) + b(i)) \quad (14)$$

$$O(t) = \sigma(W(xo) * X(t) + W(ho) * H(t - 1) + b(o)) \quad (15)$$

$$H(t) = O(t) * \tanh(C(t)) \quad (16)$$

$$Y(t) = g(W(hy) * H(t) + b(y)) \quad (17)$$

In the MIMO situation, the outputs are the dependent variables $Y(t)1$ and $Y(t)2$ as in (18) and (19).

$$Y(t)1 = g(W(hy) * H(t) + b(y)) + Y(t)2 \quad (18)$$

$$Y(t)2 = g(W(hy) * H(t) + b(y)) + Y(t)1 \quad (19)$$

In these equations, $C(t)$, $F(t)$, $I(t)$, $O(t)$ sequentially represents the cell state, forget gate, input gate, and output gate of the LSTM. The variables W and b denote the learnable weights and biases of the model, respectively. The activation functions f and g represent the non-linear activation functions applied to the hidden state and output, respectively. σ represents the sigmoid activation function, and \tanh represents the hyperbolic tangent activation function.

The fourth is Bidirectional Long Short-Term Memory (Bi-LSTM) is an extension of the LSTM architecture for time-series forecasting. As the name suggests, Bi-LSTMs involve processing the input sequence in both forward and backward directions [58]. In a standard LSTM, the output at each time

step is a function of the current input and the hidden state from the previous time step. In contrast, in a Bi-LSTM, the output at each time step is a function of the current input and the hidden states from both the forward and backward directions. This allows the network to capture dependencies not only from the past, but also from the future. The key advantage of Bi-LSTMs is their ability to capture both past and future dependencies in the data, which can be especially useful for time-series forecasting tasks where future information may be useful for predicting the next value in the sequence [59]. In the context of time-series forecasting, Bi-LSTMs can be used to predict the next value in a time series based on the previous values in both the forward and backward directions. The network is trained using a supervised learning algorithm, such as backpropagation through time, where the loss is minimized between the predicted and actual values of the target variable. (20) to (23).

$$H_f(t) = LSTM_f(X(t)) \quad (20)$$

$$H_b(t) = LSTM_b(X(t)) \quad (21)$$

$$H(t) = [H_f(t); H_b(t)] \quad (22)$$

$$Y(t) = g(W(hy) * H(t) + b(y)) \quad (23)$$

In the case of MIMO, the results are the dependent variables $Y(t)1$ and $Y(t)2$ as in (24) and (25), respectively.

$$Y(t)1 = g(W(hy) * H(t) + b(y)) + Y(t)2 \quad (24)$$

$$Y(t)2 = g(W(hy) * H(t) + b(y)) + Y(t)1 \quad (25)$$

where $H_f(t)$ is forward LSTM, $H_b(t)$ is backward LSTM, and $H(t)$ is concatenation of forward and backward LSTM.

The last is Gated Recurrent Unit (GRU). GRUs were designed to address the limitations of traditional RNNs, such as difficulty in capturing long-term dependencies and vanishing gradients [60]. GRUs are similar to LSTMs in that they use a memory cell to remember information for long periods of time. However, unlike LSTMs, GRUs use only two gates: the reset and update gates. The reset gate determines how much of the previous memory to forget, while the update gate determines how much of the current input to remember. GRUs can be used to predict the next value in a time series based on the previous values. The network is trained using a supervised learning algorithm, such as backpropagation through time, where the loss is minimized between the predicted and actual values of the target variable. One advantage of GRUs for time-series forecasting is their ability to capture long-term dependencies in the data while requiring fewer parameters than LSTMs [61]. Additionally, GRUs can handle variable-length sequences, making them useful for forecasting time-series data with irregular time intervals.

$$R(t) = \sigma(W(xr) * X(t) + W(hr) * H(t - 1) + b(r)) \quad (26)$$

$$Z(t) = \sigma(W(xz) * X(t) + W(hz) * H(t - 1) + b(z)) \quad (27)$$

$$H(t) = (1 - Z(t)) * H(t - 1) + Z(t) * \tanh(W(xh) * X(t) + R(t) * (W(hh) * H(t - 1) + b(h))) \quad (28)$$

$$Y(t) = g(W(hy) * H(t) + b(y)) \quad (29)$$

Both $Y(t)1$ and $Y(t)2$ are dependent on time in the case of MIMO, as shown in (30) and (31).

$$Y(t)1 = g(W(hy) * H(t) + b(y)) + Y(t)2 \quad (30)$$

$$Y(t)2 = g(W(hy) * H(t) + b(y)) + Y(t)1 \quad (31)$$

where $R(t)$ is reset gate and $Z(t)$ is update gate.

These equations capture the MIMO deep learning models' forward pass for time-series forecasting. The models are trained by optimizing the weights and biases to minimize the forecasting error through backpropagation and gradient descent techniques.

According to the background, this paper establishes the method to create an effective deep learning-based for fundamental trading in Bitcoin stock price. Six forecasting methods, including MLP, CNN, RNN, LSTM, Bi-LSTM, and GRU were selected and analyzed to determine the best method based on accuracy in forecasting the future price. Hyperparameter tuning is used to determine the parameter setting values of various existing methods [62]. The hyperparameter tuning method used is random search. The setting parameter for all method can be seen in Table 2.

Table 2. Setting Parameter

Method	Hidden Laves	Neuron	Activation Function	Dropout	Optimizer	Loss	Epoch	Batch Size
MLP	2	32	Sigmoid	0.2	Adam	MSE	100	64
CNN	2	32	ReLU	0.2	Adam	MSE	100	64
RNN	2	32	Tanh	0.2	Adam	MSE	100	64
LSTM	2	32	Tanh	0.2	Adam	MSE	100	64
Bi-LSTM	2	32	Tanh	0.2	Adam	MSE	100	64
GRU	2	32	Tanh	0.2	Adam	MSE	100	64

Table 2 hyperparameter choices were chosen after thoroughly assessing their importance and projected influence on model performance. The following is a complete explanation of these parameters [63]. Hidden layers determine neural network depth and complexity. Although overfitting may grow, the model can capture more intricate interactions with hidden layers. The model's ability to learn and represent complicated patterns depends on hidden layer neuron counts. More neurons can grasp more complex data correlations, although overfitting may rise. The activation function gives the neural network non-linearity to describe complicated input-output interactions. Non-saturation, smoothness, and gradient propagation differ among activation functions. The batch size determines the number of samples processed before updating the model's weights during training. A smaller batch size updates weights more frequently but may be noisy, whereas a bigger batch size may take more memory but update more smoothly. Training epochs determine how many times the model processes the dataset. If not regularised, adding epochs lets the model learn more from the data but risks overfitting. Each of these hyperparameters plays a crucial role in determining the model's performance and behavior.

2.4. Evaluation

The evaluation process uses several forecasting methods to determine the performance of the Bitcoin stock price prediction. The evaluation performance uses Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE). MAPE is used to show errors that can represent accuracy. MAPE has several ranges of value-meaning categories, as shown in Table 3 [64]. RMSE is used to detect outliers in the designed system. The equation of MAPE and RMSE can be seen in (32) to (33).

$$MAPE = \frac{1}{N} \sum_{t=1}^N \frac{|xt - \hat{xt}|}{xt} \times 100\% \tag{32}$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^N (xt - \hat{xt})^2} \tag{33}$$

Table 3. MAPE Category

Range MAPE	Description
< 10%	The ability of the forecasting model is very good
10% – 20%	The ability of the forecasting model is good
20% – 50%	The ability of the forecasting model is feasible
> 50%	Poor forecasting model capability

3. Results and Discussion

The performance comparison of different methods for MIMO time-series forecasting can be seen in Table 4. The MLP method achieved a relatively low MAPE and RMSE for both open and close prices, with a MAPE of 9.56258% for open prices and 9.77263% for close prices, and an RMSE of 0.04473 for open prices and 0.04093 for close prices. This suggests that the MLP method performed well in predicting the open and close prices of the time-series data. Overall, the MLP method performed well in predicting the open and close prices of the time series, suggesting that MLPs can be a suitable choice for time-series forecasting tasks.

Table 4. Performance Evaluation Result

Methods	Open		Close	
	MAPE (%)	RMSE	MAPE (%)	RMSE
MLP	9.56258	0.04473	9.77263	0.04093
CNN	10.61802	0.09547	10.75626	0.06094
RNN	9.14382	0.04004	8.68090	0.04764
LSTM	8.79643	0.05272	8.67726	0.05097
Bi-LSTM	9.62744	0.02013	9.63486	0.02419
GRU	9.20339	0.02520	8.73113	0.03806

The CNN method achieved a MAPE of 10.61802%, an RMSE of 0.09547 for the Open, and a MAPE of 10.75626% and an RMSE of 0.06094 for the Close, indicating that the CNN method performed better than the ARIMA model but outperformed the MLP, RNN, LSTM, and GRU methods. CNN may not perform well on time-series data with long-term dependencies, as the filters may not be able to capture the full range of temporal patterns in the data. However, CNN also requires more training data to produce good MAPE values. In this study, the 1475 data processed using CNN could still not produce optimal values. The performance of the CNN method suggests that it may be useful for some time-series forecasting tasks but may not be optimal for all applications.

Based on Table 4, the RNN method achieved a MAPE of 9.14382%, an RMSE of 0.04004 for the open price prediction, a MAPE of 8.68090%, and an RMSE of 0.04764 for the close price prediction, this indicates that the RNN method performed relatively well compared to the other methods regarding MAPE and RMSE values, especially for the open price prediction. A lower MAPE and RMSE values indicate that the RNN model was able to make more accurate predictions of the stock prices based on the previous values in the time series. It is important to note that the performance of the RNN method may vary depending on the specific data and problem being addressed. However, RNNs may have difficulty with vanishing gradients, where the gradients become very small, and the network cannot learn long-term dependencies.

The LSTM model achieved a MAPE of 8.79643% for the Open and 8.67726% for the Close. The corresponding RMSE values were 0.05272 for the Open and 0.05097 for the Close. LSTM shows the lowest MAPE for both opening and closing prices. A lower MAPE indicates better performance of the model. In this case, the LSTM model achieved a lower MAPE than all other models except for RNN, indicating its superior performance in predicting stock prices. Overall, the LSTM model demonstrated strong performance in forecasting stock prices based on historical data. Its ability to capture long-term dependencies in the time-series data and remember information for long periods likely contributed to its superior performance.

Table 4 shows the Bi-LSTM method achieved a MAPE of 9.62744%, an RMSE of 0.02013 for the Open, and a MAPE of 9.63486% and an RMSE of 0.02419 for the Close. The results suggest that Bi-LSTM performed well compared to the ARIMA method, which had the highest MAPE and RMSE values among all the methods. However, Bi-LSTM's performance was slightly lower than the RNN and

LSTM methods, which achieved lower MAPE and RMSE values. The Bi-LSTM shows the lowest RMSE for both open and close prices, indicating its ability to detect outliers in the result predictions. Bi-LSTM can capture a sequence's past and future context at each time step. Overall, the Bi-LSTM method showed promise for time-series forecasting tasks, but its performance may vary depending on the specific data.

The GRU model achieved a MAPE of 9.20339% for open prices and 8.73113% for close prices. Additionally, the model achieved an RMSE of 0.02520 for available prices and 0.03806 for close prices. The GRU model showed good performance in predicting both open and close prices, with a MAPE lower than ARIMA and Bi-LSTM for open prices and lower than ARIMA for close prices. The GRU model is known for capturing long-term dependencies in time-series data, while requiring fewer parameters than LSTMs. The results in the table suggest that the GRU model is effective for predicting stock prices and could be a useful tool for traders and investors. GRU can handle the vanishing gradient problem better than traditional RNNs.

MIMO and ensemble are two different approaches in time-series forecasting. In MIMO, multiple time series are used as inputs to the model to predict the values of all the time series simultaneously. This can be useful in situations where multiple related variables influence each other. In this research, the overall attributes could be used to predict open and closed prices. On the other hand, an ensemble involves using multiple models to make predictions and then combining the results of those models to make a final prediction; this can be useful in uncertain situations about the best model to use or where different models have different strengths and weaknesses. For example, an ensemble could include a combination MLP, LSTM, and CNN models, with the final prediction being a weighted average of the predictions made by each model [31][45][65].

Overall, the result provides insights into the performance of different DL methods for time-series forecasting and highlights the trade-offs between accuracy and complexity of the methods. The MAPE values of all methods at open and close fall into the category of the ability of the forecasting model is very good (<10%). Depending on the specific problem and methods characteristics, one may choose an appropriate method that balances the trade-offs and meets the desired performance criteria.

The results obtained in this study have significant practical implications, particularly in the context of stock trading and investment. Accurate stock price predictions can greatly benefit investors, traders, and financial institutions by providing valuable insights for decision-making. By leveraging MIMO time-series forecasting models, market participants can gain a competitive edge by identifying potential price movements, trends, and patterns in stock markets, this can help optimize trading strategies, improve risk management, and enhance portfolio performance. Moreover, the ability to accurately forecast stock prices can aid in identifying opportunities for arbitrage, hedging, and market timing, leading to increased profitability and reduced financial risks. Additionally, the performance of different methods in the study may have revealed specific patterns or trends. For example, certain deep learning models with specific architectures or hyperparameters might have better captured complex market dynamics, long-term dependencies, or nonlinear relationships. Identifying such patterns can guide practitioners in selecting appropriate models and techniques for stock price prediction tasks.

In the context of real-time or large-scale applications, it is essential to evaluate the efficiency of the models in processing and forecasting stock price data, includes assessing their ability to handle high-dimensional data, adapt to evolving market conditions, and provide timely predictions. Discussing the computational aspects of the models can help stakeholders assess the trade-offs between accuracy and computational efficiency, enabling them to make informed decisions when selecting models for real-world deployment. Future research should consider evaluating the performance of MIMO time-series forecasting models on multiple datasets from different stocks, various market conditions, and diverse periods, this will help establish the robustness and reliability of the proposed methods and provide a more comprehensive understanding of their performance across different contexts. Ultimately, choosing

a suitable method for time-series forecasting tasks depends on the specific problem, the desired performance criteria, and the trade-offs between accuracy and complexity. This study provides valuable insights into the performance of different deep learning methods for time-series forecasting and can help guide the selection of appropriate methods for various applications.

4. Conclusion

In conclusion, this study compared the performance of various deep learning methods, including MLP, CNN, RNN, LSTM, Bi-LSTM, and GRU, in predicting open and close prices for time-series data. Overall, the LSTM method demonstrated the best performance in terms of MAPE, indicating its superior ability to predict stock prices. The Bi-LSTM method achieved the lowest RMSE values, highlighting its effectiveness in detecting prediction outliers. The GRU model also performed well and is known for handling the vanishing gradient problem better than traditional RNNs. While the CNN method performed better than the ARIMA model, it was outperformed by other deep learning methods, suggesting that it may not be optimal for all time-series forecasting tasks. The MIMO and ensemble approaches provide alternative ways to improve forecasting performance by leveraging multiple time series or combining the strengths of different models. The study focused on evaluating deep learning models for MIMO time-series forecasting, but a comprehensive benchmarking and comparative analysis with conventional approaches may have been limited. Future research should evaluate a more extensive range of conventional forecasting methodologies to understand their strengths and drawbacks better. Experimentation is recommended to confirm findings and test these models on different data sets and periods.

Declarations

Author contribution. The contribution or credit of the author must be stated in this section.

Funding statement. The funding agency should be written in full, followed by the grant number in square brackets and year.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

- [1] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning : A systematic literature review: 2005–2019," *Appl. Soft Comput.*, vol. 90, p. 106181, May 2020, doi: [10.1016/j.asoc.2020.106181](https://doi.org/10.1016/j.asoc.2020.106181).
- [2] M. S. Gorus and M. Aydin, "The relationship between energy consumption, economic growth, and CO2 emission in MENA countries: Causality analysis in the frequency domain," *Energy*, vol. 168, pp. 815–822, Feb. 2019, doi: [10.1016/j.energy.2018.11.139](https://doi.org/10.1016/j.energy.2018.11.139).
- [3] H. Lan, C. Zhang, Y.-Y. Hong, Y. He, and S. Wen, "Day-ahead spatiotemporal solar irradiation forecasting using frequency-based hybrid principal component analysis and neural network," *Appl. Energy*, vol. 247, pp. 389–402, Aug. 2019, doi: [10.1016/j.apenergy.2019.04.056](https://doi.org/10.1016/j.apenergy.2019.04.056).
- [4] K. Bandara, P. Shi, C. Bergmeir, H. Hewamalage, Q. Tran, and B. Seaman, "Sales Demand Forecast in E-commerce Using a Long Short-Term Memory Neural Network Methodology," in *Lecture Notes in Computer Science*, 2019, pp. 462–474, doi: [10.1007/978-3-030-36718-3_39](https://doi.org/10.1007/978-3-030-36718-3_39).
- [5] T. Bikku, "Multi-layered deep learning perceptron approach for health risk prediction," *J. Big Data*, vol. 7, no. 1, p. 50, Dec. 2020, doi: [10.1186/s40537-020-00316-7](https://doi.org/10.1186/s40537-020-00316-7).
- [6] P. Hewage, M. Trovati, E. Pereira, and A. Behera, "Deep learning-based effective fine-grained weather forecasting model," *Pattern Anal. Appl.*, vol. 24, no. 1, pp. 343–366, Feb. 2021, doi: [10.1007/s10044-020-00898-1](https://doi.org/10.1007/s10044-020-00898-1).
- [7] X. Yang, X. Liu, and Z. Li, "Multimodel Approach to Robust Identification of Multiple-Input Single-Output Nonlinear Time-Delay Systems," *IEEE Trans. Ind. Informatics*, vol. 16, no. 4, pp. 2413–2422, Apr. 2020, doi: [10.1109/TII.2019.2933030](https://doi.org/10.1109/TII.2019.2933030).

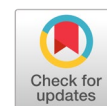
- [8] C. Deng, Y. Huang, N. Hasan, and Y. Bao, "Multi-step-ahead stock price index forecasting using long short-term memory model with multivariate empirical mode decomposition," *Inf. Sci. (Nijl)*, vol. 607, pp. 297–321, Aug. 2022, doi: [10.1016/j.ins.2022.05.088](https://doi.org/10.1016/j.ins.2022.05.088).
- [9] K. Li, G. Huang, S. Wang, B. Baetz, and W. Xu, "A Stepwise Clustered Hydrological Model for Addressing the Temporal Autocorrelation of Daily Streamflows in Irrigated Watersheds," *Water Resour. Res.*, vol. 58, no. 2, pp. 1–31, Feb. 2022, doi: [10.1029/2021WR031065](https://doi.org/10.1029/2021WR031065).
- [10] F. Amato, F. Guignard, S. Robert, and M. Kanevski, "A novel framework for spatio-temporal prediction of environmental data using deep learning," *Sci. Rep.*, vol. 10, no. 1, p. 22243, Dec. 2020, doi: [10.1038/s41598-020-79148-7](https://doi.org/10.1038/s41598-020-79148-7).
- [11] Z. Qu *et al.*, "Temperature forecasting of grain in storage: A multi-output and spatiotemporal approach based on deep learning," *Comput. Electron. Agric.*, vol. 208, p. 107785, May 2023, doi: [10.1016/j.compag.2023.107785](https://doi.org/10.1016/j.compag.2023.107785).
- [12] R. Rakholia, Q. Le, B. Quoc Ho, K. Vu, and R. Simon Carbajo, "Multi-output machine learning model for regional air pollution forecasting in Ho Chi Minh City, Vietnam," *Environ. Int.*, vol. 173, p. 107848, Mar. 2023, doi: [10.1016/j.envint.2023.107848](https://doi.org/10.1016/j.envint.2023.107848).
- [13] A. Thakkar and K. Chaudhari, "Predicting stock trend using an integrated term frequency-inverse document frequency-based feature weight matrix with neural networks," *Appl. Soft Comput.*, vol. 96, p. 106684, Nov. 2020, doi: [10.1016/j.asoc.2020.106684](https://doi.org/10.1016/j.asoc.2020.106684).
- [14] G. Ding and L. Qin, "Study on the prediction of stock price based on the associated network model of LSTM," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 6, pp. 1307–1317, Jun. 2020, doi: [10.1007/s13042-019-01041-1](https://doi.org/10.1007/s13042-019-01041-1).
- [15] J. Silva *et al.*, "An Early Warning Method for Agricultural Products Price Spike Based on Artificial Neural Networks Prediction," in *Lecture Notes in Computer Science*, 2019, pp. 622–632, doi: [10.1007/978-3-030-22741-8_44](https://doi.org/10.1007/978-3-030-22741-8_44).
- [16] S. Mishra, C. Bordin, K. Taharaguchi, and I. Palu, "Comparison of deep learning models for multivariate prediction of time series wind power generation and temperature," *Energy Reports*, vol. 6, pp. 273–286, Feb. 2020, doi: [10.1016/j.egy.2019.11.009](https://doi.org/10.1016/j.egy.2019.11.009).
- [17] C. V. Hudiyanti, J. L. Buliali, and A. Saikhu, "Modelling MIMO Transfer Functions for Analysis of The Relationship Between Temperature and Air Humidity with the Number of Confirmation, Suspect and Probable COVID-19 in Surabaya," in *2021 International Conference on Artificial Intelligence and Computer Science Technology (ICAICST)*, Jun. 2021, pp. 246–251, doi: [10.1109/ICAICST53116.2021.9497836](https://doi.org/10.1109/ICAICST53116.2021.9497836).
- [18] H. Rodriguez, M. Medrano, L. M. Rosales, G. P. Penunuri, and J. J. Flores, "Multi-step forecasting strategies for wind speed time series," in *2020 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, Nov. 2020, pp. 1–6, doi: [10.1109/ROPEC50909.2020.9258743](https://doi.org/10.1109/ROPEC50909.2020.9258743).
- [19] P. Hewage *et al.*, "Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station," *Soft Comput.*, vol. 24, no. 21, pp. 16453–16482, Nov. 2020, doi: [10.1007/s00500-020-04954-0](https://doi.org/10.1007/s00500-020-04954-0).
- [20] A. Alzagher, A. R. Abdellah, and A. Koucheryavy, "Predicting Energy Consumption for UAV-Enabled MEC Using Machine Learning Algorithm," in *Lecture Notes in Computer Science*, 2022, pp. 297–309, doi: [10.1007/978-3-030-97777-1_25](https://doi.org/10.1007/978-3-030-97777-1_25).
- [21] D. Saxena, I. Gupta, A. K. Singh, and C.-N. Lee, "A Fault Tolerant Elastic Resource Management Framework Toward High Availability of Cloud Services," *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 3, pp. 3048–3061, Sep. 2022, doi: [10.1109/TNSM.2022.3170379](https://doi.org/10.1109/TNSM.2022.3170379).
- [22] J.-J. Zhang and H.-S. Yan, "MTN optimal control of MIMO non-affine nonlinear time-varying discrete systems for tracking only by output feedback," *J. Franklin Inst.*, vol. 356, no. 8, pp. 4304–4334, May 2019, doi: [10.1016/j.jfranklin.2019.03.008](https://doi.org/10.1016/j.jfranklin.2019.03.008).
- [23] T. Niu, J. Wang, H. Lu, and P. Du, "Uncertainty modeling for chaotic time series based on optimal multi-input multi-output architecture: Application to offshore wind speed," *Energy Convers. Manag.*, vol. 156, pp. 597–617, Jan. 2018, doi: [10.1016/j.enconman.2017.11.071](https://doi.org/10.1016/j.enconman.2017.11.071).

