Predicting success of the online social network user by graph neural networks

Fail Gafarov [a,1,*], Andrey Berdnikov [a,2], Pavel Ustin [a,3]

[a] Kazan Federal University, 18, Kremlyovskaya street. Kazan, Russian Federation
[1] fgafarov@yandex.ru ; [2] andreyberdnikov1998@gmail.com; [3] pavust@mail.ru
* corresponding author

ARTICLE INFO

Article history
Received June 28, 2022
Revised July 31, 2022
Accepted August 1, 2022
Available online November 30, 2022

Keywords
Online social network
Graph neural networks
Professional success
Graph convolutional neural networks
Social graph

ABSTRACT

Online social networks provide rich information that characterizes the user's personality, his interests, hobbies, and reflects his current state. Users of social networks publish photos, posts, videos, audio, etc. every day. Online social networks (OSN) open up a wide range of research opportunities for scientists. Much research conducted in recent years using graph neural networks (GNN) has shown their advantages over conventional deep learning. In particular, the use of graph neural networks for online social network analysis seems to be the most suitable. In this article we studied the use of graph convolutional neural networks with different convolution layers (GCNConv, SAGEConv, GraphConv, GATConv, TransformerConv, GINConv) for predicting the user's professional success in VKontakte online social network, based on data obtained from his profiles. We have used various parameters obtained from users' personal pages in VKontakte social network (the number of friends, subscribers, interesting pages, etc.) as their features for determining the professional success, as well as networks (graphs) reflecting connections between users (followers/ friends). In this work we performed graph classification by using graph convolutional neural networks (with different types of convolution layers). The best accuracy of the graph convolutional neural network (0.88) was achieved by using the graph isomorphism network (GIN) layer. The results, obtained in this work, will serve for further studies of social success, based on metrics of personal profiles of OSN users and social graphs using neural network methods.

1. Introduction

In recent years, online social networks have played an important role in people's lives. They have become more and more popular in enabling people to connect, communicate, and share information. In Online social networks (OSN)s, the user's behaviour and interpersonal interaction are based on both communication mediated by virtual space, and various products of the user's virtual activity – user's personal profile metrics. The metrics of the OSN user's personal profile (for example, audio recordings, video content, avatars, friends, posts and reposts, likes, etc.) reflect real behaviour of a person by acting as characteristics of his virtual activity. Consequently, these metrics are predictors, which we can be used to predict person's behaviour in real life through the features of his behaviour in an OSN [1].

Usually, if any user registers in any online social network, he gets the ability to upload photos, share news, communicate with other users, make friends, and follow events taking place in the world. The popularization of online social networks has increased interest in using data obtained from OSNs in their study by scientists from different fields of science [2]. Modern methods, based on machine learning,

https://doi.org/10.26555/ijain.v8i3.859

deep neural networks are gaining popularity for problems connected with the analysis of information obtained from various OSNs [3].

In order to obtain more accurate results when analysing social networks, it is necessary to use not only the user's profile metrics, but also the connection structures of social networks (relations between users should be taken into account), i.e. social networks must be represented as graphs. A graph is a complex data structure consisting of vertices and edges. The data structured as graphs are widely used in natural and social sciences. For example, diagrams of metro stations, road maps, family trees, networks of protein-protein interactions, etc. can be represented in the form of graphs. Online social networks can also be represented as graphs, where the vertices are users of a social network, and edges are connections between users. As a unique non-Euclidean data structure for machine learning, graph analysis focuses on such tasks as node classification, link prediction, and clustering. Therefore, a large number of scientists are engaged in research relating to analysis of graphs by using machine learning. GNNs are the state-of-the-art models composed of nodes and edges (graph structured data). Graph neural networks are applied to analyse graph data structures by using machine learning, and use a neighbourhood aggregation strategy, where a node representation is iteratively updated by aggregating representations from its neighbours. Graph neural networks provide a powerful toolkit for embedding real-world graphs into low-dimensional spaces according to specific tasks. In particular, convolutional neural networks (CNNs) [4] [5] could be generalized to be applicable for graph data by extending the convolution operation onto graphs. The extension of CNNs to graph structures leads to graph convolution neural networks (GCNNs).

Graph neural network-based methods are currently gaining wide popularity in social network data analysis. For example, in [6] the authors used a user's local network (friends) as input for a graph neural network, and developed strategies to include both network structures and user information into graph convolutional neural networks and attention networks for predicting their social influence. A similar study was performed in [7]. In this study, a heterogeneous neural network based on social influence prediction model was developed. In particular, the authors built a heterogeneous social influence network model based on relationships between event nodes and user nodes. Then they selected a heterogeneous user subgraph, extracted the corresponding node features for each user, and predicted the probability of the users' behaviour by using a heterogeneous neural network.

OSNs allow to quickly find information about the latest events in the world, and therefore many people use them as sources of news. In [8] the authors devoted their research to developing a new graph-based convolutional neural network approach that can efficiently track foxgrapher posts (images and related texts that convey misleading information) in online social networks for detecting misinformation publications. The selection of news based on preferences of social network users is studied in [9]. In this research, insights about users were obtained by using an attention graph convolutional network for recommending news, based on social information taken from users' profiles under study.

Online social networks accumulate a lot of different kinds of information that can characterize users. Every user of a social network can upload photos, music, videos that he likes, share his opinion with other people, make likes and interact directly with other users. All this information can be used to determine different properties and behaviour of a person, his way of life, etc. For example, in [10] the authors tried to predict social statuses and users' roles in their study. To do this, they developed a graph neural network-based model that used local and global characteristics of each user and relationships between users as the model's input [10].

In the survey [11] we conducted a detailed review of the current literature on deep representation learning for social network analysis, and introduced techniques for embedding subgraphs. Furthermore, the author discussed basic models for learning node representation, including some extensions of the base models. In [12] the authors developed a novel setting methodological approach based on graph convolutional neural network for modelling the spread of the node influence in neighbourhoods based on the length of walks. They provided examples to demonstrate the ability of this model to aggregate feature information from nodes based on a parameter regulating the range of node influence.

In recent years, a growing interest has been observed in learning graph-level representations for graph classification [13]. In graph classification problems a dataset consists of multiple graphs, and a GNN-based model is trained to predict a specific property of the whole graph (one label for each individual graph). This problem is topical for various fields of research [14][15], and new methods for solving the graph classification problem are further developed. In paper [16], the authors offered a novel multi-task representation learning architecture coupled with the problem of supervised node classification for learning graph-level representations in an end-to-end manner. In [17] the authors developed a differentially private stochastic gradient descent (DP-SGD) method for deep learning on multi-graph datasets, and reported results on a variety of synthetic and public datasets for different graph classification problems. In [18] the authors offered a class of models, where a multiplication rule specifying the algebra itself was inferred from the data during training, and developed graph neural networks that leveraged properties of hypercomplex feature transformation. The authors tested the offered GNN model on several open datasets and showed that the models reached state-of-the-art performance while consuming a much lower memory footprint with 70% fewer parameters.

Social networks are used in various studies to obtain information about various personality characteristics. In [19], and in subsequent works of these authors, the conducted research is devoted to predicting the type of a personality on the basis of predictors obtained from profiles of Facebook social network users. Certain attempts are also made to use social networks to predict the academic success of students [20], or track interaction in Problem Based Learning (PBL) approach [21]. For example, using regression and correlation analysis, it was possible to classify students by academic performance with high accuracy (93.3%) [21]. Another study using Pearson's correlation coefficient showed that social networks had a negative impact on the academic performance of medical students [22]. Social networks analysis (SNA) is also used in recruitment to identify potentially successful future employees [23] [24] or to recommend suitable vacancies [25]. In our previous work, we built a model based on the use of feed-forward neural networks to predict professional success on the basis of data from user profiles in VKontakte social network [26].

User success prediction means predicting how successful a particular user of VKontakte social network will be as a professional. To accomplish this task, we obtained data from profiles of job seekers on HeadHunter (hh.ru) website, and then these same people were found on VKontakte social network. Based on information from HeadHunter site questionnaire, each user was assigned to the class of successful or unsuccessful people. In this work, we train neural network models to predict to which class each user of VKontakte social network belongs to (thus solving the problem of binary classification). The input parameters of the models are numerical characteristics that describe the user's profile in VKontakte, as well as relationships with other users in this social network.

The goal of this work is a detailed analysis of advantages of graph neural networks in comparison to classical feed-forward neural networks in terms of predicting a person's professional success by using the data obtained from his profile in a social network. An important feature of graph neural networks is that they are designed to work with data that is represented as a graph. Since a social network is a graph, we propose that this feature of graph neural networks should give a significant advantage over other types of neural networks.