- [24] P. Lara-Benítez, L. Gallego-Ledesma, M. Carranza-García, and J. M. Luna-Romera, "Evaluation of the Transformer Architecture for Univariate Time Series Forecasting," in *Lecture Notes in Computer Science*, 2021, pp. 106–115, doi: [10.1007/978-3-030-85713-4_11](https://doi.org/10.1007/978-3-030-85713-4_11).
- [25] C. Yin and Q. Dai, "A deep multivariate time series multistep forecasting network," *Appl. Intell.*, vol. 52, no. 8, pp. 8956–8974, Jun. 2022, doi: [10.1007/s10489-021-02899-x](https://doi.org/10.1007/s10489-021-02899-x).
- [26] Z. Wu, G. Luo, Z. Yang, Y. Guo, K. Li, and Y. Xue, "A comprehensive review on deep learning approaches in wind forecasting applications," *CAAI Trans. Intell. Technol.*, vol. 7, no. 2, pp. 129–143, Jun. 2022, doi: [10.1049/cit2.12076](https://doi.org/10.1049/cit2.12076).
- [27] H. Huang, Y. Wang, Y. Li, Y. Zhou, and Z. Zeng, "Debris-Flow Susceptibility Assessment in China: A Comparison between Traditional Statistical and Machine Learning Methods," *Remote Sens.*, vol. 14, no. 18, p. 4475, Sep. 2022, doi: [10.3390/rs14184475](https://doi.org/10.3390/rs14184475).
- [28] A. Truchot *et al.*, "Machine learning does not outperform traditional statistical modelling for kidney allograft failure prediction," *Kidney Int.*, vol. 103, no. 5, pp. 936–948, May 2023, doi: [10.1016/j.kint.2022.12.011](https://doi.org/10.1016/j.kint.2022.12.011).
- [29] M. Sousa, A. M. Tomé, and J. Moreira, "Long-term forecasting of hourly retail customer flow on intermittent time series with multiple seasonality," *Data Sci. Manag.*, vol. 5, no. 3, pp. 137–148, Sep. 2022, doi: [10.1016/j.dsm.2022.07.002](https://doi.org/10.1016/j.dsm.2022.07.002).
- [30] A. L. Schaffer, T. A. Dobbins, and S.-A. Pearson, "Interrupted time series analysis using autoregressive integrated moving average (ARIMA) models: a guide for evaluating large-scale health interventions," *BMC Med. Res. Methodol.*, vol. 21, no. 1, p. 58, Dec. 2021, doi: [10.1186/s12874-021-01235-8](https://doi.org/10.1186/s12874-021-01235-8).
- [31] H. Taud and J. F. Mas, "Multilayer Perceptron (MLP)," in *Lecture Notes in Geoinformation and Cartography*, 2018, pp. 451–455, doi: [10.1007/978-3-319-60801-3_27](https://doi.org/10.1007/978-3-319-60801-3_27).
- [32] B. Cai *et al.*, "Resilience evaluation methodology of engineering systems with dynamic-Bayesian-network-based degradation and maintenance," *Reliab. Eng. Syst. Saf.*, vol. 209, p. 107464, May 2021, doi: [10.1016/j.ress.2021.107464](https://doi.org/10.1016/j.ress.2021.107464).
- [33] R. Reichenberg, "Dynamic Bayesian Networks in Educational Measurement: Reviewing and Advancing the State of the Field," *Appl. Meas. Educ.*, vol. 31, no. 4, pp. 335–350, Oct. 2018, doi: [10.1080/08957347.2018.1495217](https://doi.org/10.1080/08957347.2018.1495217).
- [34] A. Safari and M. Davallou, "Oil price forecasting using a hybrid model," *Energy*, vol. 148, pp. 49–58, Apr. 2018, doi: [10.1016/j.energy.2018.01.007](https://doi.org/10.1016/j.energy.2018.01.007).
- [35] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on Convolutional Neural Networks (CNN) in vegetation remote sensing," *ISPRS J. Photogramm. Remote Sens.*, vol. 173, pp. 24–49, Mar. 2021, doi: [10.1016/j.isprsjprs.2020.12.010](https://doi.org/10.1016/j.isprsjprs.2020.12.010).
- [36] J. Lu, L. Tan, and H. Jiang, "Review on Convolutional Neural Network (CNN) Applied to Plant Leaf Disease Classification," *Agriculture*, vol. 11, no. 8, p. 707, Jul. 2021, doi: [10.3390/agriculture11080707](https://doi.org/10.3390/agriculture11080707).
- [37] R. Chauhan, K. K. Ghanshala, and R. . Joshi, "Convolutional Neural Network (CNN) for Image Detection and Recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Dec. 2018, pp. 278–282, doi: [10.1109/ICSCCC.2018.8703316](https://doi.org/10.1109/ICSCCC.2018.8703316).
- [38] I. E. Livieris, N. Kiriakidou, S. Stavroyiannis, and P. Pintelas, "An Advanced CNN-LSTM Model for Cryptocurrency Forecasting," *Electronics*, vol. 10, no. 3, p. 287, Jan. 2021, doi: [10.3390/electronics10030287](https://doi.org/10.3390/electronics10030287).
- [39] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018, doi: [10.1016/j.ejor.2017.11.054](https://doi.org/10.1016/j.ejor.2017.11.054).
- [40] S. Ghimire, Z. M. Yaseen, A. A. Farooque, R. C. Deo, J. Zhang, and X. Tao, "Streamflow prediction using an integrated methodology based on convolutional neural network and long short-term memory networks," *Sci. Rep.*, vol. 11, no. 1, p. 17497, Sep. 2021, doi: [10.1038/s41598-021-96751-4](https://doi.org/10.1038/s41598-021-96751-4).
- [41] N. Q. K. Le, E. K. Y. Yapp, and H.-Y. Yeh, "ET-GRU: using multi-layer gated recurrent units to identify electron transport proteins," *BMC Bioinformatics*, vol. 20, no. 1, p. 377, Dec. 2019, doi: [10.1186/s12859-019-2972-5](https://doi.org/10.1186/s12859-019-2972-5).

- [42] A.-N. Buturache and S. Stancu, "Solar Energy Production Forecast Using Standard Recurrent Neural Networks, Long Short-Term Memory, and Gated Recurrent Unit," *Eng. Econ.*, vol. 32, no. 4, pp. 313–324, Oct. 2021, doi: [10.5755/j01.ee.32.4.28459](https://doi.org/10.5755/j01.ee.32.4.28459).
- [43] H. V. Bitencourt, O. Orang, L. A. F. de Souza, P. C. L. Silva, and F. G. Guimarães, "An embedding-based non-stationary fuzzy time series method for multiple output high-dimensional multivariate time series forecasting in IoT applications," *Neural Comput. Appl.*, vol. 35, no. 13, pp. 9407–9420, May 2023, doi: [10.1007/s00521-022-08120-5](https://doi.org/10.1007/s00521-022-08120-5).
- [44] U. Kamath, J. Liu, and J. Whitaker, "Convolutional Neural Networks," in *Deep Learning for NLP and Speech Recognition*, Cham: Springer International Publishing, 2019, pp. 263–314, doi: [10.1007/978-3-030-14596-5_6](https://doi.org/10.1007/978-3-030-14596-5_6).
- [45] K. Sekaran, P. Chandana, N. M. Krishna, and S. Kadry, "Deep learning convolutional neural network (CNN) With Gaussian mixture model for predicting pancreatic cancer," *Multimed. Tools Appl.*, vol. 79, no. 15–16, pp. 10233–10247, Apr. 2020, doi: [10.1007/s11042-019-7419-5](https://doi.org/10.1007/s11042-019-7419-5).
- [46] S. Fan, N. Xiao, and S. Dong, "A novel model to predict significant wave height based on long short-term memory network," *Ocean Eng.*, vol. 205, p. 107298, Jun. 2020, doi: [10.1016/j.oceaneng.2020.107298](https://doi.org/10.1016/j.oceaneng.2020.107298).
- [47] G. Kumar, S. Jain, and U. P. Singh, "Stock Market Forecasting Using Computational Intelligence: A Survey," *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1069–1101, May 2021, doi: [10.1007/s11831-020-09413-5](https://doi.org/10.1007/s11831-020-09413-5).
- [48] S. Carta, A. Ferreira, A. S. Podda, D. Reforgiato Recupero, and A. Sanna, "Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting," *Expert Syst. Appl.*, vol. 164, p. 113820, Feb. 2021, doi: [10.1016/j.eswa.2020.113820](https://doi.org/10.1016/j.eswa.2020.113820).
- [49] Y. Kumar, A. Koul, S. Kaur, and Y.-C. Hu, "Machine Learning and Deep Learning Based Time Series Prediction and Forecasting of Ten Nations' COVID-19 Pandemic," *SN Comput. Sci.*, vol. 4, no. 1, p. 91, Dec. 2022, doi: [10.1007/s42979-022-01493-3](https://doi.org/10.1007/s42979-022-01493-3).
- [50] A. P. Wibawa, I. T. Saputra, A. B. P. Utama, W. Lestari, and Z. N. Izdihar, "Long Short-Term Memory to Predict Unique Visitors of an Electronic Journal," in *2020 6th International Conference on Science in Information Technology (ICSITech)*, Oct. 2020, pp. 176–179, doi: [10.1109/ICSITech49800.2020.9392031](https://doi.org/10.1109/ICSITech49800.2020.9392031).
- [51] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujiyanto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed Convolutional Neural Network," *J. Big Data*, vol. 9, no. 1, p. 44, Dec. 2022, doi: [10.1186/s40537-022-00599-y](https://doi.org/10.1186/s40537-022-00599-y).
- [52] A. R. F. Dewandra, A. P. Wibawa, U. Pujiyanto, A. B. P. Utama, and A. Nafalski, "Journal Unique Visitors Forecasting Based on Multivariate Attributes Using CNN," *Int. J. Artif. Intell. Res.*, vol. 6, no. 2, pp. 1–8, Jul. 2022, doi: [10.29099/ijair.v6i1.274](https://doi.org/10.29099/ijair.v6i1.274).
- [53] P. Dhruv and S. Naskar, "Image Classification Using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN): A Review," in *Advances in Intelligent Systems and Computing*, 2020, pp. 367–381, doi: [10.1007/978-981-15-1884-3_34](https://doi.org/10.1007/978-981-15-1884-3_34).
- [54] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Phys. D Nonlinear Phenom.*, vol. 404, p. 132306, Mar. 2020, doi: [10.1016/j.physd.2019.132306](https://doi.org/10.1016/j.physd.2019.132306).
- [55] A. W. Saputra, A. P. Wibawa, U. Pujiyanto, A. B. P. Utama, and A. Nafalski, "LSTM-based Multivariate Time-Series Analysis : A Case of Journal Visitors Forecasting," *Ilk. J. Ilm.*, vol. 14, no. 1, pp. 57–62, 2022, doi: [10.33096/ilkom.v14i1.1106.57-62](https://doi.org/10.33096/ilkom.v14i1.1106.57-62).
- [56] A. P. Wibawa, R. R. Ula, A. B. P. Utama, M. Y. Chuttur, A. Pranolo, and Haviluddin, "Forecasting e-Journal Unique Visitors using Smoothed Long Short-Term Memory," in *2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, Oct. 2021, pp. 609–613, doi: [10.1109/ICEEIE52663.2021.9616628](https://doi.org/10.1109/ICEEIE52663.2021.9616628).
- [57] A. Pranolo, Y. Mao, A. P. Wibawa, A. B. P. Utama, and F. A. Dwiyanto, "Robust LSTM With Tuned-PSO and Bifold-Attention Mechanism for Analyzing Multivariate Time-Series," *IEEE Access*, vol. 10, pp. 78423–78434, 2022, doi: [10.1109/ACCESS.2022.3193643](https://doi.org/10.1109/ACCESS.2022.3193643).

- [58] P. L. Seabe, C. R. B. Moutsinga, and E. Pindza, "Forecasting Cryptocurrency Prices Using LSTM, GRU, and Bi-Directional LSTM: A Deep Learning Approach," *Fractal Fract.*, vol. 7, no. 2, p. 203, Feb. 2023, doi: [10.3390/fractalfract7020203](https://doi.org/10.3390/fractalfract7020203).
- [59] A. Zeroual, F. Harrou, A. Dairi, and Y. Sun, "Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study," *Chaos, Solitons & Fractals*, vol. 140, p. 110121, Nov. 2020, doi: [10.1016/j.chaos.2020.110121](https://doi.org/10.1016/j.chaos.2020.110121).
- [60] G. Yigit and M. F. Amasyali, "Simple But Effective GRU Variants," in *2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, Aug. 2021, pp. 1–6, doi: [10.1109/INISTA52262.2021.9548535](https://doi.org/10.1109/INISTA52262.2021.9548535).
- [61] J. Zhao, H. Qu, J. Zhao, H. Dai, and D. Jiang, "Spatiotemporal graph convolutional recurrent networks for traffic matrix prediction," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 11, pp. 1–14, Nov. 2020, doi: [10.1002/ett.4056](https://doi.org/10.1002/ett.4056).
- [62] A. B. P. Utama, A. P. Wibawa, Muladi, and A. Nafalski, "PSO based Hyperparameter tuning of CNN Multivariate Time-Series Analysis," *J. Online Inform.*, vol. 7, no. 2, pp. 193–202, 2022, doi: [10.15575/join.v7i2.858](https://doi.org/10.15575/join.v7i2.858).
- [63] Y. Mao, A. Pranolo, A. P. Wibawa, A. B. Putra Utama, F. A. Dwiyanto, and S. Saifullah, "Selection of Precise Long Short Term Memory (LSTM) Hyperparameters based on Particle Swarm Optimization," in *2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, May 2022, pp. 1114–1121, doi: [10.1109/ICAAIC53929.2022.9792708](https://doi.org/10.1109/ICAAIC53929.2022.9792708).
- [64] A. P. Wibawa, Z. N. Izdihar, A. B. P. Utama, L. Hernandez, and Haviluddin, "Min-Max Backpropagation Neural Network to Forecast e-Journal Visitors," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIC)*, Apr. 2021, pp. 052–058, doi: [10.1109/ICAIC51459.2021.9415197](https://doi.org/10.1109/ICAIC51459.2021.9415197).
- [65] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting," *IEEE Access*, vol. 8, pp. 180544–180557, 2020, doi: [10.1109/ACCESS.2020.3028281](https://doi.org/10.1109/ACCESS.2020.3028281).

An advanced deep learning model for maneuver prediction in real-time systems using alarming-based hunting optimization



Swati Jaiswal^{a,1}, Chandra Mohan Balasubramanian^{a,2,*}

^a School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India

¹ swatijaiswal26@gmail.com; ² dr.abc@outlook.com

* corresponding author

ARTICLE INFO

Article history

Received February 28, 2023

Revised April 10, 2023

Accepted April 21, 2023

Available online June 13, 2023

Keywords

Deep learning

Autonomous vehicle driving

Traffic sign detection

Lane prediction

Controller optimization

ABSTRACT

The increasing trend of autonomous driving vehicles in smart cities emphasizes the need for safe travel. However, the presence of obstacles, potholes, and complex road environments, such as poor illumination and occlusion, can cause blurred road images that may impact the accuracy of maneuver prediction in visual perception systems. To address these challenges, a novel ensemble model, named ABHO-based deep CNN-BiLSTM has been proposed for traffic sign detection. This model combines a hybrid convolutional neural network (CNN) and bidirectional long short-term memory (BiLSTM) with the alarming-based hunting optimization (ABHO) algorithm to improve maneuver prediction accuracy. Additionally, a modified hough-enabled lane generative adversarial network (ABHO based HoughGAN) has been proposed, which is designed to be robust to blurred images. The ABHO algorithm, inspired by the defending and social characteristics of starling birds and *Canis latrans*, allows the model to efficiently search for the optimal solution from the available solutions in the search space. The proposed ensemble model has shown significantly improved accuracy, sensitivity, and specificity in maneuver prediction compared to previously utilized methods, with minimal error during lane detection. Overall, the proposed ensemble model addresses the challenges faced by autonomous driving vehicles in complex and obstructed road environments, offering a promising solution for enhancing safety and reliability in smart cities.



This is an open access article under the [CC-BY-SA](#) license.



1. Introduction

In recent years, autonomous driving has emerged as one of the most desirable research areas in the artificial intelligence (AI) community. Vehicles can now operate automatically to carry out routine driving activities effectively and safely [1]. The four functional modules that make up autonomous vehicles' core components are environment sensing, decision-making, motion planning, and motion control [2]. The decision-making and motion planning modules, which link environment sensing with motion control, constitute the autonomous vehicle's "brain" and are considered to be of the utmost importance [3], [4]. The lane-changing choice is an important component of the research in this area, and the driving decision-making system is the key technology for maintaining the driving safety of AVs [5]–[7]. Making decisions in unpredictable and dynamic traffic settings is one of the difficulties in achieving full automation of driving [8]. Making decisions involves coming up with a series of motion behaviors to carry out certain tasks, like merging into a crowded lane, navigating an unguarded crossroads, and overtaking with ease on a highway [1], [9]. The robot motion planning algorithm

frequently serves as an inspiration for conventional motion planning techniques, such as artificial potential fields (APF).

A heuristic-based hybrid APF-based motion planning technique was presented in [10]. Combining the proposed methods with optimization algorithms significantly enhanced their performance [11]. An autonomous vehicle's decision-making layer must take into account interactive and synergetic input from other cars to produce human-like driving behaviors. The intentions and responses of nearby cars may be modeled and predicted using probabilistic approaches and partially observable Markov decision processes (POMDP) [1]. Despite significant advancements, there are few reports of autonomous vehicles (AVs) making decisions that take into account other vehicles' social interactions. Capturing the features during vehicle interactions is crucial for improving the decision-making of AVs [4]. The emergence of connected and autonomous vehicles (CAVs) that can sense their environment, make decisions, exercise autonomous control, and exchange data with other vehicles and infrastructure is the result of the quick development of communication technology and autonomous driving technology [12]–[14]. The authors in [15] construct a decision-making system by merging Markov Decision Process (MDP) and RL since RL can offer numerous advantages in tackling complex uncertain sequential decision issues. Deep neural networks (DNN) are used to create a human-like decision-making system that can adjust to actual driving circumstances [4], [16].

Traditional control methods, such as constant spacing (CS) policy, constant time headway (CTH) policy, and sliding mode control (SMC) [17]–[19], have a poor ability to adapt to the environment and are unable to make precise and efficient decisions based on a variety of complex driving situations, particularly in situations where CAVs and conventional driver-controlled vehicles coexist [14]. The search engine, probabilistic sampling, prospective field, approximation curve, and mathematical optimization approaches are the five primary groups among the many algorithms that have been researched for path planning [3]. The most popular and useful path planning algorithm is the one called graph search. In terms of avoiding collisions, the graph search method performs well. The thickness of the grid, however, frequently affects its ideal course. A typical sampling approach that effectively searches the best path while taking non-holonomic restrictions into account is known as rapidly exploring random trees (RRT) [20]. However, RRT needs to continue to strengthen its security and the fineness of its intended course [4]. One drawback of these techniques is that some gesture parameters, which are frequently employed in path planning, are non-linear and non-convex, which might lead to the NP-hard (non-deterministic polynomial-time hard) issue [11], [21].

In this research, traffic sign detection and maneuver prediction-based vehicle control are used to control autonomous driving cars. Social behaviors, such as driving patterns and targets of the vehicles in the immediate surrounding area, are taken into account. The hybridized algorithm is utilized for both traffic sign detection and maneuver prediction, which is motivated by advanced algorithms for AV control and decision-making. The modified Hough-enabled Lane GAN model is used to accurately segment the surrounding driving area in the input image for maneuver prediction, while the ensembled CNN-BiLSTM classifier is then applied to predict the traffic sign and make accurate decisions for autonomous driving. In addition, Alarming-based hunting optimization, the alarming-based hunting optimization (ABHO) algorithm is well implemented in the modified Hough-enabled Lane GAN and the ensembled CNN-BiLSTM classifier for lane detection and traffic sign prediction. The shared weights in the lane detection technique and the tunable parameters in the traffic sign prediction technique are controlled by the ABHO algorithm.

The research utilizes several techniques for autonomous driving, including a modified Hough-enabled Lane GAN model for maneuver prediction, an ensemble CNN-BiLSTM classifier for traffic sign detection, and the ABHO algorithm for controlling shared weights in the lane detection technique and tunable parameters in the traffic sign prediction technique. The paper is organized into sections on existing works, safe driving in an intelligent environment method, results, and a conclusion.

2. Related works and Challenges

The implementation of autonomous vehicles (AVs) is expected to have a considerable positive impact on traffic safety by reducing the number of accidents by up to 94%. However, AV accidents can still occur due to various unforeseen environmental obstacles, such as human-driven vehicles, bicycles, animals, and passengers. Even fully autonomous cars cannot guarantee being completely crash-free under these circumstances. As a result, ethical concerns arise when dealing with such challenges, particularly when human lives are at stake. This section provides an overview of traditional approaches to decision-making-based autonomous vehicles, including path selection and braking, as well as their benefits and challenges.

In this section, the autonomous vehicle-based decision-making process using various strategies is revealed. An effective fuzzy CoCoSo approach was created by [22] built on the logarithmic method and Power Heronian function to address the problem of additional benefit selection in vehicular management techniques. Three primary stages make up the suggested MCDM paradigm. The MCDM's inputs, such as criteria, options, and experts, are chosen in the first step. The logarithmic technique is used to determine the optimal parameters in the second step. The final stage ranks the options according to the Power Heronian function. The suggested fuzzy LM PH'CoCoSo methodology's efficacy is undeniable. However, the technical intricacy of the fuzzy WPHA and fuzzy WGPHA functions for evaluating the computational technique can be a constraint. The decision-making and mobility control for traffic movements of an autonomous vehicle (AV) taking into account the human behaviors of other traffic users were addressed in this [4] unique integrated approach. When making decisions and predicting the condition of a course of an autonomous vehicle, the Stackelberg Game theory and Model Predictive Control (MPC) are both employed. The ability of the agile solution to handle various social contacts with other vehicle drivers demonstrates its viability and efficacy. Only the velocity and acceleration behaviors are taken into account for obstacle vehicles because the lane-change behaviors of these vehicles are not part of the high-speed driving situation. An automated, safe, and effective decision-making paradigm for AVs was put forth by [23] for driving at junctions. To find the best navigation rule in terms of security and protection, the deep Q-network method was used. The suggested approach might aid in developing the decision-making component of AVs to improve travel convenience and traffic flow. This study's shortcomings include the fact that the bigger standard deviations meant that driving comfort was reduced. In an environment of rapid change, [14] presented an autonomous braking decision-making technique that chooses the best course of action using deep reinforcement learning (DRL). To increase safe driving, the automobile can proactively adopt the best braking behavior in an urgent situation once the strategy has been trained correctly. To execute high-level control techniques to coordinate CAVs in typical circumstances, multi-agent reinforcement learning is necessary.

A unique LC decision (LCD) model is presented by [7] that enables autonomous cars to acquire judgments that are similar to those made by humans. This approach integrates the XGBoost algorithm with a deep autoencoder (DAE) network. The presented method is currently only relevant to the traditional LC decision-making mechanism in straight lanes or curved lanes on motorways due to the complication and instability of regular traffic. A predictive control paradigm for moral judgments in driverless vehicles is presented forth by [24] using the principles of rational ethics. The author proposes the use of powerful AI tools and reasonable procedures to develop ethical guidelines for autonomous vehicles. One such approach is the Lexicographic Optimization-based Model Predictive Controller (LO-MPC), which prioritizes barriers and restrictions to ensure the flexible application of ethical principles. To address lane change decision-making [11], the author proposes a risk-aware driving decision strategy using deep reinforcement learning's Risk Awareness Prioritized Replay Deep Q-Network (RA-PRDQN). This approach aims to identify a sequence of actions that minimize risk and prevent accidents with the host car in congested environments with both static and dynamic obstacles. The sample selection probability function can be improved by considering vehicle location sets and incorporating stopping behavior for speed regulation using deep reinforcement learning. Another decision-making [1] system prioritizes the safe and effective operation of autonomous vehicles. The author presents a simulation of passing driving situations and defines standard methods such as the intelligent driver model and minimization of overall braking caused by merging traffic. For highway overtaking, the author

proposes using the Dyna-H algorithm, which combines a modified Q-learning algorithm with a heuristic planning approach. Overall, these approaches aim to develop safe and ethical decision-making systems for autonomous vehicles. To develop online decision-making techniques for autonomous vehicles, deep learning and enhanced RL algorithms must also be combined. The proposed model achieved high accuracy in detecting traffic signs and predicting lanes compared to the existing techniques [25]. The research aims to enhance the decision-making capabilities of autonomous vehicles to ensure safety, energy efficiency, and mobility. The author [23] efficiently ranked the agents relying upon their importance in making decisions, using the CNN network that effectively learned the features and obtained the domain knowledge. The decision-making system acts as the central nerve of driverless vehicles and is important for the safe and effective operation of vehicles [26]. While considering the surrounding environment, the other car motion and the evaluation of self-esteemed vehicles, decision-making is indicated to develop reasonable and safe driving characteristics at the human level [27].

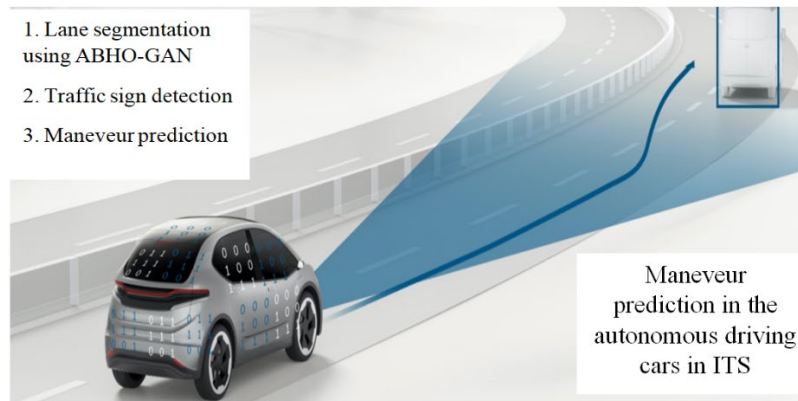
The challenges considered during the development of effective decision-making of autonomous vehicles are as follows:

- In the decision-making process for motion planning, Stackelberg game theoretic optimization and Model Predictive Control (MPC)-based optimization are used to determine the optimal course of action, which is then executed within predefined limits. However, if these limits are too narrow, the motion planner may struggle to find viable alternatives. On the other hand, setting the boundaries too broadly can significantly increase the computational complexity of position control [4].
- Therefore, it is crucial to strike a balance between setting limits that are too narrow or too broad. This will ensure that the motion planner can find feasible solutions within a reasonable timeframe. Achieving the optimal direction of flow within the expected timeframe is a complex task that requires careful consideration and balancing of various factors [4].
- However, using a fuzzy control system on a vehicle has the drawback of requiring a level of understanding to define fuzzy rules and similarity measures. The choice of the membership function is where a fuzzy logic-based control technique becomes challenging. Bandwidth is significantly impacted by the settings for the membership function and fuzzy word set [14].
- The fact that various motion requirements employed in motion planning are frequently non-linear and non-convex poses a drawback of the risk awareness prioritized replay deep Q-network technique [11]. This may result in the NP-hard (non-deterministic polynomial-time hard) problem.
- One major drawback of probabilistic-based techniques is that they solely use specialized information to provide rule-based action, failing to make the right decisions in disruptive environments and ignoring the learning aspects of the human drivers in navigation [11].
- The diminishing gradient experienced during training presents the biggest difficulty in using simple RNNs. The number of instances the gradient signal is ultimately multiplied can be as great as the time steps taken. When dealing with sequence data, a standard Recurrent Neural Network (RNN) may not be suitable for capturing long-term dependencies. This is because, in a deep or extended sequence analysis, the gradient of the network's output may struggle to impact the weights of the preceding layers. As a result, it becomes challenging for the network to record long-term dependencies in the sequence data. The network's weights won't be properly updated under gradient vanishing, leading to very low weight values [28].