This study is based on data obtained from VKontakte online social network (vk.com). This social network has a convenient API-VK interface that allows researchers to download data directly from VKontakte database. Based on data obtained from VKontakte, we had previously developed simple multilayer feed-forward neural models for predicting the professional success of users of this social network [26]. In the present work we developed a graph neural network-based model (by using a labelled dataset introduced in [26]), and by using these models we significantly improved the classification accuracy (from 0.77 to 0.88 for a 5-parameter dataset [26]). For each user and his local neighbourhood connectivity (friends and friends of friends) quantitative metrics were downloaded by using API-VK, and local connectivity graphs were built. The constructed graphs and quantitative metrics were used as input data for graph convolutional neural networks (with different convolution layers) in predicting the person’s professional success.

The results of this work will allow building a system for predicting the success of a person by using information from his profiles in social networks, which will allow employers to recruit staff more efficiently. From viewpoint of developing new methods for the analysis of social networks, the paper shows how effectively GNNs can be used to solve the problem of classifying user profiles in social networks, and how much more efficient GNNs are in solving this problem in comparison with classical perceptron neural networks.

2. Method

2.1. Data Labelling

To create a labelled dataset (classes of professionally successful and unsuccessful persons) we used job seeker profiles on HeadHunter website (hh.ru). Initially, a dataset containing 15 parameters was obtained from HeadHunter website (hh.ru) for each person: full name; age; a position which the applicant is looking for; requested salary; work experience; education; the last modified date of curriculum vitae (CV) or a new CV compilation date; position at the last place of work; the last place of work (organization name); date of employment at the last place of work; date of employment termination at the last place of work; applicant's nationality; field of professional activity; name of the locality or region in which the applicant is looking for work; languages spoken by the applicant. For data labelling (i.e., for referring a person to the class of professionally successful or unsuccessful people), we chose the only four most significant parameters that characterized the professional success of a person:

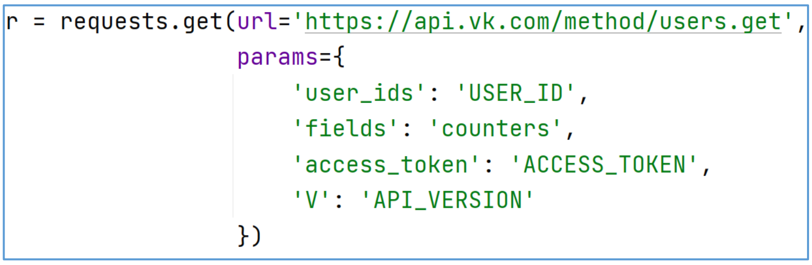
- Requested salary;
- Employment period at a previous place of work;
- Professional area;
- Job hunting region.

As values of different criteria for different regions and professional areas were different, it was necessary to scale the values of parameters according to these features. Therefore, we mapped salary values from monetary units to groups labelled from 1 to 5, where 1 was the label of job seekers with the lowest wage for a certain region and professional area, and 5 was the label of job seekers with the highest wage. All categories were ranked in the same way. Thus, a resulting dataset was divided into 5 groups, where the first group corresponded to the most unsuccessful, and the fifth – to the most successful people. In this work, we used only the first (lowest professional success) and fifth (highest professional success) groups [26] for further analysis.

2.2. Loading Data from VKontakte Online Social Network

We used the official VKontakte API to obtain the data from VKontakte social network (https://vk.com/dev/SDK). VKontakte API is a useful interface that allows to effectively download data from this social network database. To start downloading by using this API, one needs authorization by obtaining special access_token access key. VKontakte supports several ways of obtaining the access key. In this study the Implicit flow authorization method was used, which allowed us to call API methods directly from the user's device (for example, by using Javascript, Python scripts). The requests library of Python programming language was used for executing requests to VKontakte API.