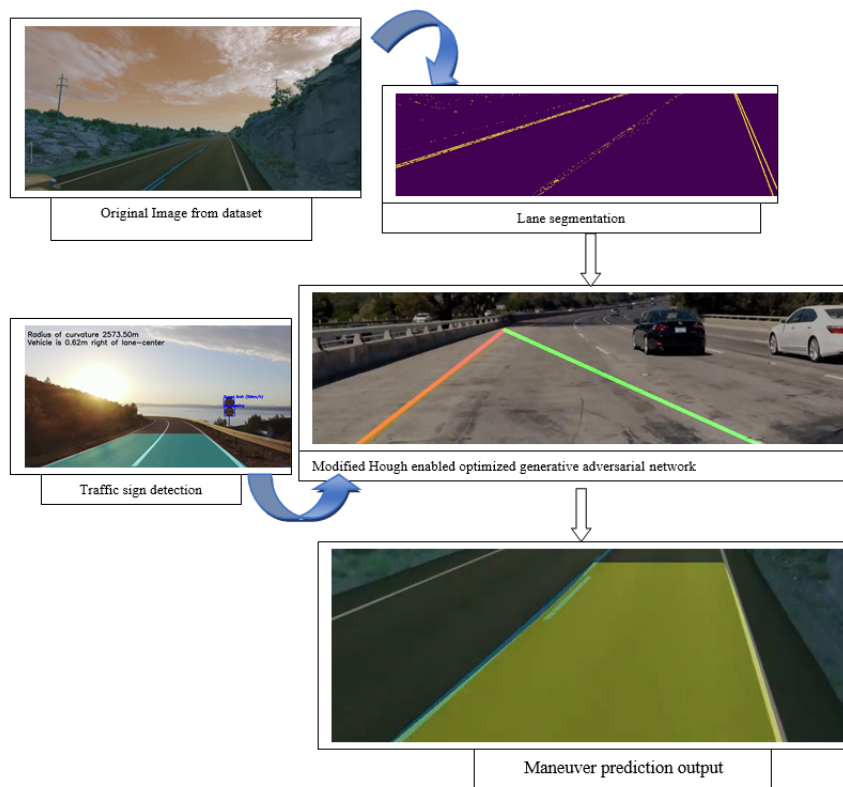
3. Method

In this section decision-making system for the autonomous driving cars using deep models are discussed. The autonomous vehicles require a strong decision controller to support the safe-driving experience in the smart cities for which the road video dataset is acquired. To the video frames, traffic sign detection and maneuver prediction is done using modified CNN-BiLSTM classifier and ABHO-Hough GAN model. The algorithm, ABHO is designed for training the classifier parameters to support

the prediction with accuracy. The ABHO algorithm is developed by integrating the hunting characteristics of *Canis latrans* with the leadership hierarchy and alarming nature of starling birds. On the other hand, the pre-processed video data is fed forward to the ABHO-Hough GAN model, which is tuned by ABHO that has shown good image enhancement and image restoration capabilities. ABHO-Hough GAN model has the tendency to update the performance based on the optimization algorithm and effective maneuver detection. Fig.1 (a) and Fig.1 (b) shows the illustration of the intelligent transportation using maneuver prediction.



(a) Real-time Driving scenario in autonomous driving system



(b) Illustration of decision-making in autonomous vehicle

Fig. 1. The illustration of the intelligent transportation using maneuver prediction

3.1. Road vehicle video database

The road vehicle video database [4] is utilized in this research for traffic sign detection and maneuver prediction as the initial step, which is expressed as

$$D = \sum_{i=0}^d D_d \tag{1}$$

where, the utilized road vehicle database is denoted as D , and the total available videos in the database is denoted as D_d , which is in the range of 0 to d . Each video from the database is supposed to hold Ff number of video frames and for ensuring the accurate support system, the frame-wise processing is enabled.

3.2. Traffic sign detection using Optimized CNN-BiLSTM classifier

The video frames are acquired from the road video, for which the traffic sign detection is done through the designed ABHO-Ensembled model. Fig. 2 shows ABHO-Ensembled model with three layers of CNN, a layer of convolution, leaky ReLU, and MaxPooling, which makes-up the ensemble model's initial channel. The deep CNN holds the filter size as 264 with the kernel size of of dimension. Initially, the frame is processed using the CNN structure of the first channel to extract spatial characteristics, but the depth of time features extracted from the raw high-dimensional data is insufficient. To finish the extraction of the data time-series features and extract the long-term dependencies between the data features, the BiLSTM structure of dimension is employed. While the model is being trained, the BiLSTM structure can prevent gradient disappearance and gradient explosion. After reshaping the output from CNN, the BiLSTM utilizes the dropout size, which is then fed to the dense layer for an efficient detection of the traffic sign in each frame.

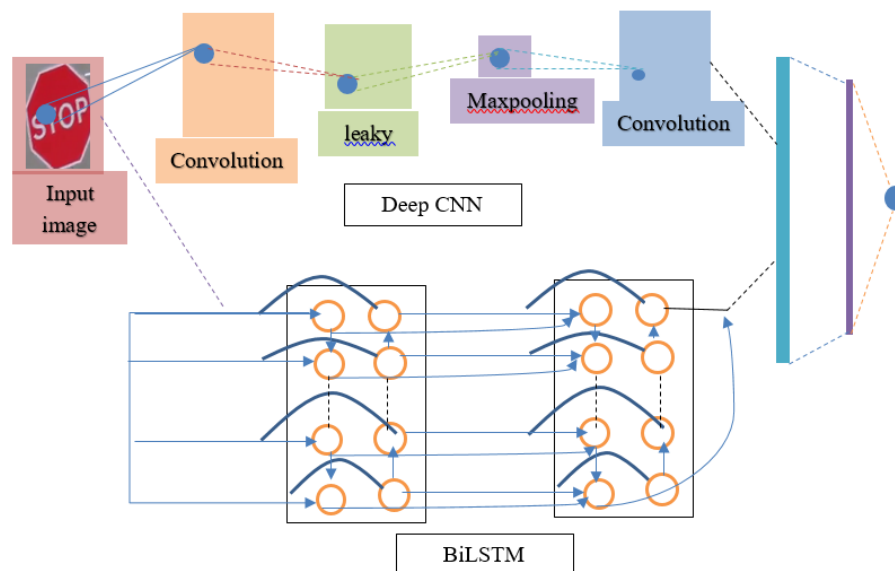


Fig. 2. ABHO-Ensembled model for traffic sign detection

3.2.1. Algorithm for tuning the Ensemble model

The traffic sign detection is the basic need for ensuring the safe driving using the autonomous cars and the accurate detection depends on the ensembled fusion parameters, which is decided optimally using ABHO. ABHO is proposed by employing the exceptional behavior of the *Canis latrans* [29] and the starling bird [30] for the observation of traffic signs in the road video. The ability of the *Canis latrans* relies in the effective balance between the exploitation and exploration stages. The social grading system of the guiding beta and a lack of emphasis on following dominant norms are what distinguish the Coyote's algorithmic behavior. The ABHO approach places more emphasis on social interactions and opinion-sharing during the hunting process. Certain common issues, such as the excessive processing time and inadequate searching potential in the *Canis latrans* performance, are resolved through the characteristics, such as scary as well as the defending characters of the starling bird. The combined behavior enhances the resilience and power of the suggested ABH optimization, leading to a good performance. The starling bird is highly intelligent and has a good memory when compared to many other small birds. The technique is to be used to solve global optimization issues due to its simplicity,

scalability, and high performance. The performance of the starling bird is evaluated to the optimization problems that belong to popular engineering applications.

1) Inspiration

The Coyote algorithm, which is based on the *Canis lupus* genus, inspired by the *Canis latrans* species, and serves as both an ecological and swarm intelligence criterion, is the basis for the population-based approach that is suggested. Even though the *Canis latrans* is used as the pack leader, the social hierarchy and dominant standards of these species are disregarded by its unique mathematical structure configuration. Furthermore, unlike grey wolf hunting, *Canis latrans* hunting emphasizes the social structure and experiences that share as a whole rather than only hunting prey. By considering the social organization of the *Canis latrans* and their environmental adaption, the suggested method offers a unique mathematical model in comparison to metaheuristics. It also provides novel techniques to balance the exploration and exploitation phases of the optimization process. The *Canis latrans*'s behavior has been linked to both internal (such as gender, social standing, and pack membership) and extrinsic (like snowfall height, snowpack severity, climate, and corpse weight) factors. As a result, the alarming-based hunting mechanism was proposed based on the social settings of the *Canis latrans* and starling birds.

2) Mathematical modeling of alarming-based hunting optimization

The three top most significant phases in the ABHO algorithm is the initialization, fitness evaluation of the population, and establishing ranking depending on the measured fitness.

- Initialization: According to the social environment, the *Canis latrans*'s worldwide population in addition to the starling bird population is randomly generated, which is expressed as,

$$P = \{P_1, P_2, P_3, \dots, P_i, \dots, P_x\} \quad (2)$$

where, x be the total population in an attained cluster, the solutions are denoted as P and equation (3) provides the following random values that are generated individually for the a^{th} *Canis latrans* in the u^{th} pack at the dimension of k .

$$G_{a,k}^{u,r} = c_k + e_k(y_k - C_k) \quad (3)$$

where the k^{th} resolution parameter is utilized to represent the upper and lower bounds as c_k and y_k in the social context G , respectively.

- Population ranking: The fitness values are measured for the attained solutions individually for the effective ranking, which assists to proceed with the following hunting process.
- Establish ranking groups: As the consequence of establishing the feasible random solutions, the *Canis latrans*'s deviation resultant to the social conditions is determined by the following equation,

$$fitnes = F(P) \quad (4)$$

The *Canis latrans* are dispersed randomly across the population, therefore they may decide to separate from the group and become alone instead of joining them. The maximum capacity of *Canis latrans* that may be separated from the group over the total population.

- Choose producer position: Unlike followers, producers can look for food in a wider variety of locations, and the producers are expected to have substantial energy stores and give followers access to foraging places or directions. It is in charge of locating locations with abundant food supplies. The evaluation of an individual's fitness values determines their degree of energy reserves. The starling bird starts chirping as alarm messages as soon as they spot the predator. The producers must guide all followers to the safe location if the alert value exceeds the safety level. Each iteration updates the producer's location as follows:

$$P^{t+1} = \begin{cases} P^t * \exp\left(\frac{-i}{\beta, \max^t}\right) & \text{if } r_2 < T \\ P^t + U.W & \text{if } r_2 \geq T \end{cases} \quad (5)$$

Depending on the social characteristics of the *Canis latrans*, the position of the starling bird is updated as,

$$P^{t+1} = \begin{cases} \frac{1}{2} \left[w_1 y_1 + w_2 y_2 + P^t \left(1 + \exp\left(\frac{-i}{\beta, \max^t}\right) \right) \right] & r_2 < T \\ P^t + U.W + rand \times P_{pers}^{best} & \end{cases} \quad (6)$$

where the weights of *Canis latrans* group are denoted as w_1 and w_2 , the personal best population is represented as P_{pers}^{best} . The random variable is represented as r_2 and U with the threshold of T at the iteration of t .

- Choose follower position: The regulations must be upheld by the followers, who should also act as producers by acting like the starling bird with the highest energy. Many hungry followers are more prone to fly to different locations in search of food to increase their energy. Followers look for food by following the producer who can offer the best food. While waiting for food, some followers may be constantly watching the producers and struggling for it to increase their predation rate. Some followers keep a closer eye on the producers, as was already mentioned. They quickly leave their present designation to struggle for food as soon as they learn that the producer has acquired nice food. If they succeed, they can instantly receive the producer's food; if not, the regulations are still followed. The following is a description of the follower's role updating formula.

$$P^{t+1} = \begin{cases} U \cdot \exp(P_{worst} - P^t) + U_{max} \cdot (P^{best} - P^t) \\ P_{pers}^{t+1} + |P^t - P_{pers}^{t+1}| \cdot B^+ \cdot W + V^{t+1} \end{cases} \quad (7)$$

where the matrix is represented as B and W , in which the velocity of a producer in approaching the food and staying away from enemies is denoted as V .

- Choose remaining followers: The starling bird in the center of the group randomly walks to be close to others, whereas the starling bird at the group's edge swiftly goes into the safe region to gain a better position when aware of the danger. It is possible to express the mathematical model as follows:

$$P^{t+1} = \begin{cases} P_{pers} |P^t - P_{pers}| + \beta |P_{glo} - P^t| \\ \frac{1}{2} \left[2P^t + k \left(\frac{P^t - P_{worst}}{(fitness - P_{worst}) + \epsilon} \right) \right] + w_1 y_1 + w_2 y_2 \end{cases} \quad \text{if } fitness > F_{glo} \quad (8)$$

where the worst as well as the global fitness is denoted as F_{worst} and F_{glo} . The *Canis latrans* share other groups' perspectives and methods for moving from one location to another, yet they lack these traits when hunting and adapting to new social conditions. Therefore, the integration of the defending characters of the starling bird prevents the *Canis latrans* from falling into the local optimum, and there needs to be a solution back so that the algorithm can avoid reaching the local optimum. Incorporating an integrating operation to increase the algorithm's ability to avoid the local optimum is the most popular remedy for this problem. In this work, the optimization is improved by integrating the social characteristics of *Canis latrans* with the starling bird. By incorporating the defending behavior while renovating the social state during opinion sharing, the ABHO optimization has greater flexibility, quick resolution, and incredibly consistent findings. Thus, to improve the effectiveness of the ABHO optimization and fine-tune the classifier's hyperparameters for improved vehicle control.

Fig. 3 shows the Proposed ABHO pseudocode, This system adjusts the positions of the starling bird and reduces energy waste in a random movement to reach ideal solutions with the fewest iterations.

```

1.      Total population  $X$ 
2.      Output: Best population
3.      Initialization
4.          Initialize population
5.           $P = \{P_1, P_2, P_3, \dots, P_i, \dots, P_x\}$ 
6.          Population ranking
7.          Based on fitness
8.          Establish ranking groups
9.          While( $t < t_{max}$ )
10.         {
11.         Call $C_1 = F(P)$  producer1( $x$ )
12.         For $\forall i, i = \{1, \dots, x\}$ , # producer 1
13.         {
14.         Update position based on equation (6)
15.         }
16.         End for
17.         Call $C_2 = F(P)$  follower2( $x$ )
18.         For $\forall (x+1 \leq i \leq m)$ , # Follower 2
19.         {
20.         Update position based on equation (7)
21.         }
22.         End for
23.         Call $C_3 = F(P)$  remaining followers ( $x$ )
24.         For $\forall (m+1 \leq i \leq m)$ , # Remaining followers
25.         {
26.         Update position based on equation (8)
27.         }
28.     Rank population
29.     Update the ranked groups
30.     Return  $P_{best}$ 
31.      $t = t + 1$ 
32. End while

```

Fig. 3. Proposed ABHO pseudocode

Once the traffic sign in the traversing road is detected, the lane segmentation is processed using ABHO-Hough GAN model for maneuver prediction. Thus, both the lane segmentation as well as the maneuver prediction is performed in order to control the autonomous vehicle.

3.3. Modified Hough enabled optimized generative adversarial network for lane segmentation and maneuver prediction

Once the traffic sign detection is accomplished, the lane segmentation and maneuver prediction is guided using the ABHO-Hough GAN model, which comprises of a generator as well as the discriminator. In this research, a ABHO-Hough GAN model is developed for the background subtraction of driving scenes, where the lanes are determined by a discriminator using shared weights and evaluated by a generator depending on the input road vehicle data. The ABHO-Hough GAN model is a remarkable tool for identifying shapes and curves in the road vehicle video images. To determine the particular location or gain geometrical details of the vehicle, it is used to detect loops, ellipses, and lines. The Hough transform is a great tool for recognizing lane lines for the self-driving automobiles in the target area, and the actual benefit of this model is that it predicts lanes that are precise and narrow rather than the broad, flexible boundaries that CNN's typically introduce. The hough Lane transform recognizes lanes in multiple continuous frames as opposed to only the current frame, which is dissimilar from the aforementioned deep-learning-based methods that only detect lanes and are considered a time-

based issue. The proposed technique can provide robust performance in lane detection under difficult circumstances with more detailed information. Using the sign detection and lane segmentation outputs, the maneuver detection is proceeded using the optimized GAN. In Fig. 4, Hough Lane enabled optimized GAN model is presented, where the lane segmentation is done using the optimized GAN model, and the hough lane transform supports the maneuver prediction, where the ABHO algorithm guides the segmentation model to acquire the accurate prediction. The detailed sketch on the ABHO algorithm is presented in section 3.2.1.

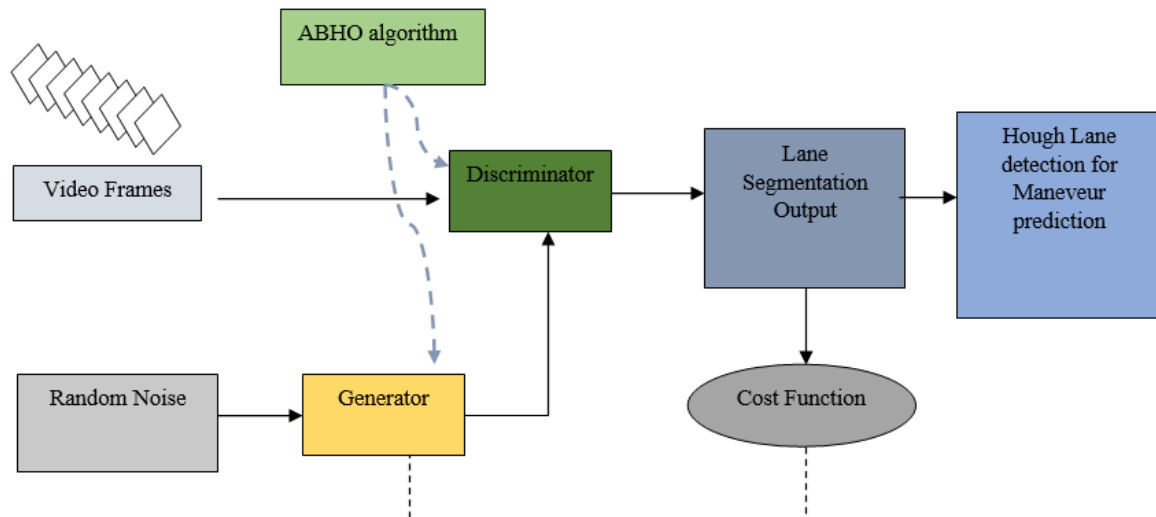


Fig. 4. ABHO-Hough GAN model for Maneuver prediction

4. Results and Discussion

In this section, the reliability of the ABHO-Hough GAN for the maneuver prediction and ABHO-tuned CNN-BiLSTM for traffic sign detection is revealed depending on the performance using the various epoch. The comparative analysis is implemented to show the better efficiency of the proposed model in the research area of an autonomous vehicle. The implementation of both lane prediction and traffic sign detection is done in python on windows 10 OS with 8 GB RAM and the road vehicle video dataset is used for estimation.

The road dataset was enumerated through the aerial images over 1171. Every aerial image is disguised over 2.25 square kilometers with 1500 dimensions from 1500 pixels. The data was divided into three sets in terms of unpremeditated. The following sets are an 1108-image training set, a 14-image validation set, and a 49-image test set. The dataset contains a large amount of urban, suburban as well as rural districts which is present in 2600 square kilometer. To obtain knowledge of real-time decision-making, the test data was helped by enclosing more than 110 square kilometers unaided. The experimental validation of the approach is visualized in Fig. 5.

The performance metrics used for the traffic sign detection along with the ABHO-tuned CNN-BiLSTM is explained as follows

- **Accuracy:** The percentage of samples that the ABHO-based CNN-BiLSTM properly identifies while determining the autonomous vehicle's decision-making system is known as accuracy, and it is given by,

$$acc = \frac{\text{total true prediction}}{\text{Total true and false prediction}} \quad (9)$$

- **Sensitivity:** The true positive outcome of the result from ABHO-based CNN-BiLSTM when the decision-making occurs on the autonomous vehicle, described the sensitivity in terms of probability and it is given as,

$$sen \frac{Total\ true_{pos}}{Total\ true_{pos}\ and\ False_{neg}} \quad (10)$$

- **Specificity:** The true negative of the result from ABHO-based CNN-BiLSTM when the decision-making occurs on the autonomous vehicle, described the specificity in terms of probability and it is given as,

$$sen \frac{Total\ true_{neg}}{Total\ true_{neg}\ and\ False_{pos}} \quad (11)$$

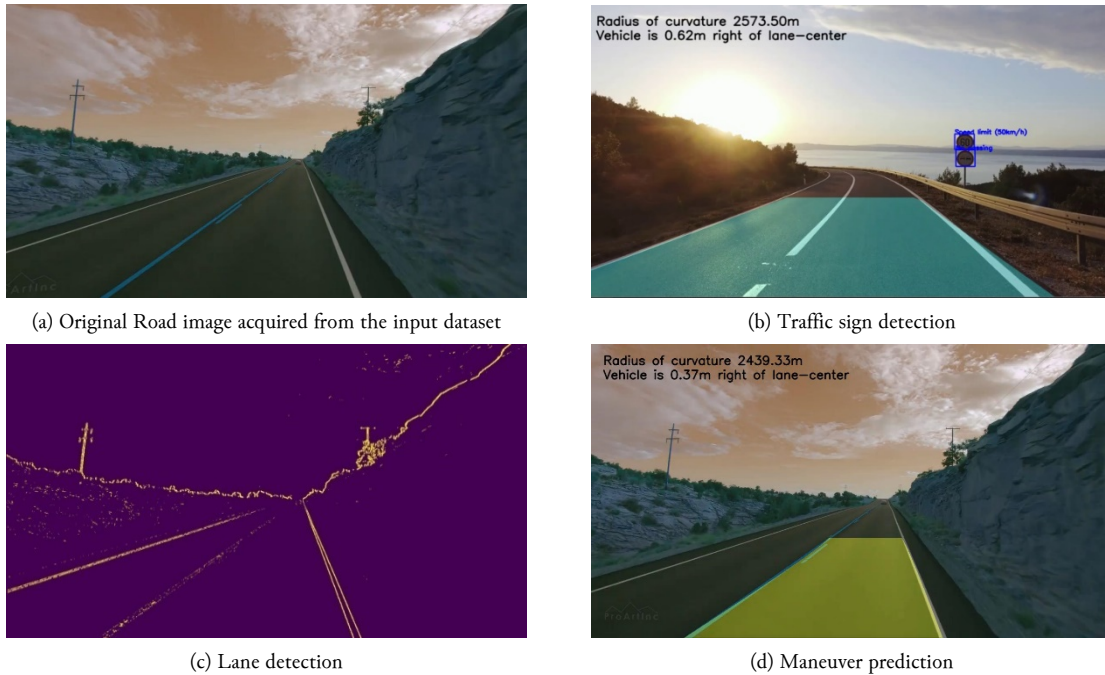


Fig. 5. Autonomous driving system- Experimentation.

The performance metrics used for the lane detection along with the ABHO-Hough GAN is explained as follows.

- **Mean Absolute Error:** The distinction between the magnitudes of the measurement of an individual with the quantity of true value for the ABHO-based Hough GAN, when identifying the lane prediction on an autonomous vehicle is defined as the Mean Absolute Error (MAE) and it is given as,

$$MAE = \frac{1}{v} \sum_{j=1}^v |q_j - q| \quad (12)$$

where, the total error is represented as v , and $|q_j - q|$ denotes an absolute error.

- **Mean Square Error:** The error quantity in the statistical model as well as the difference between the experimental and predicted rate from the ABHO-based Hough GAN in the decision-making function processed for lane prediction on the autonomous vehicle which is estimated in terms of Mean Square error (MSE) and it is given as,

$$MSE = \frac{1}{r} \sum_{j=i}^r (g_j - g_j)^2 \quad (12)$$

where, the available data is denoted as r , g_j describes prediction, and g_j represents the observed value.

- **Root mean squared error:** ABHO-based Hough GAN in an autonomous vehicle is used to determine land prediction in terms of using the mean square value of error in the root which was described as root mean squared error (RMSE) and it is given as,

$$RMSE = \sqrt{\frac{1}{Z} \sum_{j=1}^Z (Q_Z - R_j)^2} \quad (13)$$

where, the observed sample is denoted as R_j , a predicted sample is represented as Q_Z with Z observations.

The performance depending on maneuver prediction and traffic sign detection using ABHO-based Hough GAN as well as the ABHO-based deep CNN-BiLSTM are described in the following section.

4.1. Maneuver prediction and traffic sign detection analysis

The error-based values such as MAE, MSE, and RMSE for ABHO-based Hough GAN for the lane prediction are represented in Fig. 6 (a) represents both the percentage of MAE as well as the training percentage based on the epoch value.

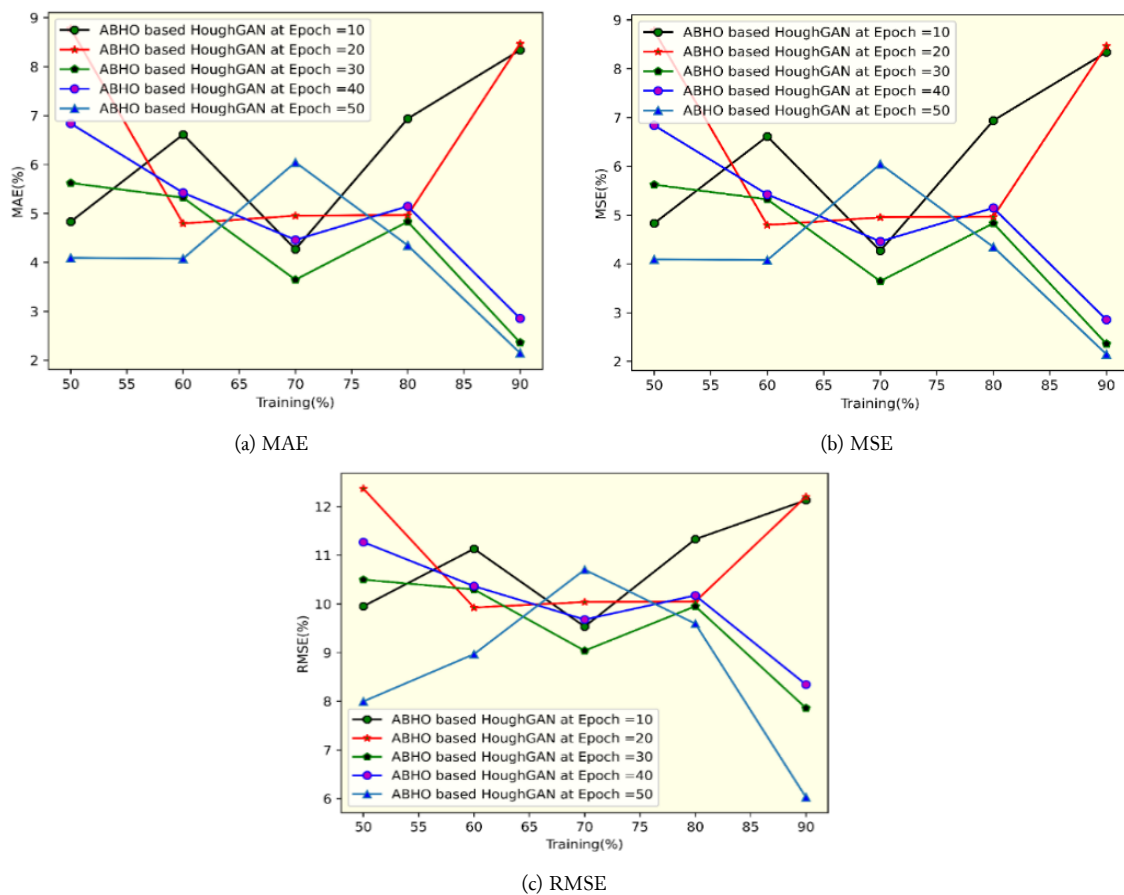


Fig. 6. Maneuver prediction analysis

When the number of training data is 90 for the epoch value 20, and then the attained value of MAE for the ABHO-based Hough GAN is 8.469. Fig. 6 (b) represents both the percentage of MSE as well as the training percentage, based on the epoch value. When the number of training data is 50 for the epoch value 20, then the attained value of MSE for the ABHO-based Hough GAN is 8.776. Fig. 6 (c) represents both the percentage of RMSE as well as the training percentage, based on the epoch value. When the number of training data is 90 for the epoch value 20, then the attained value of RMSE for the ABHO-based Hough GAN is 12.203.

The performance measures-based values such as accuracy, sensitivity, and specificity for ABHO-based deep CNN-BiLSTM for traffic sign detection are represented below in Fig. 7 (a) represents both percentage of accuracy as well as the training percentage based on the epoch value. When the number of training data is 50 for the epoch value is 20, then the attained value of accuracy for the ABHO-based deep CNN-BiLSTM is 84.848.

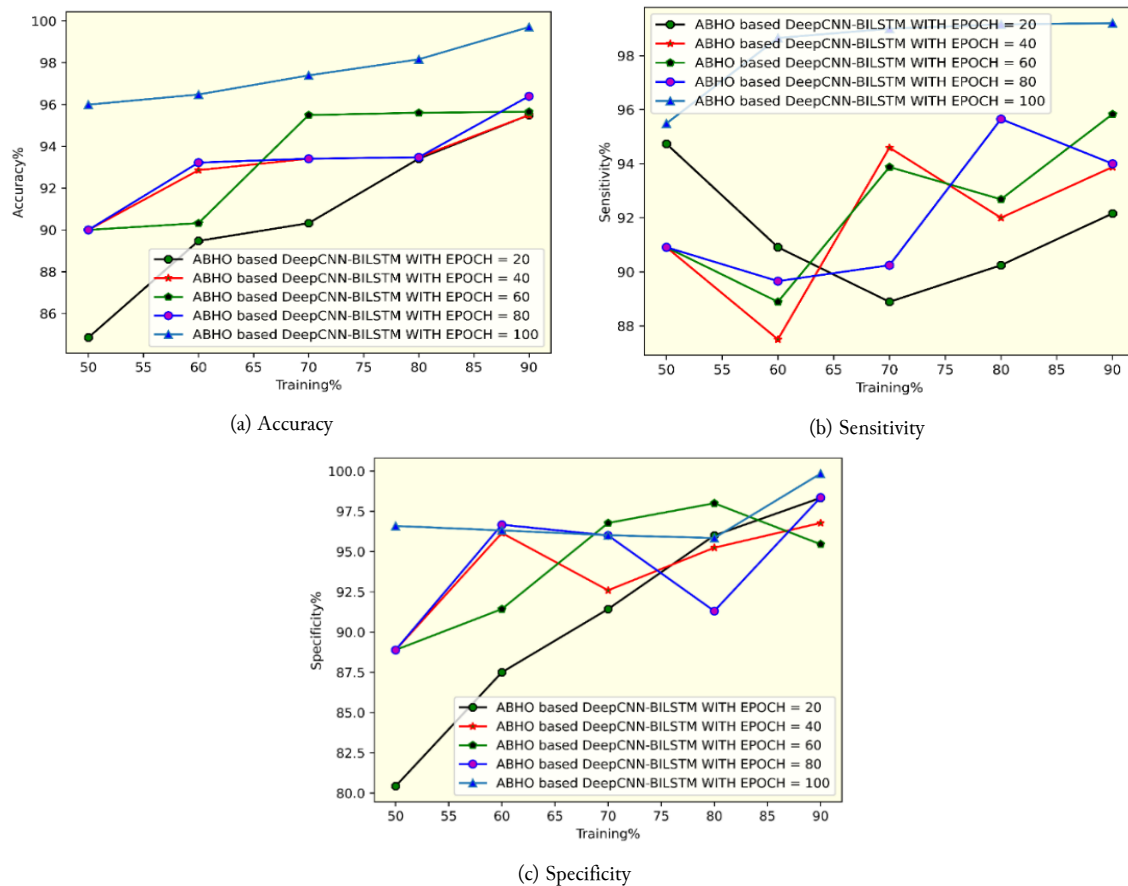


Fig. 7. Traffic sign detection analysis

Fig. 7 (b) represents both percentages of sensitivity as well as the training percentage based on the epoch value. When the number of training data is 60 for the epoch value is 40, then the attained value of sensitivity for the ABHO-based deep CNN-BiLSTM is 87.500. Fig. 7 (c) represents both percentage of specificity as well as the training percentage based on the epoch value. When the number of training data is 50 for the epoch value is 20, then the attained value of specificity for the ABHO-based deep CNN-BiLSTM is 80.429.

The methods considered for comparing the reliability of lane detection are [25], [31]–[33], SegNet-ConvLSTM with SSO, SegNet-ConvLSTM with GWO, SegNet-ConvLSTM with FHO, GAN Model, GAN with COA, GAN with GWO, TCWO based GAN, GAN with SSO. The methods considered for comparing the reliability of traffic sign prediction are [34]–[40], TCWO-based ensemble deep CNN-BiLSTM, and deep CNN-BiLSTM with SSA, respectively.

Accurate and reliable traffic sign recognition is crucial for self-driving vehicles to make informed decisions and avoid accidents. With better prediction accuracy, automated driving systems can respond more quickly and appropriately to traffic signs, such as speed limit signs, stop signs, and yield signs, leading to improved safety for passengers and other road users. Additionally, accurate traffic sign recognition can help optimize vehicle speed and reduce energy consumption, leading to improved efficiency and reduced emissions. Overall, the practical implications of improved traffic sign prediction accuracy are numerous and essential for the successful implementation of automated driving systems in real-world environments.

4.2. Comparison of maneuver prediction models

For evaluating the errors MAE, MSE, and RMSE, the proposed ABHO-based Hough GAN is compared with the other existing methods represented in Fig. 8 (a) represents the MAE for both the proposed as well as the existing depending on the percentage of trained data. When the number of training data is 90 %, the error rate of the proposed method is 2.149. Then the attained improved variation of MAE for the ABHO-based Hough GAN is 64.39 % when compared with the existing GAN with SSO model.

Fig. 8 (b) represents the MSE for both the proposed as well as the existing depending on the percentage of trained data. When the number of training data is 80 %, the error rate of the proposed method is 4.348. Then the attained improved variation of MSE for the ABHO-based Hough GAN is 54.82 % when compared with the existing GAN with SSO model. Fig. 8 (c) represents the RMSE for both the proposed as well as the existing depending on the percentage of trained data. When the number of training data is 90 %, the error rate of the proposed method is 6.027. Then the attained improved variation of RMSE for the ABHO-based Hough GAN is 44.03 % when compared with the existing GAN with SSO model.

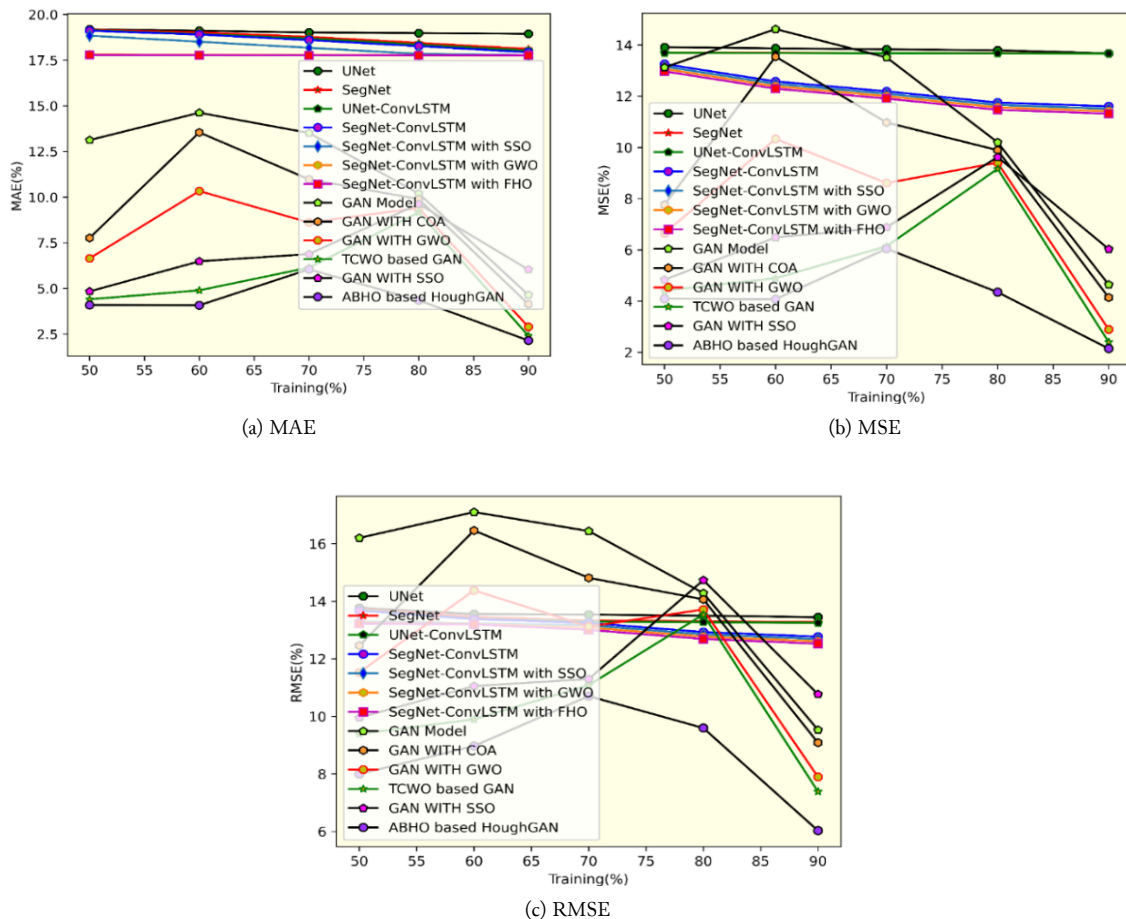


Fig. 8. Comparative analysis for maneuver prediction models

For evaluating the performance measures accuracy, sensitivity, and specificity, the proposed ABHO-based deep CNN-BiLSTM is compared with the other existing methods. The accuracy for both the proposed as well as the existing depending on the percentage of trained data. When the number of training data is 90 %, the accuracy rate of the proposed method is 99.703 %. Then the attained improved variation of accuracy for the ABHO-based deep CNN-BiLSTM is 2.20 % when compared with the exiting deep CNN-BiLSTM with the SSA model.

When the number of training data is 90 %, the sensitivity rate of the proposed method is 99.200 %. Then the attained improved variation of sensitivity for the ABHO-based deep CNN-BiLSTM is 1.47 % when compared with the exiting deep CNN-BiLSTM with the SSA model. When the number of training data is 90 %, the specificity rate of the proposed method is 99.839 %. Then the attained improved variation of specificity for the ABHO-based deep CNN-BiLSTM is 4.11 % when compared with the exiting deep CNN-BiLSTM with the SSA model.

The performance of the ABHO-based maneuver prediction and traffic sign detection approaches is presented in Table 1 and Table 2. The proposed model outperforms other methods in terms of accuracy, sensitivity, and specificity for both traffic sign detection and maneuver prediction, which is crucial for effective decision-making and control of autonomous vehicles. The proposed maneuver prediction model also exhibits lower mean absolute error, mean square error, and root mean square error compared to existing models.

Table 1. Comparison for maneuver prediction

Methods	Manaveur prediction 90 % Training		
	MAE	MSE	RMSE
UNet	18.940	13.670	13.445
SegNet	18.111	13.667	13.278
UNet-ConvLSTM	18.022	13.664	13.247
SegNet-ConvLSTM	17.932	11.605	12.768
SegNet-ConvLSTM with SSO	17.770	11.510	12.687
SegNet-ConvLSTM with GWO	17.767	11.414	12.607
SegNet-ConvLSTM with FHO	17.765	11.319	12.526
GAN Model	4.646	4.646	9.529
GAN with COA	4.142	4.142	9.083
GAN with GWO	2.895	2.895	7.895
TCWO based GAN	2.406	2.406	7.387
GAN with SSO	6.034	6.034	10.768
ABHO based HoughGAN	2.149	2.149	6.027

Table 2. Comparison for traffic sign detection

Methods	Traffic Sign Detection 90 % Training		
	Accuracy (%)	Sensitivity (%)	Specificity (%)
SVM	93.182	91.304	92.857
DeepRNN	93.333	92.308	94.595
DeepCNN	93.590	92.308	95.238
DeepCNN with SSA	94.643	95.122	96.154
DeepCNN with GWO	95.522	96.429	97.059
DeepCNN with FHO	97.778	97.727	97.917
DeepCNN-BiLSTM	94.593	98.970	94.593
DeepCNN-BiLSTM with COA	98.667	98.875	98.667
DeepCNN-BiLSTM with GWO	98.875	99.619	98.875
TCWO based Ensample with DeepCNN-BiLSTM	98.875	98.364	98.875
DeepCNN-BiLSTM with SSA	97.505	97.737	95.740
ABHO based deep CNN-BiLSTM	99.703	99.200	99.839

5. Conclusion

This research proposes an efficient and precise autonomous decision-making technique for AVs to promptly exit hazardous situations. The approach involves developing traffic sign detection and maneuver prediction models using ABHO-based deep CNN-BiLSTM and ABHO-based HoughGAN techniques. The modified Hough-enabled lane GAN is responsible for accurately segmenting the driving area from the input image based on shared weights to facilitate decision-making. The ensemble CNN-BiLSTM classifier is used to anticipate traffic signs and assist the driver in making informed decisions.

The ABHO-based model improves traffic sign prediction accuracy by 2.20%, 1.42%, and 4.11. The use of the modified Hough-enabled lane GAN technique and ensemble CNN-BiLSTM classifier for lane detection and traffic sign prediction respectively, both effectively implemented by the ABHO algorithm, can provide a strong foundation for the development of a comprehensive autonomous decision-making approach. The ABHO algorithm can also contribute to improving the accuracy and efficiency of the models by regulating shared weights and tunable parameters more precisely. The performance improvement achieved in the traffic sign prediction model in this research can potentially be further enhanced in future work. Additionally, future research can explore the interactions of coexisting AVs and employ multi-agent learning algorithms to implement wide control algorithms to manage CAVs in typical circumstances. Other potential avenues for future work may include exploring the robustness and scalability of the proposed approach, as well as addressing ethical and legal issues related to the use of AVs. Moreover, there is the challenge of integrating the autonomous decision-making approach with existing transportation infrastructure and regulatory frameworks. New regulations and policies will need to be developed to ensure the safe and effective deployment of autonomous vehicles on public roads. The integration of autonomous vehicles into existing transportation infrastructure will also require significant upgrades to road infrastructure and communication systems.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

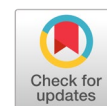
References

- [1] T. Liu *et al.*, “Heuristics-oriented overtaking decision making for autonomous vehicles using reinforcement learning,” *IET Electr. Syst. Transp.*, vol. 10, no. 4, pp. 417–424, Dec. 2020, doi: [10.1049/IET-EST.2020.0044](https://doi.org/10.1049/IET-EST.2020.0044).
- [2] P. Hang, X. Chen, and F. Luo, “LPV/H Controller Design for Path Tracking of Autonomous Ground Vehicles Through Four-Wheel Steering and Direct Yaw-Moment Control,” *Int. J. Automot. Technol.*, vol. 20, no. 4, pp. 679–691, Aug. 2019, doi: [10.1007/S12239-019-0064-1](https://doi.org/10.1007/S12239-019-0064-1).
- [3] P. Hang, C. Lv, Y. Xing, C. Huang, and Z. Hu, “Human-Like Decision Making for Autonomous Driving: A Noncooperative Game Theoretic Approach,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2076–2087, Apr. 2021, doi: [10.1109/TITS.2020.3036984](https://doi.org/10.1109/TITS.2020.3036984).
- [4] P. Hang, C. Lv, C. Huang, J. Cai, Z. Hu, and Y. Xing, “An Integrated Framework of Decision Making and Motion Planning for Autonomous Vehicles Considering Social Behaviors,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14458–14469, Dec. 2020, doi: [10.1109/TVT.2020.3040398](https://doi.org/10.1109/TVT.2020.3040398).
- [5] C. Hatipoglu, Ü. Özgüner, and K. A. Redmill, “Automated lane change controller design,” *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 1, pp. 13–22, Mar. 2003, doi: [10.1109/TITS.2003.811644](https://doi.org/10.1109/TITS.2003.811644).
- [6] Y. Guo, Q. Sun, R. Fu, and C. Wang, “Improved Car-Following Strategy Based on Merging Behavior Prediction of Adjacent Vehicle from Naturalistic Driving Data,” *IEEE Access*, vol. 7, pp. 44258–44268, 2019, doi: [10.1109/ACCESS.2019.2908422](https://doi.org/10.1109/ACCESS.2019.2908422).
- [7] X. Gu, Y. Han, and J. Yu, “A novel lane-changing decision model for autonomous vehicles based on deep autoencoder network and XGBoost,” *IEEE Access*, vol. 8, pp. 9846–9863, 2020, doi: [10.1109/ACCESS.2020.2964294](https://doi.org/10.1109/ACCESS.2020.2964294).
- [8] W. Schwarting, J. Alonso-Mora, and D. Rus, “Planning and Decision-Making for Autonomous Vehicles,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, May 2018, doi: [10.1146/annurev-control-060117-105157](https://doi.org/10.1146/annurev-control-060117-105157).
- [9] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, “Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles,” *IEEE Intell. Veh. Symp. Proc.*, pp.

- 1671–1678, Jul. 2017, doi: [10.1109/IVS.2017.7995949](https://doi.org/10.1109/IVS.2017.7995949).
- [10] K. Tu, S. Yang, H. Zhang, and Z. Wang, “Hybrid A based motion planning for autonomous vehicles in unstructured environment,” *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2019-May, pp. 1–4, 2019, doi: [10.1109/ISCAS.2019.8702779](https://doi.org/10.1109/ISCAS.2019.8702779).
- [11] G. Li, Y. Yang, S. Li, X. Qu, N. Lyu, and S. E. Li, “Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness,” *Transp. Res. Part C Emerg. Technol.*, vol. 134, p. 103452, Jan. 2022, doi: [10.1016/J.TRC.2021.103452](https://doi.org/10.1016/J.TRC.2021.103452).
- [12] J. Guanetti, Y. Kim, and F. Borrelli, “Control of connected and automated vehicles: State of the art and future challenges,” *Annu. Rev. Control*, vol. 45, pp. 18–40, Jan. 2018, doi: [10.1016/J.ARCONTROL.2018.04.011](https://doi.org/10.1016/J.ARCONTROL.2018.04.011).
- [13] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, “A Survey of the State-of-the-Art Localization Techniques and Their Potentials for Autonomous Vehicle Applications,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 829–846, Apr. 2018, doi: [10.1109/JIOT.2018.2812300](https://doi.org/10.1109/JIOT.2018.2812300).
- [14] Y. Fu, C. Li, F. R. Yu, T. H. Luan, and Y. Zhang, “A Decision-Making Strategy for Vehicle Autonomous Braking in Emergency via Deep Reinforcement Learning,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5876–5888, Jun. 2020, doi: [10.1109/TVT.2020.2986005](https://doi.org/10.1109/TVT.2020.2986005).
- [15] R. Zheng, C. Liu, and Q. Guo, “A decision-making method for autonomous vehicles based on simulation and reinforcement learning,” *Proc. - Int. Conf. Mach. Learn. Cybern.*, vol. 1, pp. 362–369, 2013, doi: [10.1109/ICMLC.2013.6890495](https://doi.org/10.1109/ICMLC.2013.6890495).
- [16] L. Li, K. Ota, and M. Dong, “Humanlike Driving: Empirical Decision-Making System for Autonomous Vehicles,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 6814–6823, Aug. 2018, doi: [10.1109/TVT.2018.2822762](https://doi.org/10.1109/TVT.2018.2822762).
- [17] G. Guo and W. Yue, “Autonomous platoon control allowing range-limited sensors,” *IEEE Trans. Veh. Technol.*, vol. 61, no. 7, pp. 2901–2912, 2012, doi: [10.1109/TVT.2012.2203362](https://doi.org/10.1109/TVT.2012.2203362).
- [18] M. Di Bernardo, P. Falcone, A. Salvi, and S. Santini, “Design, Analysis, and Experimental Validation of a Distributed Protocol for Platooning in the Presence of Time-Varying Heterogeneous Delays,” *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 413–427, Mar. 2016, doi: [10.1109/TCST.2015.2437336](https://doi.org/10.1109/TCST.2015.2437336).
- [19] X. Guo, J. Wang, F. Liao, and R. S. H. Teo, “Distributed Adaptive Integrated-Sliding-Mode Controller Synthesis for String Stability of Vehicle Platoons,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 9, pp. 2419–2429, Sep. 2016, doi: [10.1109/TITS.2016.2519941](https://doi.org/10.1109/TITS.2016.2519941).
- [20] M. Kothari and I. Postlethwaite, “A probabilistically robust path planning algorithm for UAVs using rapidly-exploring random trees,” *J. Intell. Robot. Syst. Theory Appl.*, vol. 71, no. 2, pp. 231–253, Aug. 2013, doi: [10.1007/S10846-012-9776-4](https://doi.org/10.1007/S10846-012-9776-4).
- [21] H. Wang, Y. Huang, A. Khajepour, Y. Zhang, Y. Rasekhipour, and D. Cao, “Crash Mitigation in Motion Planning for Autonomous Vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3313–3323, 2019, doi: [10.1109/TITS.2018.2873921](https://doi.org/10.1109/TITS.2018.2873921).
- [22] M. Deveci, D. Pamucar, and I. Gokasar, “Fuzzy Power Heronian function based CoCoSo method for the advantage prioritization of autonomous vehicles in real-time traffic management,” *Sustain. Cities Soc.*, vol. 69, p. 102846, Jun. 2021, doi: [10.1016/J.SCS.2021.102846](https://doi.org/10.1016/J.SCS.2021.102846).
- [23] G. Li *et al.*, “Deep Reinforcement Learning Enabled Decision-Making for Autonomous Driving at Intersections,” *Automot. Innov.*, vol. 3, no. 4, pp. 374–385, Dec. 2020, doi: [10.1007/S42154-020-00113-1](https://doi.org/10.1007/S42154-020-00113-1).
- [24] H. Wang, Y. Huang, A. Khajepour, D. Cao, and C. Lv, “Ethical Decision-Making Platform in Autonomous Vehicles with Lexicographic Optimization Based Model Predictive Controller,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8164–8175, Aug. 2020, doi: [10.1109/TVT.2020.2996954](https://doi.org/10.1109/TVT.2020.2996954).
- [25] S. Jaiswal and B. C. Mohan, “An Efficient Real Time Decision Making System for Autonomous Vehicle Using Timber Chased Wolf Optimization Based Ensemble Classifier,” *J. Eng. Sci. Technol. Rev.*, vol. 16, no. 1, pp. 75–84, 2023, doi: [10.25103/JESTR.161.10](https://doi.org/10.25103/JESTR.161.10).
- [26] Z. Li, C. Lu, Y. Yi, and J. Gong, “A Hierarchical Framework for Interactive Behavior Prediction of Heterogeneous Traffic Participants Based on Graph Neural Network,” *IEEE Trans. Intell. Transp. Syst.*, vol.