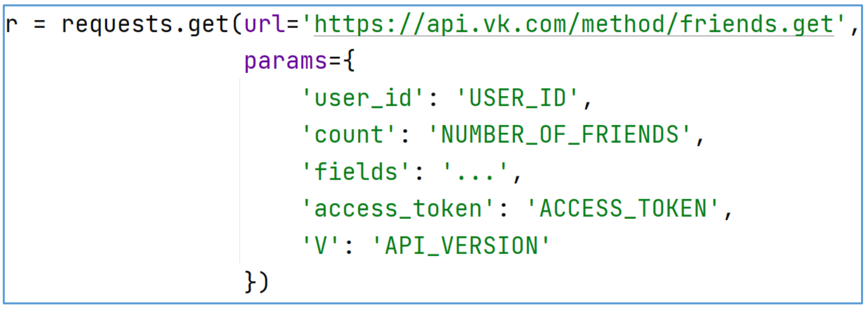
For each user from the labelled dataset (presented in the previous subsection) we downloaded the following data: number of friends, number of subscribers, number of interesting pages (communities), number of audio recordings, and number of video recordings from their profiles in VKontakte social network. To obtain this data users.get API method was used. An example script describing the process of calling this method by using the requests library is shown in Fig 1, where user_ids is the user ID in VKontakte OSN, fields are additional profile parameters to be obtained (in this case, the numeric values of the parameter specified – quantitative profile data), access_token is the user's access key, v is the version of VKontakte API used.



```
r = requests.get(url='https://api.vk.com/method/users.get',
                 params={
                     'user_ids': 'USER_ID',
                     'fields': 'counters',
                     'access_token': 'ACCESS_TOKEN',
                     'V': 'API_VERSION'
                 })
```

Fig. 1. An example of VKontakte API-based users.get request for obtaining information from the user profile with specific USER_ID.

To build a user graph describing local friendship connectivity, in addition to the numerical profile metrics, we downloaded the users' friends and friends of their friends. To get the user's friends, friends.get method of VKontakte API was used. As a result it returned a list of identifiers of the user's friends, and extended information about friends. An example of calling this method by using the requests library is shown in Fig 2, where user_id is the ID of the user to get friends for, count is the number of friends to be returned by this method.



```
r = requests.get(url='https://api.vk.com/method/friends.get',
                 params={
                     'user_id': 'USER_ID',
                     'count': 'NUMBER_OF_FRIENDS',
                     'fields': '...',
                     'access_token': 'ACCESS_TOKEN',
                     'V': 'API_VERSION'
                 })
```

Fig. 2. An example request for a list of friends' IDs, obtained for a specific user with the help of friends.get method of VKontakte API.

For local social friendship networks (friends and friends of friends) of the users considered in this study, quantitative metrics of their profiles were downloaded, as well.

Data from HeadHunter website (hh.ru) were used to classify specific people as professionally successful or unsuccessful, i.e., only for data labelling. Further, these people were found in VKontakte social network by last name, first name, and date of birth through VK API (https://vk.com/dev/SDK), and all the necessary information was obtained from the profiles of these people by using this API. These data were combined into a single dataset. Input features for neural networks were created by using the data obtained from https://vk.com/dev/SDK, whereas the target values (0-unsuccessful, 1- successful) were created based on HeadHunter data.

2.3. Graph Building

In this work, we modelled each user's local network as an undirected graph. Formally, a graph **G** is defined by a set of vertexes (nodes) **V** and set of edges **E** connecting pairs of nodes [27]. The set of nodes **V** corresponded to users of a social network. We used the terms "node" and "user" interchangeably in this article. The set of edges **E** corresponded to links between users in the online social network (users are "friends" in VKontakte OSN) is encoded into an adjacency matrix. The adjacency matrix of **G** is |**N**| × |**N** | binary square matrix **A**, where $\mathbf{A}_{\mathbf{ij}} \in \{\mathbf{0}, \mathbf{1}\}$ is **1**, if there is an arc connecting the **i**-th and j-th nodes, and **0** otherwise. In many practical applications the graph **G** is enriched with additional node and edge information. In our case, each node **N** was associated with a particular feature vector $\mathbf{X}_{\mathbf{i}}$, which contained quantitative information obtained from the user's profile by users.get API method, as described in the subsection above.

Based on the data describing connections between users graphs were built for each user from the initial dataset. For each user under study the authors constructed a graph, in which the vertices were analysed persons, their friends and friends of their friends, and the edges were the connections (friendship) between them.

NetworkX library was used to build graphs. NetworkX (https://networkx.org/) is a library for Python programming language designed to create, manage and study complex networks (graphs). An example of a graph built for 4 users from a labelled dataset is shown in Fig. 3.