- 23, no. 7, pp. 9102–9114, Jul. 2022, doi: [10.1109/TITS.2021.3090851](https://doi.org/10.1109/TITS.2021.3090851).
- [27] K. Muhammad, A. Ullah, J. Lloret, J. Del Ser, and V. H. C. De Albuquerque, “Deep Learning for Safe Autonomous Driving: Current Challenges and Future Directions,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4316–4336, Jul. 2021, doi: [10.1109/TITS.2020.3032227](https://doi.org/10.1109/TITS.2020.3032227).
- [28] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” *J. F. Robot.*, vol. 37, no. 3, pp. 362–386, Apr. 2020, doi: [10.1002/ROB.21918](https://doi.org/10.1002/ROB.21918).
- [29] J. Pierezan and L. Dos Santos Coelho, “Coyote Optimization Algorithm: A New Metaheuristic for Global Optimization Problems,” *2018 IEEE Congr. Evol. Comput. CEC 2018 - Proc.*, pp. 1–8, Sep. 2018, doi: [10.1109/CEC.2018.8477769](https://doi.org/10.1109/CEC.2018.8477769).
- [30] J. Xue and B. Shen, “A novel swarm intelligence optimization approach: sparrow search algorithm,” *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 22–34, Jan. 2020, doi: [10.1080/21642583.2019.1708830](https://doi.org/10.1080/21642583.2019.1708830).
- [31] L. A. Tran and M. H. Le, “Robust u-net-based road lane markings detection for autonomous driving,” *Proc. 2019 Int. Conf. Syst. Sci. Eng. ICSSE 2019*, pp. 62–66, Jul. 2019, doi: [10.1109/ICSSE.2019.8823532](https://doi.org/10.1109/ICSSE.2019.8823532).
- [32] J. Dou, J. Xue, and J. Fang, “SEG-VoxelNet for 3D vehicle detection from RGB and LiDAR data,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2019–May, pp. 4362–4368, May 2019, doi: [10.1109/ICRA.2019.8793492](https://doi.org/10.1109/ICRA.2019.8793492).
- [33] J. P. T. Yusiong and P. C. Naval, “Unsupervised monocular depth estimation of driving scenes using siamese convolutional LSTM networks,” *Int. J. Innov. Comput. Inf. Control*, vol. 16, no. 1, pp. 91–106, Feb. 2020, doi: [10.24507/IJICIC.16.01.91](https://doi.org/10.24507/IJICIC.16.01.91).
- [34] M. Tanveer, T. Rajani, R. Rastogi, Y. H. Shao, and M. A. Ganaie, “Comprehensive review on twin support vector machines,” *Ann. Oper. Res.*, pp. 1–46, Mar. 2022, doi: [10.1007/S10479-022-04575-W](https://doi.org/10.1007/S10479-022-04575-W).
- [35] V. Veerasamy *et al.*, “LSTM Recurrent Neural Network Classifier for High Impedance Fault Detection in Solar PV Integrated Power System,” *IEEE Access*, vol. 9, pp. 32672–32687, 2021, doi: [10.1109/ACCESS.2021.3060800](https://doi.org/10.1109/ACCESS.2021.3060800).
- [36] C. A. Hamm *et al.*, “Deep learning for liver tumor diagnosis part I: development of a convolutional neural network classifier for multi-phasic MRI,” *Eur. Radiol.* 2019 297, vol. 29, no. 7, pp. 3338–3347, Apr. 2019, doi: [10.1007/S00330-019-06205-9](https://doi.org/10.1007/S00330-019-06205-9).
- [37] L. Jianhua and W. Zhiheng, “A hybrid sparrow search algorithm based on constructing similarity,” *IEEE Access*, vol. 9, pp. 117581–117595, 2021, doi: [10.1109/ACCESS.2021.3106269](https://doi.org/10.1109/ACCESS.2021.3106269).
- [38] S. Gupta and K. Deep, “A novel Random Walk Grey Wolf Optimizer,” *Swarm Evol. Comput.*, vol. 44, pp. 101–112, Feb. 2019, doi: [10.1016/J.SWEVO.2018.01.001](https://doi.org/10.1016/J.SWEVO.2018.01.001).
- [39] M. Azizi, S. Talatahari, and A. H. Gandomi, “Fire Hawk Optimizer: a novel metaheuristic algorithm,” *Artif. Intell. Rev.*, vol. 56, no. 1, pp. 287–363, Jan. 2023, doi: [10.1007/S10462-022-10173-W](https://doi.org/10.1007/S10462-022-10173-W).
- [40] Y. Cheng, K. Hu, J. Wu, H. Zhu, and X. Shao, “A convolutional neural network based degradation indicator construction and health prognosis using bidirectional long short-term memory network for rolling bearings,” *Adv. Eng. Informatics*, vol. 48, p. 101247, Apr. 2021, doi: [10.1016/J.AEI.2021.101247](https://doi.org/10.1016/J.AEI.2021.101247).

Pneumonia detection on x-ray imaging using softmax output in multilevel meta ensemble algorithm of deep convolutional neural network transfer learning models



Simeon Yuda Prasetyo ^{a,1,*}, Ghinaa Zain Nabiilah ^{a,2}, Zahra Nabila Izdihar ^{a,3}, Sani Muhamad Isa ^{b,4}

^a Computer Science Department, School of Computer Science, Bina Nusantara University, Jakarta 11480, Indonesia

^b Computer Science Department, BINUS Graduate Program - Master of Computer Science, Jakarta and 11530, Indonesia

¹ simeon.prasetyo@binus.ac.id; ² ghinaa.nabiilah@binus.ac.id; ³ zahra.izdihar@binus.ac.id; ⁴ sani.m.isa@binus.ac.id

* corresponding author

ARTICLE INFO

Article history

Received August 02, 2022

Revised April 2, 2023

Accepted April 16, 2023

Available online July 8, 2023

Keywords

Pneumonia classification

Transfer learning

Deep convolutional neural network

Softmax

Multilevel meta ensemble

ABSTRACT

Pneumonia is the leading cause of death from a single infection worldwide in children. A proven clinical method for diagnosing pneumonia is through a chest X-ray. However, the resulting X-ray images often need clarification, resulting in subjective judgments. In addition, the process of diagnosis requires a longer time. One technique can be applied by applying advanced deep learning, namely, Transfer Learning with Deep Convolutional Neural Network (Deep CNN) and modified Multilevel Meta Ensemble Learning using Softmax. The purpose of this research was to improve the accuracy of the pneumonia classification model. This study proposes a classification model with a meta-ensemble approach using five classification algorithms: Xception, Resnet 15V2, InceptionV3, VGG16, and VGG19. The ensemble stage used two different concepts, where the first level ensemble combined the output of the Xception, ResNet15V2, and InceptionV3 algorithms. Then the output from the first ensemble level is reused for the following learning process, combined with the output from other algorithms, namely VGG16 and VGG19. This process is called ensemble level two. The classification algorithm used at this stage is the same as the previous stage, using KNN as a classification model. Based on experiments, the model proposed in this study has better accuracy than the others, with a test accuracy value of 98.272%. The benefit of this research could help doctors as a recommendation tool to make more accurate and timely diagnoses, thus speeding up the treatment process and reducing the risk of complications.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Pneumonia is an inflammation of the lungs due to an acute respiratory infection caused by viruses, bacteria, or fungi. In a healthy state of breathing, the lungs will consist of tiny sacs (alveoli) filled with air. In contrast, the alveoli will be filled with pus and fluid for people with pneumonia, making breathing painful and limiting oxygen intake. Pneumonia is the leading cause of death due to children's single most extensive infection worldwide. In 2019, around 740,180 children under five died [1]. While data in Indonesia in 2019 revealed that about 314,455 children under five died due to pneumonia based on data from The Ministry of Health of the Republic of Indonesia [2].

Pneumonia can be treated using medications such as antibiotics as the primary step to preventing death due to complications of pneumonia. Early diagnosis can be made so people with pneumonia can get early treatment. A well-known and effective clinical method for diagnosing pneumonia is chest X-

ray images [3]. However, diagnosing pneumonia from chest X-ray images is still incredibly challenging, even for radiologists. X-ray images are often not clear, so it can confuse the results of a diagnosis with other diseases [4]. This results in inconsistencies leading to many subjective decisions, and there are differences between radiologists in diagnosing pneumonia. In addition, radiologists' analysis and diagnosis process often requires a long time because it is still done manually.

A computer-based system is necessary to aid radiologists in detecting pneumonia through analyzing chest X-ray images. The development of machine learning models has good ability in the classification process, including image classification. Chandra and Verma [5] have proven the ability of machine learning models to detect chest X-ray images by comparing several machine learning classification models such as Regression Logistics, Multilayer Perceptron, Random Forest, and Sequential Minimal Optimization. This study also used feature extraction before classification. The optimal result of this research is using logistic regression with 95.53% accuracy. Although it is accurate, machine learning models have shortcomings in managing large amounts of data. The optimal results obtained from the study [6] used a limited amount of data with 412 image data from chest X-rays.

The development research related to image classification is more developed using deep learning methods. Deep learning methods are considered to have an excellent ability to handle large amounts of data. One of the best methods for image classification is to use the Convolutional Neural Network (CNN) along with various variations of CNN [7]. CNN is also successfully used in classifying images for medical problems, such as detecting breast cancer, lung cancer, brain tumor segmentation, and skin diseases [8], [9].

The idea of Transfer Learning, which involves acquiring new knowledge by leveraging existing knowledge, was introduced by Pan *et al.* in 2010. Essentially, this approach involves identifying commonalities between known and unknown knowledge. Since some knowledge areas may be complex to learn from scratch, the use of existing knowledge to facilitate learning is crucial [10].

The fundamental concept of Transfer Learning is to find the correlation between familiar and unfamiliar knowledge to obtain new insights. Transfer Learning refers to the process of transferring knowledge from a known domain (called Source Domain) to an unknown domain (called Target Domain). The primary aim of Transfer Learning is to explore ways to transfer knowledge from the Source Domain to the Target Domain. In the field of machine learning, Transfer Learning aims to use existing models to apply previously acquired knowledge to new knowledge. There are four categories of learning approaches used in Transfer Learning based on their learning methods, which include features, model-based, relationship, and sample-based methods [11].

Research on image classification for pneumonia using deep learning has previously been carried out. As research conducted by Enes and Murat [12] uses two modified models, namely Xception and a Vgg 16-based model that uses transfer learning, fine-tuning, and data augmentation in Categorizing medical images of chest X-rays for individuals with pneumonia. This study showed that the Xception model outperformed the Vgg 16 model in detecting pneumonia. In 2021, Zhang *et al.* [13] modified the architecture of the VGG model by using layers and with fewer parameters. The results of this study compared with other model architectures such as VGG-16, RES-50, Xception, DenseNet21, and MobileNet. However, the performance of the proposed model is superior to those models with 96.068% of accuracy and 0.99107 AUC. Based on this research, in this study, an experiment was carried out by applying ensemble stacking modifications from several model architectures such as Xception, Resnet152V2, InceptionV3, VGG16, and VGG19. This modification was later named Multilevel Ensemble Stacking. With the modification experiment in this study, it is hoped that the results of the diagnostic image from X-rays of pneumonia patients can be more accurate.

Several techniques have been presented in recent years to illustrate the brief process of detecting pneumonia based on chest X-ray images using deep learning or deep CNN methods. The deep CNN method is more widely used and is considered to be successfully applied to improve computer

performance, especially in image classification in the medical field. An example of implementation is the research conducted by Radjpukar [14] which uses a CNN model with 121 layers to detect pneumonia using a chest X-ray image. The dataset used in this study has more than 100,000 images with 14 other diseases associated with chest X-rays, including pneumonia. This study had different results, with radiologists having higher F1 scores detecting all 14 diseases, including pneumonia. Rahman *et al.* conducted a study [15] that utilized a transfer learning approach based on Deep CNN to detect pneumonia. Four deep learning algorithms based on CNN, namely AlexNet, ResNet18, DenseNet201, and SqueezeNet, were trained to classify chest X-ray images of pneumonia and non-pneumonia patients. The results showed that DenseNet201 outperformed the other three deep CNN networks, achieving an optimal accuracy of 98%.

The success of deep learning methods in analyzing images makes Deep CNN models get a lot of attention for classifying images. However, research conducted by Varshni *et al.* [16] tried to use machine learning methods such as SVM, Naive Bayes, KNN, and Random Forest, which is also well known in the classification process as a method for classifying images. Deep CNN models such as Xception, VGG-16, VGG-19, ResNet-50, DenseNet-121, and DenseNet-169 were also used in this study as feature extraction. The optimal result of this research is to use DenseNet-169 and SVM with AUC 0.8002.

Due to the success of the deep CNN model in classifying images, the latest research related to image classification tries to add the application of ensemble learning to get better accuracy results by combining some of the best models. Research conducted by Chouhan *et al.* [17] implements ensemble learning that combines Deep CNN models such as AlexNet, DenseNet121, InceptionV3, ResNet18, and GoogleNet. The results of this study reached an accuracy of 96.4%. Research that applies ensemble learning has also been carried out by Mubrouk *et al.* [18] which combines Deep CNN models such as Vision Transformer, MobileNetV2, and DenseNet169. The optimal result of this research is an accuracy value of 93.91%.

Deep CNN is an alternative learning method because it can identify patterns automatically, even in low-resolution images. Junior *et al.* [19] proposed a deep CNN model with different training strategies. This study evaluates the preprocessing of different images to improve classification by image cropping and histogram equalization. In this research, the ensemble technique uses the VGG16 architecture by modifying the fully connected layer with the Multilayer Perceptron.

The goal of this research is to create and assess a system that can identify pneumonia in chest X-ray images by implementing a multilevel meta-ensemble method. This approach involves combining the predictions of multiple models at different levels of abstraction, using the Softmax activation function as the output. The motivation for using the multilevel meta-ensemble method is to improve the model's performance in detecting pneumonia in chest X-ray images. In the detection of pneumonia in chest X-ray images, there are often variations in image quality, such as lighting, contrast, and radiograph quality. Therefore, by using the multilevel meta-ensemble method, it is expected that the model can learn and extract output features (in the form of Softmax or probabilities output) from various high-level feature maps features, thereby improving the accuracy of pneumonia detection.

The contribution of using the multilevel meta ensemble method is to improve accuracy in detecting pneumonia in chest X-ray images. By utilizing this technique, the model can leverage information derived from various output probabilities (Softmax output), thereby enhancing its ability to distinguish between pneumonia (inflammation) and normal conditions in X-ray images. Furthermore, the use of multilevel meta ensemble method can also contribute to the development of more advanced medical technology. By improving the accuracy of detecting pneumonia in X-ray images, this technology can assist doctors in making more accurate and timely diagnoses, thereby speeding up the treatment process and reducing the risk of complications.

Overall, the use of multilevel meta ensemble method in detecting pneumonia in X-ray images has a strong motivation and contribution to improving the accuracy of detecting this disease, as well as providing a positive impact on the development of medical technology as a whole.

2. Method

2.1. Dataset Collection

This study used an open-source dataset created by Mooney and sourced from Kaggle [20]. The dataset contains approximately 5,865 Chest X-Ray Images, which are categorized into two groups: Pneumonia and Normal. The chest X-ray images used in this study were chosen from a group of previous cases of pediatric patients between the ages of one and five who had undergone chest X-ray imaging during their regular clinical checkups at the Guangzhou Women and Children's Medical Center in Guangzhou. These images were obtained using the anterior-posterior view, which is the usual technique for capturing this type of image [21].

The normal chest X-ray, which is shown in the left panel, depicts lungs that are clear without any visible areas of abnormal opacification. Bacterial pneumonia, as depicted in the middle panel, typically shows a focal lobar consolidation, which means that there is a localized area of the lung appearing dense on the X-ray image. In this case, the consolidation is in the right upper lobe and can be identified by the white arrows. On the other hand, viral pneumonia, as shown in the right panel, has a more diffuse "interstitial" pattern that appears throughout both lungs. This pattern is characterized by a fine, reticular shadowing that is visible on the X-ray image. Understanding these visual characteristics of pneumonia on chest X-rays is important for accurate diagnosis and treatment of patients.

In order to ensure that the data used in the study is reliable, the chest x-ray images were subjected to quality control screening, and any images of low quality or that were unreadable were excluded. The diagnoses for the remaining images were then assessed by two expert physicians to guarantee the accuracy of the training data used in the AI system. To minimize the potential impact of grading errors, a third expert also evaluated the dataset. The Pneumonia dataset has data distribution which is divided into training data, test data and validation with 4695, 521 and 624, respectively. The pneumonia dataset consists of various images of chest X-rays of pneumonia patients and non-pneumonia patients. Fig. 1 contain an example of a dataset image.

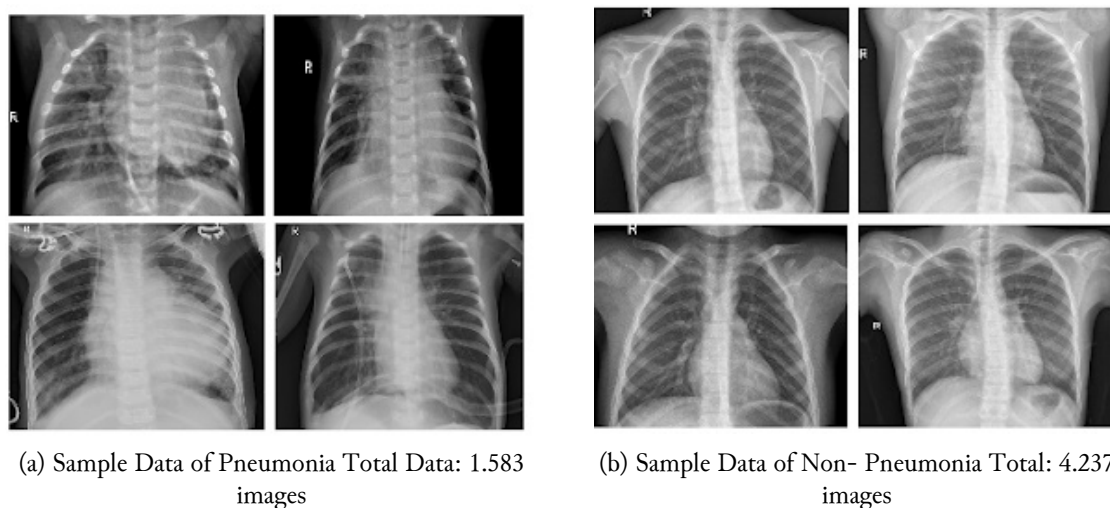


Fig. 1. Example of a dataset image

2.2. Data Preprocessing

In order to standardize the dataset and account for discrepancies in the size of the X-ray images, a decision was made to resize all images to a uniform size of 224 x 224 pixels. Since the dataset is not consistent and images may vary in size, this process is crucial to ensure accurate and reliable analysis. To accomplish this, RGB reordering has been implemented, resulting in a final input of 224 x 224 x 3 for the proposed model. This standardization of the images enables the use of established models and facilitates the analysis process.

The use of a uniform size for all images eliminates inconsistencies that may arise from differences in image sizes, ensuring the accuracy of the analysis. Given that the data set was not uniform and comprised of images of different sizes, it was necessary to transform all images to 224×224 pixels. RGB reordering was employed to make the images consistent and provide a final input of $224 \times 224 \times 3$ to the proposed model. This standardization of the images ensures that the model can effectively process the data set, improving the accuracy and reliability of the analysis.

2.3. Classification using Deep Convolutional Neural Network

The convolutional neural network (CNN) is a type of neural network that is proficient in handling large amounts of data with high precision and minimal computational expense due to its multiple layers. The fundamental architecture of CNN includes various layers, such as convolutional, flattening, pooling and dense layers [22].

After conducting initial preprocessing, the following step is to extract features that meet the criteria. The suggested approach in this study involves utilizing a multilevel ensemble stacking method in a transfer learning model of a Convolutional Neural Network (CNN). CNN offers numerous benefits, including the ability to handle both feature extraction and classification tasks with a single structure. Additionally, this network can extract more complex 2-D features and is completely adaptive, ensuring invariance to geometric and local changes in the image [23].

The Convolutional Neural Network (CNN) consists of three main types of layers: Convolution layer, Pooling layer (Subsampling), and Output layer. These layers are arranged in a feed-forward structure within the network. The convolution layer is always followed by a pooling layer, while the last convolution layer is followed by an output layer. The convolution and pooling layers are 2-D layers, whereas the output layer is a 1-D layer. Each 2-D layer of the CNN comprises multiple planes, where each plane is made up of a 2-D array of neurons. The output of a plane is known as a feature map. The proposed methodology's architectures will be explained in detail in the following section [24].

In this research, several pretrained models such as InceptionV3, Xception, VGG, and ResNet will be used, which will then be combined using a multilevel meta ensemble approach that utilizes output in the form of probabilities (using Softmax activation function). Inception is a development model of the first CNN [25]. Unlike the Inception-V3 which is a combination of all improvements from Inception V2 which uses Label Smoothing as a component to prevent overfitting, factorized 7×7 convolutions, and additional classifier equalization to bring label information to the lower network accompanied by normalization. batch on the layer on the side head [26].

Xception (extreme inception) is a convolutional neural network with 71 deep layers. The Xception model architecture is a convolution that can be separated in depth by improving the Inception model architecture [27]. In the Xception model, a revolutionary deep convolutional neural network architecture is proposed. This causes the Xception model to outperform the V3 inception in the case of image classification with larger data, consisting of 350 million images and 17,000 classes. More effective use of model parameters rather than additional capacity makes for increased performance on the Xception model. The parameters used in the Xception model are also the same as those in the InceptionV3 model [28].

VGG is an architectural model consisting of 16 trained layers that have weights. The VGG model architecture contains a CNN model architecture that utilizes a convolutional layer specification of a small 3×3 convolutional filter. Due to the small size of the convolutional filter, the neural network can have more convolutional layers [29]. This model was created in 2015 and trained on one million images from the ImageNet database [30].

Residual Networks or ResNets were created to solve the gradient problem. The ResNet architecture introduces the concept of residual block or skip connections. The activation layer connects to the next

layer by passing through several layers to form a residual block. To make a network, stack leftover blocks on top of each other; for example, ResNet-50 uses this block in fifty layers [31].

The selected configuration for the pretrained model involves utilizing the weight architecture solely for feature extraction in the convolutional section. This means that during the training process, the weights are adjusted to help the model learn to identify important features in the input data. Once the high-level feature maps are obtained from the convolution, they are flattened, or reshaped into a single vector, and then directly fed into a fully connected layer with a size of 2. In this layer, each of the flattened features is connected to every neuron in the layer. Finally, Softmax activation function is applied to the outputs of the fully connected layer, which allows the model to output a probability distribution over the possible classes that the input data could belong to.

2.4. Ensemble Method

Ensemble learning is how an algorithm learns data using a combination of several algorithms or models (where several learning algorithms are used simultaneously, then combined) to get output with higher accuracy when compared to using only one algorithm. Several types of Ensemble learning include Bagging, Boosting, and Stacking [32].

In this study, a classification system with a meta ensemble approach is proposed to detect chest X-rays of patients with pneumonia and non-pneumonia patients. The classifier combines the predictions of several models to conclude the final result [33]. Specifically, in this study, there are five different classification models on the data that have been processed previously and then assigned a weighting factor to the predictions of each model. The meta-ensemble learning model aims to fit complex data better, lowering uncertainty estimates. Fig. 2 contains an illustration of the usual meta ensemble process.

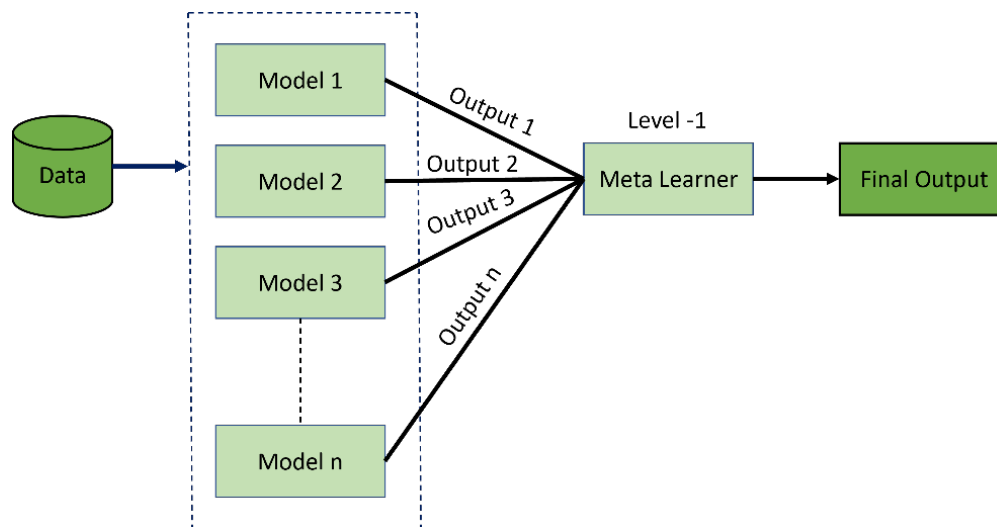


Fig. 2. Meta Ensemble Model

K-Nearest Neighbors (KNN) is one of the algorithms used to classify data based on training data. The data grouping process is based on the number of nearest neighbors (k nearest neighbors). The nearest neighbor search technique that is usually done is to use the Euclidean distance calculation. Euclidean distance is used to find the distance between two points. Equation (1) is a formula for calculating the Euclidean distance [34]. In this study, KNN will be utilized as the meta learner:

$$d = \sqrt{\sum_{i=1}^m (x_i - y_i)^2} \quad (1)$$

where, d is a distance, while x and y are features. In this study, we used one nearest k number. Fig. 3 illustrates the process of grouping data based on a predetermined number of K .

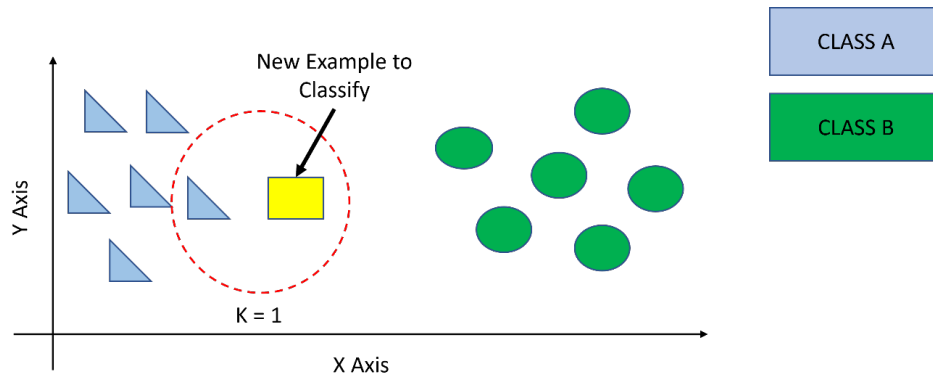


Fig. 3. Illustration of KNN (K-Nearest Neighbors)

2.5. Model Deployment

This study proposes a classification model with a meta ensemble approach using five classification algorithms: Xception, Resnet 15V2, InceptionV3, VGG16, and VGG19 as shown in Fig. 4.

The five algorithms are used after the pre-processing process to classify chest X-ray images. Where each algorithm will produce output in the form of probabilities (Softmax). This study uses the classification results in the form of Softmax in each classification model so that the results are varied. So that the knowledge of the model in classifying data is more diverse. Then the learning process is carried out with the ensemble technique that applies the meta ensemble approach with the KNN classification algorithm. The ensemble technique used in this study also combines the classification results in Softmax. Finally, the KNN algorithm will perform the classification process by selecting the output value that has the closest probability.

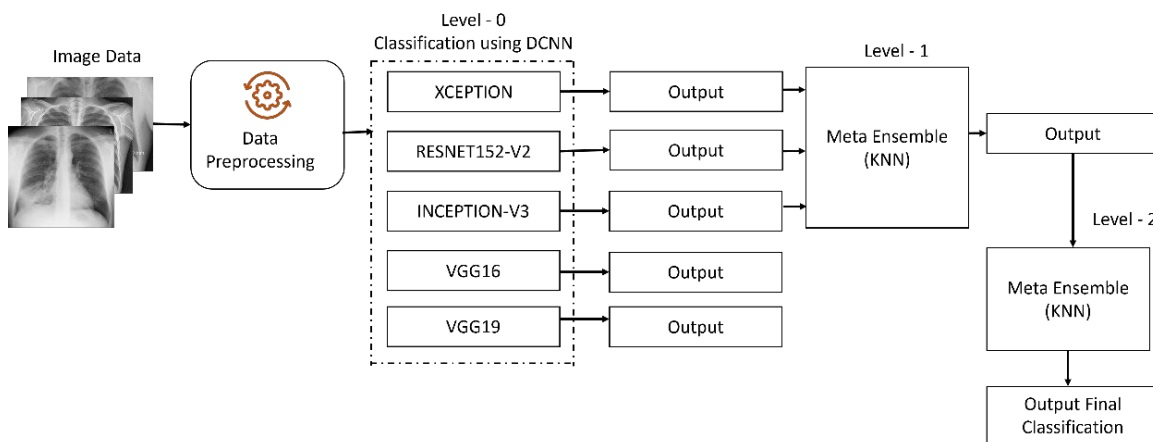


Fig. 4. Schematic Diagram of Model Development

The ensemble stage in this study was carried out using two different concepts, where the first level ensemble was carried out by combining the output of the Xception, ResNet15V2, and InceptionV3 algorithms. Then the output from the first ensemble level is reused for the following learning process, combined with the output from other algorithms, namely VGG16 and VGG19. This process is called ensemble level two. The classification algorithm used at this stage is the same as the previous stage, using KNN as a classification model.

The model learning process is expected to run better using the ensemble model proposed in this study. Such as making the model classification results in the form of probability (Softmax) and using the meta ensemble technique in two levels with the technique of combining the results of the model classification proposed in this study. This is because the classification results produced are in the form of probability (Softmax), and the learning process is carried out more than once using the same

classification algorithm, namely KNN in the ensemble process, but using more diverse classification results so that the final classification model is richer in knowledge from the process previous learning.

2.6. Evaluation Matrix

The evaluation matrix used in this study includes accuracy, precision, recall, and F1 scores. The model's classification was evaluated using four metrics: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). A chest X-ray of a pneumonia patient correctly identified as pneumonia by the model is considered a True Positive. If the model correctly identifies a normal chest X-ray (not pneumonia), it is a True Negative. False Positive occurs when the model incorrectly identifies a normal chest X-ray as pneumonia, while False Negative occurs when a chest X-ray of a pneumonia patient is incorrectly identified as normal by the model. Calculations of accuracy, precision, recall, and F1-score values are found in (2), (3), (4), and (5).

$$\text{accuracy} = \frac{TP+TN}{TN+FP+TP+FN} \quad (2)$$

$$\text{precision} = \frac{TP}{FP+TP} \quad (3)$$

$$\text{recall} = \frac{TP}{FN+FP} \quad (4)$$

$$F1 = 2 \times \frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \quad (5)$$

3. Results and Discussion

This section describes the experimental stages and analyzes the results of the experiments carried out in this study. The experiment in this study was carried out using the Google Colab Pro platform with GPU resources that allocated 25GB of memory for machine learning projects. Google Colab is an interactive computing document that allows users to write, run, and save programs. It is comparable to a Jupyter notebook and operates on a cloud-based platform accessed through a browser. This study also uses the python programming language with a Keras library to run deep learning algorithm commands and TensorFlow as a library that is used to develop and implement Machine Learning and other algorithms that have many mathematical operations. Based on the experiments that have been carried out, [Table 1](#) contains the results of model accuracy using several models proposed in this study.

Table 1. The Pre-Trained Results for Deep CNN Architectural Model

Pre-Trained Model Architecture	Accuracy	Precision	Recall	F1-Score
Xception	0.97696	0.98	0.99	0.98
ResNet152V2	0.97312	0.99	0.97	0.98
InceptionV3	0.96737	0.98	0.98	0.98
VGG19	0.95393	0.96	0.99	0.98
VGG16	0.96353	0.99	0.95	0.97

The first experiment was initiated by classifying using pre-trained Deep CNN models such as Xception, ResNet153V2, InceptionV3, VGG19, and VGG16. The architecture of the model used in this research is modified first in the layered architecture, which is directly connected from the extractor to the fully connected layer. The optimal result of this research is to use Xception. Because the model proposed in this study applies the ensemble technique, it is necessary to use several algorithms to carry out the ensemble process. Based on the experimental results in [Table 1](#), the classification algorithm chosen for the level 1 ensemble process is the algorithm that has the best classification results. It is hoped that the ensemble model will learn more from models with high accuracy. A learning process will affect the classification results when combined again at level 2 with other models with low accuracy. The

next experiment was carried out by comparing the usual meta ensemble method (non-Softmax output), ordinary meta ensemble with Softmax output, multilevel meta ensemble with standard output (non-Softmax output), and multilevel meta ensemble with Softmax output. Table 2 contains the classification results from the experiments carried out.

Table 2. Accuracy Comparison of Original Ensemble Stacking and Multilevel Stacking Ensemble

Architecture	Accuracy	Precision	Recall	F1-Score
Original Ensemble Stacking (using Softmax)	0.9750	0.9800	0.9900	0.9800
Multilevel Ensemble Stacking (using Softmax)	0.9827	0.9800	0.9900	0.9900
Original Ensemble Stacking (non Softmax)	0.9808	0.9800	0.9900	0.9900
Multilevel Ensemble Stacking (non Softmax)	0.9750	0.9800	0.9900	0.9800

This study uses a pre-trained model or a model trained on massive data with weights from ImageNet training so that the pre-trained model has better accuracy. This pre-training model is also included in the Keras library. The model proposed in this study has better accuracy from the trials than the other models, with a test accuracy value of 98.272%. Several things influence this, including the proposed multilevel meta-ensemble technique. This makes the model learn more due to merging the classification model and the relearning technique in stage two. So that makes the model richer in knowledge and makes it easier for the model to find patterns in the classification process. In addition, the classification results using probability (Softmax) also help the model in determining classification because the number of classification classes is more diverse when using probability, so the model studies more variations of classification classes.

Even so, the proposed model has classification results similar to the original stacking technique, whether using Softmax or not, so the ensemble technique using several Deep CNN models can provide good classification results. So that the model performance will be more optimal if it applies the Multilevel Ensemble Stacking technique using Softmax as proposed in this study. In order to test the performance of the proposed classification model, this study made comparisons with other studies that used the similar datasets. Table 3 contains the classification results from the experiments conducted and comparisons with the classification results in other studies.

Table 3. Accuracy comparison of the proposed model in this study with other studies

References	Architecture	Accuracy	Precision	Recall	F1-Score
Muhaza, <i>et al.</i> [35]	CNN Model with Oversampling	-	0.9500	0.9600	-
Shagun, <i>et al.</i> [36]	Proposed NN with VGG16	0.9215	0.9428	0.9308	0.9540
Enes, <i>et al.</i> [37]	Proposed Ensemble CNN	0.9583	-	-	-
Alhassan, <i>et al.</i> [38]	Ensemble Model (DenseNet 169, MobileNetV2, and Vision Transformer)	0.9391	0.9396	0.9299	0.9343
Proposed Method	Multi-Level Meta Stacking using Softmax	0.9808	0.9800	0.9900	0.9900

Based on testing other studies using similar data, this study's proposed multilevel ensemble model has better precision compared to research conducted by Muhaza *et al.* [35] and accuracy, which far beyond the research conducted by Shagun *et al.* [36]. The author also compares with a similar concept, namely ensemble learning, such as research by Enes *et al.* [37] that proposes the ResNet-50, MobileNet, and Xception ensemble models with CNN. However, the model proposed in this study obtains better accuracy. In addition, ensembles using DenseNet 169, MobileNetV2, and Vision Transformer have also been carried out by Alhassan *et al.* [38]. The model proposed in this study obtained better accuracy and

F1 values from the results of these experiments. The confusion matrix is used to evaluate the efficacy of the classification model, as shown in Table 4. In this investigation, the confusion matrix with the highest True Positive Rate (TPR) was obtained, as 383 out of 386 cases were correctly classified for a percentage of 99.222%. The True Negative Rate (TNR) was determined to be 129 out of 135 cases, or 95.555% accuracy, while the False Positive Rate (FPR) was 6 out of 135 cases, or 4.440% error rate. In contrast, the False Negative Rate (FNR) yielded a value of 3 out of 386 cases, or a percentage of 0.777%.

Table 4. Best result of confusion matrix

		Predicted		
		Non-Pneumonia	Pneumonia	Total
Actual	Non-Pneumonia	129 (TN)	6 (FP)	135
	Pneumonia	3 (FN)	383 (TP)	386
	Total	132	389	521

4. Conclusion

The use of the proposed multilevel meta-ensemble architecture and the use of output with probability (Softmax) in this study can affect the model's performance and produce better accuracy. This can help the model explore and learn more words with a more significant number of variations because the ensemble process is carried out in two stages with Softmax. However, the proposed model has the disadvantage that the model training process requires more time. This is due to the use of the deep CNN model architecture and large amounts of data. To further analyze the proposed model architecture in this study, future research can conduct experiments using other classification algorithms, such as machine learning algorithms, or even classification algorithms that do not yet have an excellent ability to recognize image patterns in data. This is intended so that a more in-depth analysis can be carried out regarding the proposed model in this study and whether it can be applied to various classification algorithms to obtain better results.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

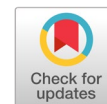
References

- [1] "Pneumonia in children," *World Health Organization*, 2022. Accessed Dec. 01, 2022. [Online]. Available at: <https://www.who.int/news-room/fact-sheets/detail/pneumonia>.
- [2] Kementerian Kesehatan Republik Indonesia, "Pneumonia Pada Anak bisa Dicegah dan Diobati", 2020. Accessed May 01, 2022. [Online]. Available at: <https://www.kemkes.go.id/>.
- [3] W. Khan, N. Zaki, and L. Ali, "Intelligent Pneumonia Identification From Chest X-Rays: A Systematic Literature Review," *IEEE Access*, vol. 9, pp. 51747–51771, Jul. 2021, doi: [10.1109/ACCESS.2021.3069937](https://doi.org/10.1109/ACCESS.2021.3069937).
- [4] O. Stephen, M. Sain, U. J. Maduh, and D.-U. Jeong, "An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare," *J. Healthc. Eng.*, vol. 2019, pp. 1–7, Mar. 2019, doi: [10.1155/2019/4180949](https://doi.org/10.1155/2019/4180949).
- [5] T. B. Chandra and K. Verma, "Pneumonia Detection on Chest X-Ray Using Machine Learning Paradigm," in *Advances in Intelligent Systems and Computing*, vol. 1022 AISC, Springer Science and Business Media Deutschland GmbH, 2020, pp. 21–33, doi: [10.1007/978-981-32-9088-4_3](https://doi.org/10.1007/978-981-32-9088-4_3).
- [6] N. Sharma, V. Jain, and A. Mishra, "An Analysis Of Convolutional Neural Networks For Image Classification," *Procedia Comput. Sci.*, vol. 132, pp. 377–384, Jan. 2018, doi: [10.1016/j.procs.2018.05.198](https://doi.org/10.1016/j.procs.2018.05.198).

- [7] P. Chagas *et al.*, "Evaluation of Convolutional Neural Network Architectures for Chart Image Classification," in *2018 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2018, vol. 2018-July, pp. 1–8, doi: [10.1109/IJCNN.2018.8489315](https://doi.org/10.1109/IJCNN.2018.8489315).
- [8] Z. Wang *et al.*, "Breast Cancer Detection Using Extreme Learning Machine Based on Feature Fusion With CNN Deep Features," *IEEE Access*, vol. 7, pp. 105146–105158, 2019, doi: [10.1109/ACCESS.2019.2892795](https://doi.org/10.1109/ACCESS.2019.2892795).
- [9] W. Alakwaa, M. Nassef, and A. Badr, "Lung Cancer Detection and Classification with 3D Convolutional Neural Network (3D-CNN)," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 8, pp. 66–73, 2017, doi: [10.14569/IJACSA.2017.080853](https://doi.org/10.14569/IJACSA.2017.080853).
- [10] J. C. Hung, K.-C. Lin, and N.-X. Lai, "Recognizing learning emotion based on convolutional neural networks and transfer learning," *Appl. Soft Comput.*, vol. 84, p. 105724, Nov. 2019, doi: [10.1016/j.asoc.2019.105724](https://doi.org/10.1016/j.asoc.2019.105724).
- [11] F. Zhuang *et al.*, "A Comprehensive Survey on Transfer Learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: [10.1109/JPROC.2020.3004555](https://doi.org/10.1109/JPROC.2020.3004555).
- [12] E. Ayan and H. M. Unver, "Diagnosis of Pneumonia from Chest X-Ray Images Using Deep Learning," in *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)*, Apr. 2019, pp. 1–5, doi: [10.1109/EBBT.2019.8741582](https://doi.org/10.1109/EBBT.2019.8741582).
- [13] D. Zhang, F. Ren, Y. Li, L. Na, and Y. Ma, "Pneumonia Detection from Chest X-ray Images Based on Convolutional Neural Network," *Electronics*, vol. 10, no. 13, p. 1512, Jun. 2021, doi: [10.3390/electronics10131512](https://doi.org/10.3390/electronics10131512).
- [14] P. Rajpurkar *et al.*, "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning," *arXiv Comput. Vis. Pattern Recognit.*, pp. 1–7, Nov. 2017, doi: [10.48550/arXiv.1711.05225](https://doi.org/10.48550/arXiv.1711.05225).
- [15] T. Rahman *et al.*, "Transfer Learning with Deep Convolutional Neural Network (CNN) for Pneumonia Detection Using Chest X-ray," *Appl. Sci.*, vol. 10, no. 9, p. 3233, May 2020, doi: [10.3390/app10093233](https://doi.org/10.3390/app10093233).
- [16] D. Varshni, K. Thakral, L. Agarwal, R. Nijhawan, and A. Mittal, "Pneumonia Detection Using CNN based Feature Extraction," in *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Feb. 2019, pp. 1–7, doi: [10.1109/ICECCT.2019.8869364](https://doi.org/10.1109/ICECCT.2019.8869364).
- [17] V. Chouhan *et al.*, "A Novel Transfer Learning Based Approach for Pneumonia Detection in Chest X-ray Images," *Appl. Sci.*, vol. 10, no. 2, p. 559, Jan. 2020, doi: [10.3390/app10020559](https://doi.org/10.3390/app10020559).
- [18] A. Mabrouk, R. P. Diaz Redondo, A. Dahou, M. Abd Elaziz, and M. Kayed, "Pneumonia Detection on Chest X-ray Images Using Ensemble of Deep Convolutional Neural Networks," *Appl. Sci.*, vol. 12, no. 13, p. 6448, Jun. 2022, doi: [10.3390/app12136448](https://doi.org/10.3390/app12136448).
- [19] J. R. Ferreira, D. Armando Cardona Cardenas, R. A. Moreno, M. de Fatima de Sa Rebelo, J. E. Krieger, and M. Antonio Gutierrez, "Multi-View Ensemble Convolutional Neural Network to Improve Classification of Pneumonia in Low Contrast Chest X-Ray Images," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, vol. 2020-July, pp. 1238–1241, doi: [10.1109/EMBC44109.2020.9176517](https://doi.org/10.1109/EMBC44109.2020.9176517).
- [20] "Chest X-Ray Images (Pneumonia)," *Kaggle*. Accessed May 02, 2018. [Online]. Available at: <https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia>.
- [21] D. S. Kermany *et al.*, "Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning," *Cell*, vol. 172, no. 5, pp. 1122–1131.e9, Feb. 2018, doi: [10.1016/j.cell.2018.02.010](https://doi.org/10.1016/j.cell.2018.02.010).
- [22] H. Panwar, P. K. Gupta, M. K. Siddiqui, R. Morales-Menendez, and V. Singh, "Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet," *Chaos, Solitons & Fractals*, vol. 138, p. 109944, Sep. 2020, doi: [10.1016/j.chaos.2020.109944](https://doi.org/10.1016/j.chaos.2020.109944).
- [23] H. Polat and H. Danaei Mehr, "Classification of Pulmonary CT Images by Using Hybrid 3D-Deep Convolutional Neural Network Architecture," *Appl. Sci.*, vol. 9, no. 5, p. 940, Mar. 2019, doi: [10.3390/app9050940](https://doi.org/10.3390/app9050940).
- [24] A. Mortazi and U. Bagci, "Automatically Designing CNN Architectures for Medical Image Segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture*

- Notes in Bioinformatics*), vol. 11046 LNCS, Springer Verlag, 2018, pp. 98–106, doi: [10.1007/978-3-030-00919-9_12](https://doi.org/10.1007/978-3-030-00919-9_12).
- [25] C. Szegedy *et al.*, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, vol. 07-12-June, pp. 1–9, doi: [10.1109/CVPR.2015.7298594](https://doi.org/10.1109/CVPR.2015.7298594).
- [26] M. I. Sarker, H. Kim, D. Tarasov, and D. Akhmetzanov, “Inception Architecture and Residual Connections in Classification of Breast Cancer Histology Images,” *arXiv Image Video Process.*, vol. 1–8, Dec. 2019, doi: [10.48550/arXiv.1912.04619](https://doi.org/10.48550/arXiv.1912.04619).
- [27] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, vol. 2017-Janua, pp. 1800–1807, doi: [10.1109/CVPR.2017.195](https://doi.org/10.1109/CVPR.2017.195).
- [28] S. H. Kassani, P. H. Kassani, R. Khazaeinezhad, M. J. Wesolowski, K. A. Schneider, and R. Deters, “Diabetic Retinopathy Classification Using a Modified Xception Architecture,” in *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Dec. 2019, pp. 1–6, doi: [10.1109/ISSPIT47144.2019.9001846](https://doi.org/10.1109/ISSPIT47144.2019.9001846).
- [29] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, “Going Deeper in Spiking Neural Networks: VGG and Residual Architectures,” *Front. Neurosci.*, vol. 13, p. 95, Mar. 2019, doi: [10.3389/fnins.2019.00095](https://doi.org/10.3389/fnins.2019.00095).
- [30] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2014, pp. 1–14, doi: [10.48550/arXiv.1409.1556](https://doi.org/10.48550/arXiv.1409.1556).
- [31] D. Sarwinda, R. H. Paradisa, A. Bustamam, and P. Anggia, “Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer,” *Procedia Comput. Sci.*, vol. 179, pp. 423–431, Jan. 2021, doi: [10.1016/j.procs.2021.01.025](https://doi.org/10.1016/j.procs.2021.01.025).
- [32] Z.-H. Zhou, “Ensemble Learning,” in *Machine Learning*, Singapore: Springer Singapore, 2021, pp. 181–210, doi: [10.1007/978-981-15-1967-3_8](https://doi.org/10.1007/978-981-15-1967-3_8).
- [33] A. Almasri, E. Celebi, and R. S. Alkhalwaleh, “EMT: Ensemble Meta-Based Tree Model for Predicting Student Performance,” *Sci. Program.*, vol. 2019, pp. 1–13, Feb. 2019, doi: [10.1155/2019/3610248](https://doi.org/10.1155/2019/3610248).
- [34] W. Xing and Y. Bei, “Medical Health Big Data Classification Based on KNN Classification Algorithm,” *IEEE Access*, vol. 8, pp. 28808–28819, 2020, doi: [10.1109/ACCESS.2019.2955754](https://doi.org/10.1109/ACCESS.2019.2955754).
- [35] R. Yacouby and D. Axman, “Probabilistic Extension of Precision, Recall, and F1 Score for More Thorough Evaluation of Classification Models,” in *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, Nov. 2020, pp. 79–91, doi: [10.18653/v1/2020.eval4nlp-1.9](https://doi.org/10.18653/v1/2020.eval4nlp-1.9).
- [36] M. Liebenlito, Y. Irene, and A. Hamid, “Classification of Tuberculosis and Pneumonia in Human Lung Based on Chest X-Ray Image using Convolutional Neural Network,” *Inpr. Indones. J. Pure Appl. Math.*, vol. 2, no. 1, pp. 24–32, Mar. 2020, doi: [10.15408/inprime.v2i1.14545](https://doi.org/10.15408/inprime.v2i1.14545).
- [37] S. Sharma and K. Guleria, “A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks,” *Procedia Comput. Sci.*, vol. 218, pp. 357–366, Jan. 2023, doi: [10.1016/j.procs.2023.01.018](https://doi.org/10.1016/j.procs.2023.01.018).
- [38] E. Ayan, B. Karabulut, and H. M. Ünver, “Diagnosis of Pediatric Pneumonia with Ensemble of Deep Convolutional Neural Networks in Chest X-Ray Images,” *Arab. J. Sci. Eng.*, vol. 47, no. 2, pp. 2123–2139, Feb. 2022, doi: [10.1007/s13369-021-06127-z](https://doi.org/10.1007/s13369-021-06127-z).

Multidisciplinary classification for Indonesian scientific articles abstract using pre-trained BERT model



Antonius Angga Kurniawan ^{a,1,*}, Sarifuddin Madenda ^{a,2}, Setia Wirawan ^{a,3}, Ruddy J Suhatri ^{a,4}

^a Department of Information Technology, Gunadarma University, Depok 16424, Indonesia

¹ anggaku@staff.gunadarma.ac.id; ² sarif@staff.gunadarma.ac.id; ³ setia@staff.gunadarma.ac.id; ⁴ ruddy_js@staff.gunadarma.ac.id

* corresponding author

ARTICLE INFO

Article history

Received March 3, 2023

Revised April 12, 2023

Accepted April 24, 2023

Available online July 8, 2023

Keywords

Abstract

BERT

Classification

Fine-tuned hyperparameter

Multidisciplinary

ABSTRACT

Scientific articles now have multidisciplinary content. These make it difficult for researchers to find out relevant information. Some submissions are irrelevant to the journal's discipline. Categorizing articles and assessing their relevance can aid researchers and journals. Existing research still focuses on single-category predictive outcomes. Therefore, this research takes a new approach by applying a multidisciplinary classification for Indonesian scientific article abstracts using a pre-trained BERT model, showing the relevance between each category in an abstract. The dataset used was 9,000 abstracts with 9 disciplinary categories. On the dataset, text preprocessing is performed. The classification model was built by combining the pre-trained BERT model with Artificial Neural Network. Fine-tuning the hyperparameters is done to determine the most optimal hyperparameter combination for the model. The hyperparameters consist of batch size, learning rate, number of epochs, and data ratio. The best hyperparameter combination is a learning rate of 1e-5, batch size 32, epochs 3, and data ratio 9:1, with a validation accuracy value of 90.8%. The confusion matrix results of the model are compared with the confusion matrix results by experts. In this case, the highest accuracy result obtained by the model is 99.56%. A software prototype used the most accurate model to classify new data, displaying the top two prediction probabilities and the dominant category. This research produces a model that can be used to solve Indonesian text classification-related problems.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



1. Introduction

Nowadays, scientific articles are growing very rapidly with increasingly diverse topics in various disciplines [1]. The scientific articles created have the possibility of containing one discipline and or multidisciplinary at the same time. Akshai and Anitha [2] conducted research for the early detection of plant diseases using deep learning with the Convolutional Neural Network (CNN) model. This example shows that this scientific article belongs to multidisciplinary research because it contains more than one discipline, namely agricultural and computer science.

One of the main concerns of many researchers is finding scientific articles relevant to their discipline and or object of research as scientific literature. This process can be time-consuming. In addition, sometimes, from the many articles submitted to the journal portal, there are still articles whose disciplines differ from those in the journal portal. Therefore, automatically classifying scientific articles into relevant disciplinary categories and knowing the relevance of one discipline to another in a scientific article can be helpful to researchers and journal portals.

Many researches have been conducted related to text classification such as abstract, sentiment, news category classification, and others. However, in previous researches there are still some things that can be developed to be better new research. In general, text classification conducted by previous researchers still uses traditional word embedding models such as Word2Vec, FastText, and TF-IDF [1], [3]–[9]. Currently, an advanced word embedding model provides impressive results in solving natural language processing (NLP) problems, namely the pre-trained BERT model [10]. Most of the text classifications using BERT conducted by previous researchers used English datasets and the research topics were mostly about sentiment or news category classification [11]–[17]. Such researches did not go through the text preprocessing stage on the dataset used [9], [12], [13], where the research stated that the use of text preprocessing can affect the accuracy results obtained by the model. In addition, most of the previous researches only conducted one test of the hyperparameter combination of learning rate, batch size, number of epochs, and data ratio [3], [12], [13], [16], [17]. In text classification, to get a model with the best accuracy, it is necessary to test the hyperparameters more than once or commonly referred to as fine-tuning hyperparameters. In previous research [1], [16], [18]–[21], the text classification carried out only focuses on prediction results with one category without seeing the possibility that a classified text can contain one and or more than one categories.

Based on the results of the search conducted by the previous researchers, pre-trained BERT models applied to Indonesian datasets for the classification of scientific articles based on abstracts is challenging. In addition, this research takes a new approach by proposing the development and implementation of multidisciplinary classification on abstracts of scientific articles in Indonesian using the pre-trained BERT model. The main contribution of this research, the first is to combine the pre-trained BERT model with Artificial Neural Network (ANN) to classify abstracts of scientific articles in Indonesian. The second is to perform text preprocessing such as lowercase text, text cleaning, tokenizing, and stopwords removal on the dataset used. The third is to perform hyperparameter fine-tuning to carry out on the value of the learning rate, batch size, number of epochs, and data ratio to get a combination of hyperparameters with the most optimal model. The fourth, the model with the most optimal results is implemented into a prototype to classify abstracts of scientific articles automatically. In addition, two categories that have the highest probability are displayed, so that the relevance between one discipline and another in a scientific article can be known and can place a scientific article according to the appropriate discipline and or research object.

2. Method

2.1. Research Stages

The method used in this research is described into nine stages: Data Collection, Data Labeling, Text Preprocessing, Data Splitting, Model Architecture Development, Fine-Tuning Hyperparameter Model, Model Evaluation, Model Implementation into Software Prototype, and Testing New Data. Fig. 1 shows the general stages of this research.

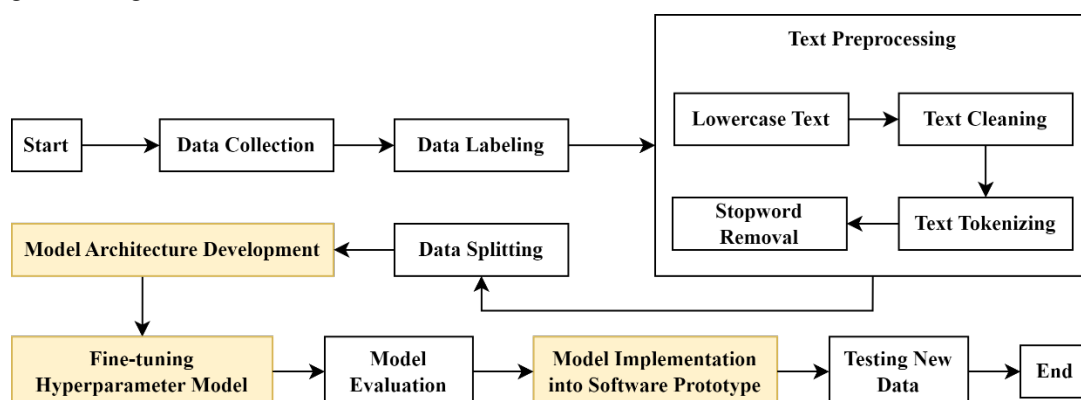


Fig. 1. Research stages

2.2. Data Collection

In this research, the data used are abstracts in the Indonesian language in scientific articles. The abstracts come from national journal portals that have been accredited by Science and Technology Index (SINTA). The abstracts used have been published and have gone through a review process by editors or expert reviewers in each journal.

The abstracts used consist of nine categories of science fields, namely Business, Law, Health, Computer, Communication, Mathematics, Education, Agriculture, and Engineering. These categories refer to several fields of science in accordance with the existing study programs in the formal and applied science clusters based on the Decree of the Minister of Research, Technology and Higher Education in 2019. The amount of data collected was 9,000 abstracts. Fig. 2 shows the number of abstracts in each category is 1,000 abstracts. The data is saved into a file with Comma Separated Value (CSV) format.

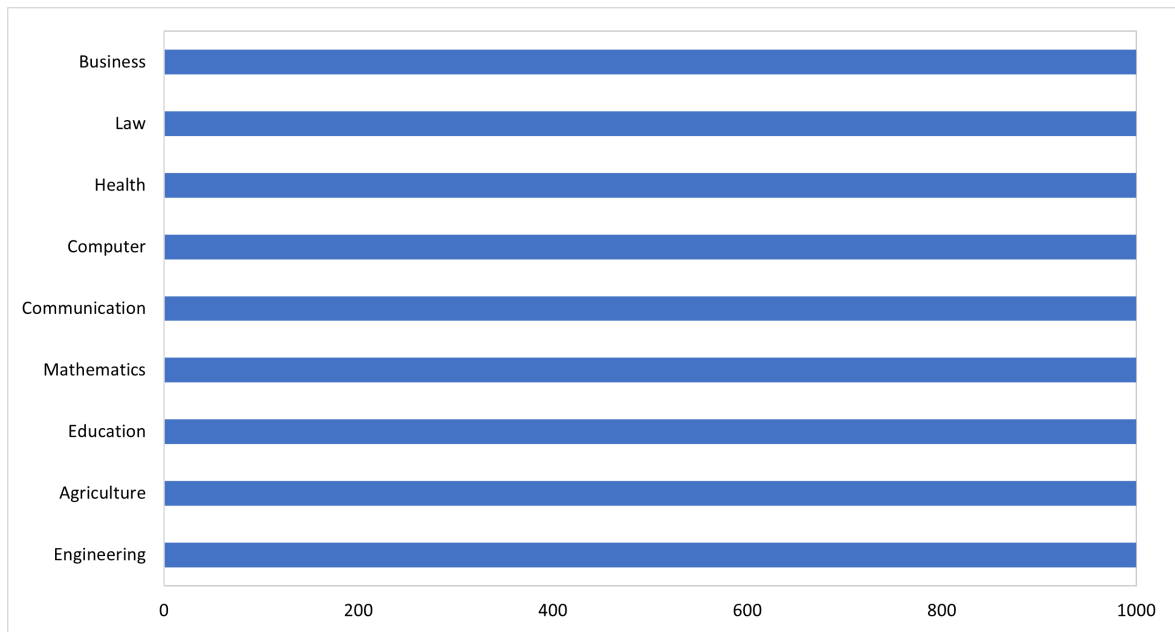


Fig. 2. Total number of abstracts per category

2.3. Data Labeling

The data labeling process is carried out to provide a separate annotation to each abstract into the specified science field category. The label is determined based on the field of science on the journal portal in which the abstract is published. The characteristics of the data are divided into two, namely abstract data with the category of one discipline and multidisciplinary fields of science. Labeling was done by automatically assigning numerical numbers from 0 to 8 to 9,000 abstracts based on their category labels. The labels for each category can be seen in Table 1.

Table 1. Category label

Category	Label
Business	0
Law	1
Health	2
Computer	3
Communication	4
Mathematics	5
Education	6
Agriculture	7
Engineering	8

The validation data is also manually labeled by experts. The validation data will be labeled by experts according to the list of science field categories in Table 1. The labeling process on validation data by

experts allows an abstract to have one and or more than one science field categories. Validation data that has been labeled by experts will become ground truth for the classification process of abstracts of multidisciplinary scientific articles.

2.4. Text Preprocessing

Text preprocessing is one of the key processes of classification, especially in natural language processing (NLP). The steps in text preprocessing aim to improve word structure and to reduce the ambiguity value when extracting features [22], [23]. Text preprocessing in this research consists of lowercase text, text cleaning, text tokenizing, and stopword removal.

2.5. Lowercase Text

The lowercase text process is carried out to convert the entire set of abstract text into lowercase letters. The process is done to help the machine read and process the abstract better. The lowercase text function used is a string function from the pandas python library, namely `str.lower()`. The function will check and read each line of the abstract. If there is a letter that is included in the capital letter it will be immediately changed to lowercase, after that the next line is checked and read until the last line of the abstract.

2.6. Text Cleaning

The process of preparing raw text for NLP so that machines can understand human language is known as text cleaning [22]. Text cleaning of abstract data consists of removing tabs, new lines, multiple spaces, punctuation, special characters, urls, numbers, and single characters.

2.7. Text Tokenizing

The text tokenizing process is carried out to break the text in the form of sentences in the abstract into pieces in the form of tokens, namely words. So that each line in the abstract column contains pieces of tokens that come from each abstract [6]. In this research, a function from NLTK python is used, namely `word_tokenize`. At this stage a machine receives input in the form of text to be tokenized. Next, the text will be separated word by word. After that, the words are converted into an array on each line, then the result of the tokenization process is obtained.

2.8. Stopword Removal

Stopword removal is the process of filtering or selecting words that are important for the display of text. Stopword removal aims to eliminate words that lack meaning or are irrelevant to the data subject used. These words include prepositions, determiners, conjunctions, and other similar words [23]. The stopword removal process is done by using the `nlTK corpus stopwords` library in python. In addition, this study also used a stopword corpus with the Indonesian language created by Devid and Martijn [24] with reference from Tala research [25]. This corpus contains a set of words that are considered not very important or have no meaning in a text or sentence. The input used in the stopword removal process is the result of the text tokenizing process. The stopword remover reads the word and checks word by word whether it is available in the stopword corpus. If so, the word is removed, otherwise, the word is kept.

2.9. Data Splitting

Data that has completed the text preprocessing process will then be separated into two parts, namely training data and validation data. In performing classification, the data used to train the model must be larger than the validation data. The validation data used will validate the model and prevent overfitting. Epoch performs the training and validation process sequentially. When the training is complete, the validation process continues. Therefore, the data used needs to be separated into training data and validation data. In this research, the validation data used will be stored and then manually labeled by experts.

In this research, there is also test data as much as 90 abstract data. Test data is data used to test the model after the training and validation process is complete. The test data is included in the unseen data.

This means that the data has never been seen or recognized by the model during the training process. This test data will be tested using the best model that has been implemented into the prototype. This is done to find out how well the model performs to predict data that has never been seen or recognized before.

2.10. Model Architecture Development

In this research, the Indonesian pre-trained BERT model was used. The model was pre-trained with the Indonesian Wikipedia. The Indonesian pre-trained BERT model used is "cahya/bert-base-indonesian-522M". The model was pre-trained with 522MB of Indonesian Wikipedia that had been converted to lowercase text and tokenized using WordPiece and a vocabulary size of 32,000 [26]. The Transformer-based model is commonly used for text classification, text generation, and others. Fig. 3 shows the architecture of the model built by combining the pre-trained model cahya/bert-base-indonesian-522M and ANN.

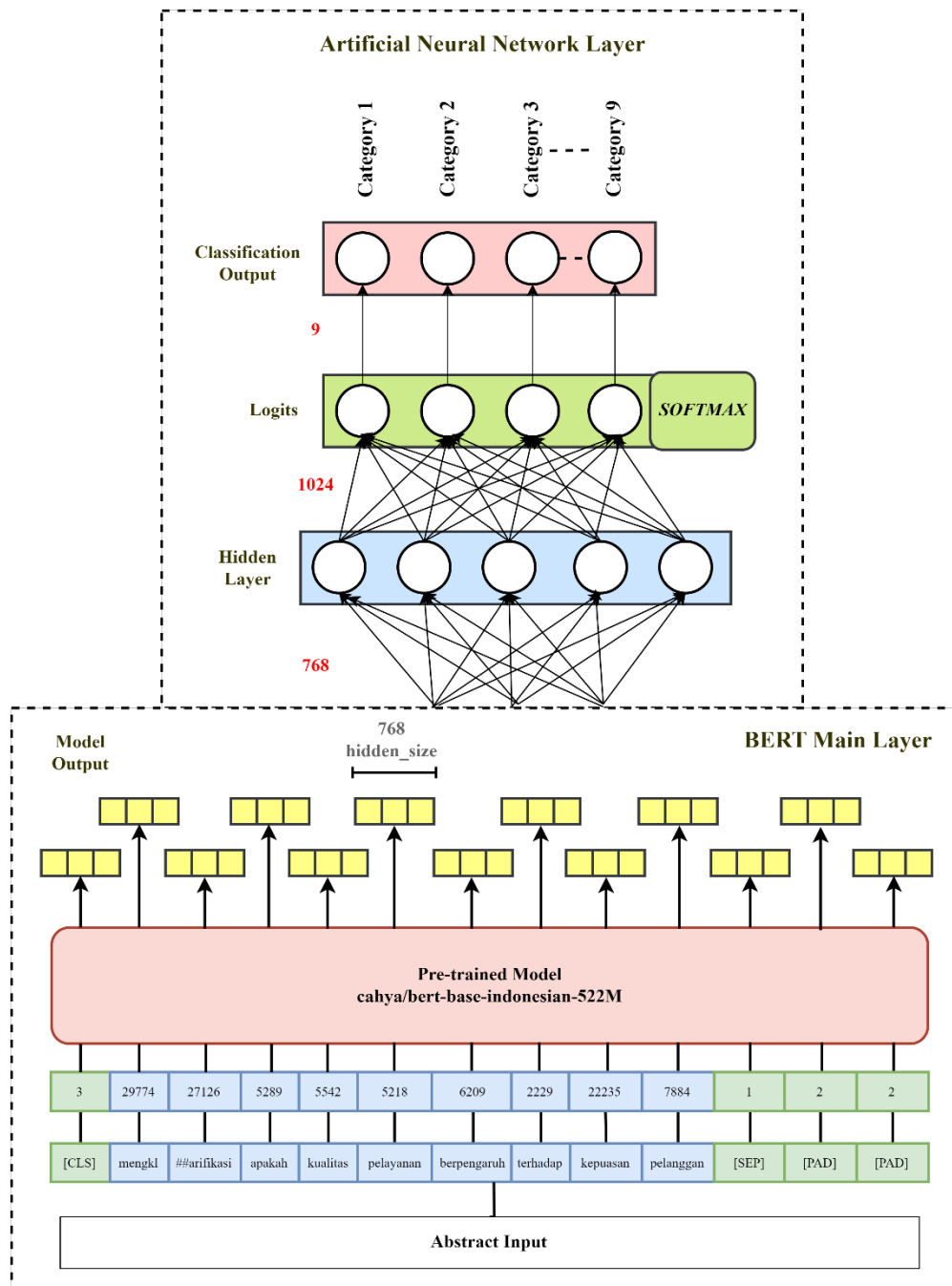


Fig. 3. Model architecture for classification of abstracts of scientific articles

Before the dataset is classified, a sentence or text needs to be converted into the input representation required by BERT. In Fig. 3 there is an example of one sentence to be classified. A special token [CLS] is added to the beginning of the classified sentence to represent the meaning of the whole sentence. Then a special token [SEP] is added at the end of the sentence to separate the first sentence from the next. The record is then set to the initialized maximum length by either truncating the record if it is longer than the maximum length or giving it the special character [PAD] if it is shorter than the maximum length. After that, each text is matched with a unique number or ID contained in the vocabulary and the unique number is stored as a token ID. Each single character, subword, and word has its own special number in the vocabulary. The word or token must be changed using the ID. Based on the example sentence entered, the ID sequence is [3, 29774, 27126, 5289, 5542, 5218, 6209, 2229, 22235, 7884, 1, 2, 2]. The token ID will then proceed through the next stack of encoders. Then self-attention is applied by each encoder and given an output through the feed-forward network. After that it continues into the next encoder. This research uses the Indonesian BERT Base model, so the process takes place 12 times. After all encoders have been successfully passed, the next output is given in the form of a vector of each token according to its position. The given vector has a hidden size of 768 as seen in Fig. 3. The token [CLS] represents the output vector derived from the BERT model. Furthermore, the vector is used as input for abstract classification. Average collection of word tokens is performed by [CLS] with the aim of obtaining vectors derived from sentences. The output layer as the last layer creates logits. In this case, logits are output values in the form of rough probability predictions based on the classified sentence or text. Softmax then converts this logit to a probability value by computing the exponential value of each logit value, so the total probability is exactly 1. In this case, the probability values range from 0 to a positive number.

The architecture of the model built has one input layer, where in the input layer there is also an attention mask. In this case, the attention layer usually uses padding tokens in the context seen for each token in the sentence. To tell the attention layer to ignore token padding, it is necessary to add an attention mask. With the right attention mask, the resulting prediction will be the same for a given sentence, whether using padding or not using padding [27]. In this research, the attention mask is useful to show the model which tokens need attention and which tokens do not. Attention mask will show the model the position of the padding index, so the model does not need to process that token. In this case if there is a sentence whose number of tokens is less than the specified maximum token length, then there will be a [PAD] or padding token.

Next, there is one hidden layer with ReLU activation function. ReLU is used because the output of this layer ranges between 0 and infinity. ReLU significantly speeds up the convergence process performed in stochastic gradient descent compared to Sigmoid or Tanh. Furthermore, ReLU basically only creates a range of zeros. That is, x equals zero if x is less than equal to zero and x equals x if x is greater than zero [28]. The number of neurons or nodes used in the hidden layer is 1024. This is because too few neurons used in the hidden layer can lead to so-called underfitting. Underfitting occurs because there are too few neurons in the hidden layer to detect enough signal in complex datasets. On the other hand, using too many neurons in the hidden layer can lead to overfitting. Overfitting occurs when neural networks are highly intelligent and the training set has insufficient information to train all of the neurons in the hidden layer. Despite having enough training data. The use of a large number of neurons in the hidden layer can lengthen the time it takes to train the network. It can increase training time and prevent the neural network from training properly. Overfitting prevents the model from predicting possible outputs for unknown or untrained inputs, preventing the model from producing accurate outputs [29].

Finally, an output layer useful for classification using a fully connected neural network and Softmax as an activation function with 9 neurons. In terms of the number of neurons determined, it must be equal to the number of class labels or categories used in the data. In this research, the output categories produced are more than two categories of science fields, where the output produced is 9 categories, namely Business, Law, Health, Computer, Communication, Mathematics, Education, Agriculture, and Engineering.

2.11. Fine-Tuning Hyperparameter Model

In the research, four training scenarios were tested. In the training process, fine-tuning of the model was carried out by adjusting four hyperparameters, namely learning rate, batch size, number of epochs, and data ratio with a maximum word length of 150. The four training scenarios can be seen in Table 2.

Table 2. Scenario of hyperparameter tuning

Hyperparameters	Hyperparameter Combination				
	Learning Rate	Batch Size	Epochs	Data Ratio	
Learning Rate	1e-5	32	4	9:1	
	2e-5	32	4	9:1	
	5e-5	32	4	9:1	
Batch Size	Best Learning Rate in the previous scenario	8	4	9:1	
		16	4	9:1	
		32	4	9:1	
Epochs	Best Learning Rate in the previous scenario	Best Batch Size in the previous scenario	64	4	9:1
			2	4	9:1
			3	5	9:1
Data Ratio	Best Learning Rate in the previous scenario	Best Batch Size in the previous scenario	Best Epoch in the previous scenario	9:1	
				8:2	
				7:3	

The first training scenario tests the learning rate values used during the model training process. Learning rate is one of the most important parameters for improving model performance. Learning rate is used to determine the step size at each iteration to get the weight value adjustment with the minimum error in the training process. Choosing a learning rate that is too low and a learning rate that is too high will reduce the performance of the model [30]. Determining the right learning rate value can find the weight for the minimum error value, so that an optimal solution can be achieved in the model. The learning rates tested in the first training scenario are 1e-5, 2e-5, and 5e-5. In addition, testing was conducted using other specified hyperparameters, namely the number of epochs of 4, batch size of 32, and data ratio of 9:1. These other hyperparameters were chosen because based on the results of research conducted by previous researchers using 4 epochs [10], batch size 32 [10], [12] has good accuracy results.

A second training scenario was tested to establish the best batch size in the created model's training procedure. The number of training samples used by the neural network in one iteration (batch) is referred to as batch size. Determining the batch size is important to be adjusted to the model built because the size of the number of samples entering the neural network can determine a more optimal weight value [30]. The testing hyperparameters used consist of the best learning rate obtained from testing in the first training scenario as well as using other predetermined hyperparameters, namely the number of epochs of 4 and the data ratio of 9:1. The batch sizes tested in the second training scenario were 8, 16, 32, and 64.

A third training scenario is tested to determine the effect of the number of epochs on the accuracy validation score of the built model. Epochs are useful for determining the number of times the training process is carried out in the neural network on the entire amount of data. This is quite important because training all data to update and get the most optimal weight value is not enough if only done using one epoch [31]. The hyperparameters for the third training scenario use the best hyperparameters from the learning rate and batch size generated in the previous scenario. In addition, other predetermined hyperparameters such as a data ratio of 9:1 were used. The number of epochs tested in the third training scenario were 2, 3, 4, and 5.

The fourth training scenario was evaluated to see how the ratio of training data and validation data affected the model's accuracy validation value. Training data is used to train an algorithm to identify a

suitable model, and validation data is used to test and determine the performance of the trained model. Comparison of the amount of training data and validation data is quite important during the learning process carried out by the model because it can help ensure that the model created is more accurate. The hyperparameters for the fourth training scenario used the best hyperparameters from the learning rate, batch size, and epoch generated in the previous scenario. In the fourth training scenario, the training data and validation data ratios are 9:1, 8:2, and 7:3.

2.12. Model Evaluation

In this section, the evaluation of the model that has been built is carried out to see the validation value of accuracy, computation time, and confusion matrix from the model training process. Accuracy validation is the value of the accuracy of model predictions obtained during the testing process. Computation time is the time required to train and test the model. A confusion matrix is an evaluation method for measuring the performance or accuracy of a model produced in a classification process [32].

The validation data that has been labeled by experts is then converted into a confusion matrix. The confusion matrix generated by the model is then compared with the confusion matrix by the expert. This aims to validate the classification suitability results produced by the model both inside and outside the diagonal line of the confusion matrix due to the possibility of abstracts that are multidisciplinary.

2.13. Model Implementation into Prototype

In this research, the model that has been built, trained, and evaluated with the most optimal evaluation results will be saved. The model will be implemented into a web-based software prototype to classify new input data into abstract categories that are more dominant than the 9 predetermined categories. The prototype software was built using a Web Server Gateway Interface (WSGI) application framework, Flask. The programming language used is python and the interface used is html.

2.14. Testing New Data

Testing is done by processing new abstract data that has never been trained or recognized by the model as much as 90 abstract data. In this study, the same steps as the training process were carried out, namely text preprocessing, then the new abstract data was used as input to be processed into the most optimal model that had previously been stored. The model will classify the input text into one of the more dominant categories from the 9 initialized abstract label categories.

In the prototype built, an input form will appear. Next, enter the abstract of the new text you want to predict. After entering the abstract, press Enter. Next, the model will process the classification results of the newly entered abstract. This process also performs a text preprocessing process and then will display the prediction results of the science field category based on the abstract entered. Finally, the results of the prediction probability values obtained from the model are shown for the two categories with the highest probabilities.

3. Results and Discussion

3.1. Example of Data Labeling Results by Experts

An example of the results of data labeling by experts on validation data can be seen in [Table 3](#). In abstract number 1 it can be said that the abstract is likely to fall into the multidisciplinary category. This is because in abstract number 1 there is a relevance between the fields of Engineering and Agriculture, where in the abstract the object of research is related to Engineering and the scientific field is Agriculture. In abstract number 2 it can be said that the abstract is likely to fall into the category of one discipline. This is because in abstract number 2 the scientific field and the object of research carried out are only related to the field of Law.

3.2. Model Evaluation Results

In this research, hyperparameter fine-tuning is performed on the model using four training scenarios. This aims to get the most optimal hyperparameter combination for the model built. The hyperparameters used from the four training scenarios consist of learning rate, batch size, number of epochs, and number of data ratios. The first training scenario tested the value of the learning rate consisting of $1e-5$, $2e-5$, and $5e-5$. Table 4 displays the model's results for the first training scenario.

Table 3. Examples of abstract data with single-discipline and multidiscipline categories

No	Abstract	Category
1	<p><i>Pada umumnya, pembasmian hama padi dilakukan dengan cara penyemprotan pestisida. Hal ini akan mengakibatkan tanah dan tanaman padi tercemar. Penelitian ini bertujuan untuk membuat alat pembasmi hama otomatis yang ramah lingkungan tanpa menggunakan pestisida. ... Pada perancangan ini, mikrokontroler Atmega 328 Arduino UNO digunakan sebagai pusat pengendali sistem, sensor LDR digunakan sebagai pengganti sakelar lampu DC di malam hari untuk membuat hama mendekat sesuai dengan karakteristiknya yang tertarik dengan cahaya, ... Pemrograman menggunakan software Arduino IDE. Berdasarkan hasil pengujian selama 3 hari, alat pembasmi hama padi ini dapat membasmi 39 hama kepik hitam, penggerek batang padi, dan walang sangit.</i></p> <p>English Translation:</p> <p>In general, the eradication of rice pests is done by spraying pesticides. This will result in contaminated soil and rice plants. This research aims to make an automatic pest exterminator tool that is environmentally friendly without using pesticides. ... In this design, the Atmega 328 Arduino UNO microcontroller is used as the system control center, the LDR sensor is used as a substitute for a DC light switch at night to make pests approach according to their characteristics that are attracted to light, ... Programming using Arduino IDE software. Based on the results of testing for 3 days, this rice pest exterminator can eradicate black ladybugs, rice stem borers, and stink bugs.</p>	Engineering and Agriculture (Multidiscipline)
2	<p><i>Praperadilan merupakan wewenang pengadilan negeri untuk memeriksa dan memutus menurut cara yang diatur dalam undang-undang, tentang sah atau tidaknya suatu penangkapan ke pengadilan. ... Penelitian ini menggunakan metode penelitian hukum normatif yakni adanya kekosongan norma hukum di dalam Pasal 21 ayat (3) KUHAP dan Pasal 46 ayat (3) Peraturan Kapolri Nomor 14 Tahun 2012 ... Hasil dari penelitian ini adalah mekanisme penyidik Kepolisian dalam melakukan penahanan kepada tersangka kepada keluarga tersangka. Akibat hukum bagi penyidik Kepolisian yang belum menyampaikan tembusan surat perintah tersebut dapat dijadikan dasar oleh pihak keluarga untuk menyatakan bahwa penahanan tersebut tidak sah karena telah melanggar hak asasi atau kebebasan hidup seseorang.</i></p> <p>English Translation:</p> <p>Pretrial is the authority of the district court to examine and decide in the manner provided for in the law, about the legality or not of an arrest to the court. ... This research uses normative legal research methods, namely the existence of a legal norm vacuum in Article 21 paragraph (3) of the Criminal Procedure Code and Article 46 paragraph (3) of the National Police Chief Regulation Number 14 of 2012 ... The result of this research is the mechanism of Police investigators in detaining a suspect to the suspect's family. The legal consequences for Police investigators who have not submitted a copy of the warrant can be used as a basis by the family to declare that the detention is invalid because it has violated the human rights or freedom of life of a person.</p>	Law (One-discipline)

Table 4 shows that different learning rates can affect model training results. The highest accuracy validation value is obtained when the learning rate used is $1e-5$. In addition, in Table 4 it can also be seen that the higher the value of the learning rate, the faster the time required, but the lower the validation accuracy. This is because the smaller the learning rate value, the smaller the change in gradient descent, so the possibility of finding the weight value with the minimum error will be greater. This can

make the model more precise and can achieve a more optimal accuracy validation value. However, a smaller learning rate value requires a longer time to reach the most optimal solution. This can be shown in Table 4. However, considering the insignificant time difference between learning rate 1e-5 and 2e-5 and the higher validation accuracy value at learning rate 1e-5, it was decided to set learning rate 1e-5 as the best learning rate value in the first training scenario.

The second training scenario was tested to determine the optimal batch size for the model. The batch sizes tested in the second training scenario were 8, 16, 32, and 64. The learning rate hyperparameter used is the best learning rate obtained from testing in the first training scenario, which is 1e-5. Table 5 displays the model's results for the second training scenario.

Table 4. First Scenario Testing Results - Learning Rate

Learning Rate	Batch Size	Epochs	Data Ratio	Validation Accuracy	Time
1e-5	32	4	9:1	90%	16min 44s
2e-5	32	4	9:1	88.1%	16min 31s
5e-5	32	4	9:1	87.3%	15min 21s

According to the results in Table 5, the higher the batch size, the shorter the time required for training. In the second training scenario, the best validation accuracy is achieved when the batch size used is 32. This is because the larger the batch size used can help reduce noise in error calculation. However, a batch size that is too large also does not guarantee that the accuracy of the model will be better, because the number of samples used in one iteration becomes smaller.

Table 5. Second Scenario Testing Results – Batch Size

Learning Rate	Batch Size	Epochs	Data Ratio	Validation Accuracy	Time
1e-5	8	4	9:1	86.8%	25min 16s
1e-5	16	4	9:1	89.4%	20min 3s
1e-5	32	4	9:1	90%	16min 44s
1e-5	64	4	9:1	89.2%	13min 45s

A third training scenario is tested to determine the effect of the number of epochs on the accuracy validation score of the model. The hyperparameters for the third training scenario use the optimal batch size and learning rate obtained from testing in the previous scenario, namely a learning rate of 1e-5 and a batch size of 32. The number of epochs tested in the third training scenario, namely 2, 3, 4, and 5. Results for his third training scenario of the model are shown in Table 6.

Table 6. Third Scenario Testing Results – Epochs

Learning Rate	Batch Size	Epochs	Data Ratio	Validation Accuracy	Time
1e-5	32	2	9:1	89.7%	9min 17s
1e-5	32	3	9:1	90.8%	12min 45s
1e-5	32	4	9:1	90%	16min 44s
1e-5	32	5	9:1	89.2%	21min 3s

The results in Table 6 show that the higher the number of epochs, the longer the training process takes. The optimal number of epochs for a model built with the highest accuracy validation value is 3 epochs. The more the number of epochs, the more weight values will be updated. So the possibility to get the most optimal weight value will be greater. However, the number of epochs that are too many is

also not necessarily good for the model built, because at an epoch of 5 the accuracy validation value has decreased. This is because the number of epochs depends on the amount of data used during training.

The fourth training scenario was evaluated to see how the ratio of training data and validation data affected the model's validation accuracy value. Hyperparameters for the fourth training scenario use the best batch size, learning rate, and epoch resulting from testing in the first to third scenarios, namely learning rate $1e-5$, batch size 32, and number of epochs of 3. In the fourth training scenario, the training data and validation data ratios were 9:1, 8:2, and 7:3. Results for his fourth training scenario for the model are shown in [Table 7](#).

Table 7. Fourth Scenario Testing Results – Data Ratio

Learning Rate	Batch Size	Epochs	Data Ratio	Validation Accuracy	Time
1e-5	32	3	9:1	90.8%	12min 45s
1e-5	32	3	8:2	89.3%	12min 21s
1e-5	32	3	7:3	89.9%	11min 36s

Based on the results in [Table 7](#), it appears that the more training data that is used, the longer the training process takes. The highest accuracy validation result obtained is when the data ratio used is 9:1 with an accuracy validation of 90.8%.

Based on four training scenarios that have been carried out on the model, a combination of hyperparameters with the highest accuracy validation value is obtained. The combination consists of hyperparameters with a learning rate of $1e-5$, a number of epochs of 3, a batch size of 32, and a data ratio of 9:1. This hyperparameter combination resulted in an accuracy validation of 90.8% with a computation time of 12 minutes 45 seconds. The model with the best hyperparameter combination is named the AbBERT model (Abstract BERT).

This research has also tested several traditional machine learning models and deep learning models based on models used in previous studies such as SVM [1], [4], CNN [7], [16], LSTM [8], and other models such as Naive Bayes, Logistic Regression, KNN, Random Forest. Testing of these models is done using the same dataset and architecture as the AbBERT model. [Table 8](#) shows the results of accuracy validation on each of these models.

Table 8. Accuracy Validation Results of Several Machine Learning and Deep Learning Models

Model	Validation Accuracy
Naïve Bayes	87.5%
SVM	89.3%
Logistic Regression	88.3%
KNN	77.7%
Random Forest	86.1%
CNN	84.4%
LSTM	76.7%

The amount and data used for classification using several machine learning and deep learning models are the same as the AbBERT model, which is 9,000 abstract data. The ratio of training data and validation data used is 9:1 with a maximum word length of 150. Then for CNN and LSTM models using the same hyperparameters as the best hyperparameters generated in the previous four training scenarios, namely batch size of 32, the number of epochs of 3, learning rate of $1e-5$. According to the results in [Table 8](#), the SVM model has the highest accuracy validation value, with an accuracy validation of 89.3%. However, these results are still lower when compared to the accuracy validation value of the AbBERT model.

Next, the confusion matrix results of the AbBERT model with an accuracy validation value of 90.8% can be seen in [Fig. 4](#). Based on the results in the confusion matrix, it can be seen that of the 900 data used as validation data, there are 817 or about 91% of the data that is predicted correctly. So it can be

said that the model that has been built can classify validation data into each category quite well. In addition, from the confusion matrix, it can also be seen that there are 83 or around 9% of data that are predicted to be less in accordance with the category label.

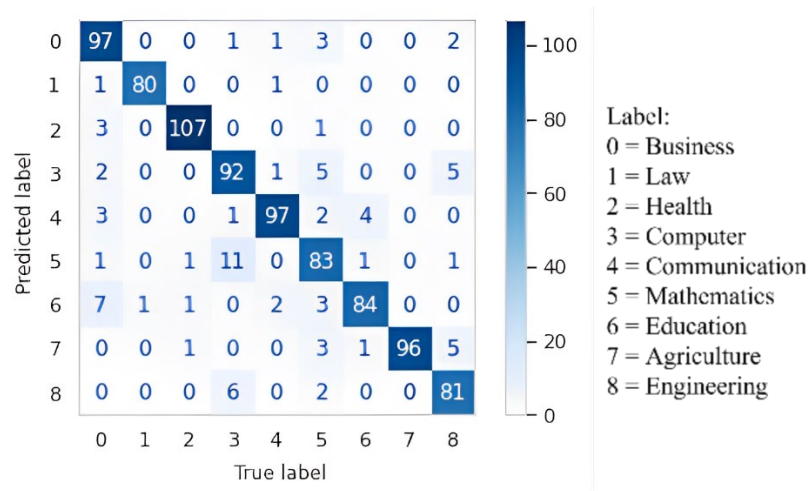


Fig. 4. AbBERT model confusion matrix results

In this research, the labeling of validation data that has been carried out by experts is then poured into a confusion matrix as shown in Fig. 5.

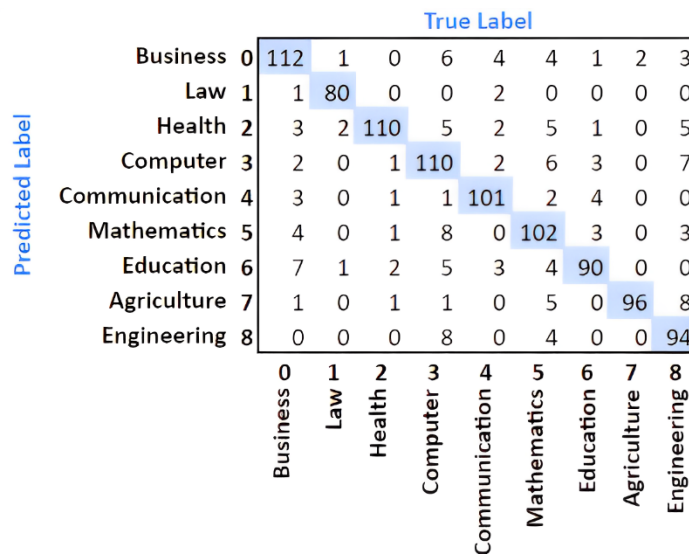


Fig. 5. Confusion matrix results by experts

Fig. 5 shows that there is abstract data that has a relevance between one category and another. This can be seen in the confusion matrix, where the sum exceeds the amount of validation data given, so it can be said that there is data that is categorized with one category and or more than one category by experts. Then when compared with the confusion matrix generated by the AbBERT model as shown in Fig. 4, there are similarities in the off-diagonal part. Although in the confusion matrix by the expert there are more data categorized outside the diagonal compared to the confusion matrix by the AbBERT model, there is a possibility that the data is predicted by the model as a category within the diagonal line. This cannot be said to be wrong, because the data outside the diagonal is likely to be abstract with multidisciplinary categories. This shows that the AbBERT model is not completely wrong in predicting 83 abstract data or 9% of the data mentioned earlier. This is because there is a relationship between the categories of science in an abstract that is tested. So that the AbBERT model will display predictions according to the more dominant category.

However, there are still 4 abstract data that are not predicted correctly by the model. This can be seen in the Computer and Education categories. Where there are 11 abstract data predicted as Mathematics in the Computer category, while based on labeling by experts, there are only 8 abstracts that can be categorized as Mathematics. Then, in the Education category there is 1 data predicted as Agriculture, while in the confusion matrix by experts, there is no data that can be categorized as Agriculture in the Education category. So it can be said that there are only 4 data predicted incorrectly by the model, or it can be concluded that the highest accuracy value of the AbBERT model is 99.56%. Prediction errors can occur due to words that contextually have the same meaning but are also found in several different categories, or because there are no words in the abstract that have information values that match the category to be used as a reference by the model in classification.

This research produces the AbBERT model, which is a development of the pre-trained BERT model combined with the ANN deep learning model. Fine-tuning hyperparameters has been done on the AbBERT model so that the model can produce more optimal accuracy values for text classification in Indonesian natural language processing. In this case the AbBERT model can be used to solve other problems related to text classification in Indonesian language to be even better.

3.3. Results of Model Implementation into Prototype and New Abstract Data Testing

In this research, the AbBERT model was successfully implemented into a prototype to classify new abstract data. In the prototype there is an input form that is used to input the abstract text that you want to classify. The abstract is processed into the text preprocessing stage which consists of lowercase text, text cleaning, tokenizing, and stopword removal. After text preprocessing, the abstract will be forwarded to the AbBERT model or the best model that was previously stored. The model will process the abstract to be classified into disciplinary categories that correspond to the nine predetermined categories. An example of the abstract classification results using the prototype is shown in Fig. 6.

Abstract Classification Results

Abstract

Ganyong adalah tanaman pangan yang mempunyai kandungan gizi cukup tinggi. Tujuan penelitian yaitu untuk mengetahui cara pembuatan bioetanol dari umbi ganyong dengan metode Solid State Fermentation (SSF) dan menentukan harga konstanta Michaelis-Menten pada pembuatan bioetanol dari umbi ganyong dalam pH bervariasi menggunakan metode SSF. Umbi ganyong dicuci dengan air, dikupas kulitnya, dan diparut. Dari hasil parutan diperas dan dipanaskan dalam panci dengan suhu 100oC selama ±30 menit. Bubur umbi ganyong didinginkan untuk dilanjutkan proses fermentasi. Bubur umbi ganyong diambil sebanyak 250 mL, lalu ditambahkan Saccharomyces cerevisiae sebanyak 10 g, NPK sebanyak 5 g, urea sebanyak 5 g, dan volume starter 200 mL. Bubur umbi dimasukkan ke dalam tempat fermentasi dan ditutup agar tidak terjadi kontak langsung dengan udara. Sampel dianalisis pada pH 4, 4,5 dan 5 dengan waktu fermentasi selama 0, 3, 5, 7, 9, 11, 13, 15, dan 17 hari. Berdasarkan hasil penelitian, kadar air, serat kasar, dan pati yang terkandung dalam umbi ganyong berturut-turut adalah 10,10 %, 40,17 %, dan 0,35 %. Kadar glukosa tertinggi dalam penelitian ini pada waktu fermentasi 0 hari dengan pH 5 sebesar 8,2 %. Sedangkan kadar bioetanol tertinggi sebesar 46,08% pada pH 4 pada waktu fermentasi 17 hari. Model kinetika reaksi fermentasi yang paling sesuai adalah Lineweaver and Burk pada pH 5 dengan harga Km sebesar 94,26 g/L.hari dan Vmaks sebesar 30,66 g/L dengan R2 sebesar 0,98.

English Translation:

Canna is a food plant that has a fairly high nutritional content. The research objectives were to find out how to produce bioethanol from canna tubers using the Solid State Fermentation (SSF) method and to determine the Michaelis-Menten constant for the production of bioethanol from canna tubers at varying pH using the SSF method. Canna tubers are washed with water, peeled, and grated. The grated results are squeezed and heated in a pan with a temperature of 100oC for ± 30 minutes. The canna tuber pulp is cooled to continue the fermentation process. As much as 250 mL of canna tuber pulp was taken, then 10 g of Saccharomyces cerevisiae was added, 5 g of NPK, 5 g of urea, and 200 mL of starter volume. The tuber pulp is put into the fermentation area and covered so that there is no direct contact with air. Samples were analyzed at pH 4, 4.5 and 5 with fermentation times of 0, 3, 5, 7, 9, 11, 13, 15 and 17 days. Based on the research results, the water content, crude fiber, and starch contained in canna tubers were 10.10%, 40.17% and 0.35%, respectively. The highest glucose level in this study was at 0 days of fermentation with a pH of 5 of 8.2%. While the highest bioethanol content was 46.08% at pH 4 during 17 days of fermentation. The most suitable fermentation reaction kinetics model is Lineweaver and Burk at pH 5 with a Km value of 94.26 g/L/day and a Vmax of 30.66 g/L with an R2 of 0.98.

Label Description

Business: 0; Law: 1; Health: 2; Computer: 3; Communication: 4; Mathematics: 5; Education: 6; Agriculture: 7; Engineering: 8

Two Highest Probability Results:

Label	Category	Probability
8	Engineering	0.5964999198913574
7	Agriculture	0.383487731218338

Abstract prediction results are more dominant to:

8 - Engineering

Probability

0.5964999198913574

Fig. 6. An example of test results with new abstract data

The results of the abstract classification are displayed in the form of the abstract text, the two highest probability results based on the nine predetermined categories, then the prediction results of the category of the most dominant abstract. Thus, based on the results displayed, it can be determined the tendency of a scientific article to the discipline and or research object. The two highest probability results displayed can see the possibility of a scientific article belonging to a multidisciplinary scientific article.

A total of 90 new abstract data that has never been trained or recognized is used as input to test the AbBERT model. Based on the results obtained, it is possible to conclude that the model developed has

a high level of accuracy. This is evidenced by testing using 90 new abstract data getting prediction results in accordance with each discipline as much as 86 abstract data. There are 4 abstracts that get different prediction results from the specified discipline category. In this case it does not mean that the model used has inaccurate accuracy, but in this case, it shows that the model built can recognize or find out new abstract data that is classified as having a higher probability in accordance with the scientific discipline and / or object of research. Therefore, it can be said that the abstract covers multidisciplinary or belongs to a multidisciplinary scientific article.

4. Conclusion

Classification of Indonesian scientific articles based on abstracts was successfully carried out. The classification successfully displayed the prediction probability value of classification for each category, especially the two categories with the highest probability value. The results found that the most suitable hyperparameter combination consisted of a batch size of 32, a number of epochs of 3, a learning rate of $1e-5$, and a ratio of training and test data of 9:1. The model with this hyperparameter combination resulted in an accuracy validation value of 90.8% and was then named the AbBERT (Abstract BERT) model. When compared to other models such as Naive Bayes, SVM, Logistic Regression, KNN, Random Forest, CNN, and LSTM, the AbBERT model built has evaluation results with higher accuracy validation values. The confusion matrix generated by the AbBERT model is then compared with the confusion matrix created based on the labeling of validation data by experts. Based on the analysis, it is known that the AbBERT model is not completely wrong in predicting data outside the diagonal line of the confusion matrix. This is due to the relationship between the categories of science fields in an abstract that is tested. The highest accuracy value that the AbBERT model can get in this case is 99.56%.

The implementation of the AbBERT model into a software prototype was successful. In this prototype, new abstract data that has never been trained or recognized can be used as input to be classified based on its category. The result of the classification will be displayed not only the highest prediction of a category, but also the two highest classification prediction probability values based on the specified category. However, because the prototype is built with an Indonesian-based model, so that in the prototype the abstracts of scientific articles that can be classified are only Indonesian abstracts. Therefore, in future research, a prototype should be built with a model that has been trained with many languages so that the prototype can classify abstracts in various languages. The number of datasets and categories used can be enriched and more varied so that the training process carried out by the model is even better. In addition, in future research, the addition of hidden layers and a more varied number of neurons can be tested in order to find out the comparison of the accuracy validation results obtained by the model.

Acknowledgment

The authors would like to thank the Department of Information Technology, Gunadarma University, which has supported this research.

Declarations

Author contribution. All authors contributed equally to the paper's main contributor. The final paper was read and approved by all authors.

Funding statement. The funding agency should be spelled out completely, followed by the grant number in square brackets and the year.

Conflict of interest. The authors declare that they have no conflicts of interest.

Additional information. There is no additional information for this paper.

References

- [1] F. R. Lumbanraja, E. Fitri, Ardiansyah, A. Junaidi, and R. Prabowo, "Abstract Classification Using Support Vector Machine Algorithm (Case Study: Abstract in a Computer Science Journal)," *J. Phys. Conf. Ser.*, vol. 1751, no. 1, pp. 0–12, 2021, doi: [10.1088/1742-6596/1751/1/012042](https://doi.org/10.1088/1742-6596/1751/1/012042).

- [2] A. KP and J. Anitha, "Plant disease classification using deep learning," in *2021 3rd International Conference on Signal Processing and Communication (ICSPSC)*, May 2021, pp. 407–411, doi: [10.1109/ICSPSC51351.2021.9451696](https://doi.org/10.1109/ICSPSC51351.2021.9451696).
- [3] I. N. Khasanah, "Sentiment Classification Using fastText Embedding and Deep Learning Model," *Procedia CIRP*, vol. 189, pp. 343–350, 2021, doi: [10.1016/j.procs.2021.05.103](https://doi.org/10.1016/j.procs.2021.05.103).
- [4] I. M. Fadhil and Y. Sibaroni, "Topic Classification in Indonesian-language Tweets using Fast-Text Feature Expansion with Support Vector Machine (SVM)," in *2022 International Conference on Data Science and Its Applications (ICoDSA)*, Jul. 2022, pp. 214–219, doi: [10.1109/ICoDSA55874.2022.9862899](https://doi.org/10.1109/ICoDSA55874.2022.9862899).
- [5] R. Adipradana, B. P. Nayoga, R. Suryadi, and D. Suhartono, "Hoax analyzer for Indonesian news using RNNs with fasttext and glove embeddings," *Bull. Electr. Eng. Informatics*, vol. 10, no. 4, pp. 2130–2136, Aug. 2021, doi: [10.11591/eei.v10i4.2956](https://doi.org/10.11591/eei.v10i4.2956).
- [6] R. Kusumaningrum, I. Z. Nisa, R. P. Nawangsari, and A. Wibowo, "Sentiment analysis of Indonesian hotel reviews: from classical machine learning to deep learning," *Int. J. Adv. Intell. Informatics*, vol. 7, no. 3, pp. 292–303, Nov. 2021, doi: [10.26555/ijain.v7i3.737](https://doi.org/10.26555/ijain.v7i3.737).
- [7] M. S. David and S. Renjith, "Comparison of word embeddings in text classification based on RNN and CNN," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1187, no. 1, p. 012029, Sep. 2021, doi: [10.1088/1757-899X/1187/1/012029](https://doi.org/10.1088/1757-899X/1187/1/012029).
- [8] M. I. Alfarizi, L. Syafaah, and M. Lestandy, "Emotional Text Classification Using TF-IDF (Term Frequency-Inverse Document Frequency) And LSTM (Long Short-Term Memory)," *JUITA J. Inform.*, vol. 10, no. 2, p. 225, Nov. 2022, doi: [10.30595/juita.v10i2.13262](https://doi.org/10.30595/juita.v10i2.13262).
- [9] K. Boonchuay, "Sentiment Classification Using Text Embedding for Thai Teaching Evaluation," *Appl. Mech. Mater.*, vol. 886, pp. 221–226, Jan. 2019, doi: [10.4028/www.scientific.net/AMM.886.221](https://doi.org/10.4028/www.scientific.net/AMM.886.221).
- [10] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Naacl-Hlt 2019*, no. M1m, pp. 1-16, 2019, doi: [10.48550/arXiv.1810.04805](https://doi.org/10.48550/arXiv.1810.04805).
- [11] M. Munikar, S. Shakya, and A. Shrestha, "Fine-grained Sentiment Classification using BERT," *Int. Conf. Artif. Intell. Transform. Bus. Soc. AITB 2019*, vol. 1, pp. 1–5, 2019, doi: [10.1109/AITB48515.2019.8947435](https://doi.org/10.1109/AITB48515.2019.8947435).
- [12] S. Abdul, Y. Qiang, S. Basit, and W. Ahmad, "Using BERT for Checking the Polarity of Movie Reviews," *Int. J. Comput. Appl.*, vol. 177, no. 21, pp. 37–41, 2019, doi: [10.5120/ijca2019919675](https://doi.org/10.5120/ijca2019919675).
- [13] W. Maharani, "Sentiment Analysis during Jakarta Flood for Emergency Responses and Situational Awareness in Disaster Management using BERT," *2020 8th Int. Conf. Inf. Commun. Technol. ICoICT 2020*, pp. 1–5, 2020, doi: [10.1109/ICoICT49345.2020.9166407](https://doi.org/10.1109/ICoICT49345.2020.9166407).
- [14] J. Ravi and S. Kulkarni, "Text embedding techniques for efficient clustering of twitter data," *Evol. Intell.*, pp. 1-11, Feb. 2023, doi: [10.1007/s12065-023-00825-3](https://doi.org/10.1007/s12065-023-00825-3).
- [15] M. Khadhraoui, H. Bellaaj, M. Ben Ammar, H. Hamam, and M. Jmaiel, "Survey of BERT-Base Models for Scientific Text Classification: COVID-19 Case Study," *Appl. Sci.*, vol. 12, no. 6, p. 2891, Mar. 2022, doi: [10.3390/app12062891](https://doi.org/10.3390/app12062891).
- [16] X. Chen, P. Cong, and S. Lv, "A Long-Text Classification Method of Chinese News Based on BERT and CNN," *IEEE Access*, vol. 10, pp. 34046–34057, 2022, doi: [10.1109/ACCESS.2022.3162614](https://doi.org/10.1109/ACCESS.2022.3162614).
- [17] G. Danilov, T. Ishankulov, K. Kotik, Y. Orlov, M. Shifrin, and A. Potapov, "The Classification of Short Scientific Texts Using Pretrained BERT Model," vol. 281, pp. 83–87, 2021, doi: [10.3233/SHTI210125](https://doi.org/10.3233/SHTI210125).
- [18] I. M. Rabbimov and S. S. Kobilov, "Multi-Class Text Classification of Uzbek News Articles using Machine Learning," in *Journal of Physics: Conference Series*, May 2020, vol. 1546, no. 1, pp. 012097, doi: [10.1088/1742-6596/1546/1/012097](https://doi.org/10.1088/1742-6596/1546/1/012097).
- [19] A. Bogdanchikov, D. Ayazbayev, and I. Varlamis, "Classification of Scientific Documents in the Kazakh Language Using Deep Neural Networks and a Fusion of Images and Text," *Big Data Cogn. Comput.*, vol. 6, no. 4, p. 123, Oct. 2022, doi: [10.3390/bdcc6040123](https://doi.org/10.3390/bdcc6040123).

- [20] A. Barua, O. Sharif, and M. M. Hoque, "Multi-class Sports News Categorization using Machine Learning Techniques: Resource Creation and Evaluation," *Procedia Comput. Sci.*, vol. 193, pp. 112–121, 2021, doi: [10.1016/j.procs.2021.11.002](https://doi.org/10.1016/j.procs.2021.11.002).
- [21] D. Ali, M. M. S. Missen, and M. Husnain, "Multiclass Event Classification from Text," *Sci. Program.*, vol. 2021, pp. 1–15, Jan. 2021, doi: [10.1155/2021/6660651](https://doi.org/10.1155/2021/6660651).
- [22] Y. A. Putra and M. L. Khodra, "Deep learning and distributional semantic model for Indonesian tweet categorization," in *2016 International Conference on Data and Software Engineering (ICoDSE)*, 2016, pp. 1–6, doi: [10.1109/ICODSE.2016.7936108](https://doi.org/10.1109/ICODSE.2016.7936108).
- [23] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manag.*, vol. 50, no. 1, pp. 104–112, Jan. 2014, doi: [10.1016/j.ipm.2013.08.006](https://doi.org/10.1016/j.ipm.2013.08.006).
- [24] D. Haryalesmana and M. Wieriks, "Indonesian Stopword Corpus," 2016. [Online]. Available at: <https://github.com/masdevid/ID-Stopwords>
- [25] F. Z. Tala, "A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia," Institute for Logic, Language and Computation. Universiteit van Amsterdam, The Netherlands., 2003. [Online]. Available at: <https://eprints.ilc.uva.nl/id/eprint/740/1/MoL-2003-02.text.pdf>.
- [26] W. C, "BERT-base-indonesian-522M," *Hugging Face*, 2021. [Online]. Available at: <https://huggingface.co/cahya/bert-base-indonesian-522M>.
- [27] A. Vaswani *et al.*, "Attention is all you need," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010, doi: [10.48550/arXiv.1706.03762](https://doi.org/10.48550/arXiv.1706.03762).
- [28] Y. Bai, "RELU-Function and Derived Function Review," *SHS Web Conf.*, vol. 144, p. 02006, Aug. 2022, doi: [10.1051/shsconf/202214402006](https://doi.org/10.1051/shsconf/202214402006).
- [29] S. Pothuganti, "Review on over-fitting and under-fitting problems in Machine Learning and solutions," *Int. J. Adv. Res. Electr. Electron. Instrum. Eng.*, vol. 7, pp. 3692–3695, Sep. 2018. [Online]. Available at: http://www.ijareeie.com/upload/2018/september/11A_PS_NC.PDF.
- [30] I. Kandel and M. Castelli, "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset," *ICT Express*, vol. 6, no. 4, pp. 312–315, Dec. 2020, doi: [10.1016/j.icte.2020.04.010](https://doi.org/10.1016/j.icte.2020.04.010).
- [31] H. Jindal, N. Sardana, and R. Mehta, "Analyzing Performance of Deep Learning Techniques for Web Navigation Prediction," *Procedia Comput. Sci.*, vol. 167, pp. 1739–1748, 2020, doi: [10.1016/j.procs.2020.03.384](https://doi.org/10.1016/j.procs.2020.03.384).
- [32] M. Hossin and M. N. Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations," *Int. J. Data Min. Knowl. Manag. Process.*, vol. 5, no. 2, pp. 01–11, Mar. 2015, doi: [10.5121/ijdkp.2015.5201](https://doi.org/10.5121/ijdkp.2015.5201).