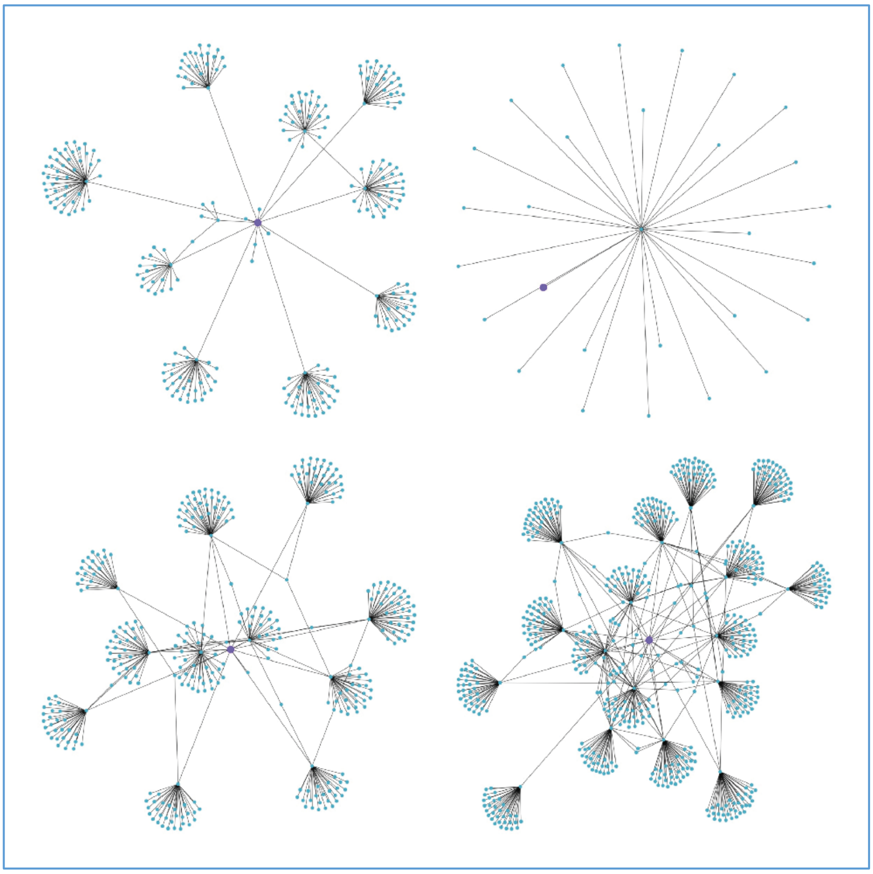


Fig. 3. Example graph visualizations of specific user's friends in VKontakte social network. The user under analysis is depicted in the center of the graphs.

2.4. Graph Neural Network Architecture

There have been several attempts made in the literature to extend neural networks to deal with arbitrarily structured graphs. In early works, researchers used recursive neural networks to process data represented in graph domains [28], [29]. GNNs were first introduced in [30] and [31] as a generalization of recursive neural networks that could directly deal with cyclic, directed and undirected graphs. Their framework consisted of an iterative process, which produced an output for each node and propagated the node states until equilibrium was reached. In [32] a convolution-like propagation rule was introduced for graphs along with the first methods for a graph-level classification based on this rule.

In the survey [33] the authors proposed a general design pipeline for graph neural network models and discussed variants of each component, systematically categorized the applications, and proposed open problems for future research. The work [34] is a good tutorial introduction to the field graph neural networks; it introduces the basic building blocks that can be combined to design novel and effective GNN models. This paper introduces a generalized formulation of graph representation learning based on a local iterative approach to structured information processing. In review paper [35] the authors provided a novel taxonomy for graph neural networks, and classified them into the corresponding categories. The authors also summarized future research directions so as to overcome the challenges faced.

To solve the graph classification problem, firstly, it is necessary to apply node aggregation techniques by using GNN layers. After having obtained a single graph representation, it is straightforward to make classification via feed-forward neural networks. We used 4 graph convolutional layers followed by a global mean pooling (GMP), because our task was to classify individual graphs with potentially different numbers of nodes. In this case, all graphs were represented with the fixed-size vector. Finally, the GMP features were fed to one linear layer with a binary cross-entropy loss function for adjusting model weights during training (see Fig 4). We used Python language-based machine learning framework PyTorch Geometrics (https://pytorch-geometric.readthedocs.io/) to create and train graph neural networks. It consists of various methods for deep learning on graphs with GPU support from a variety of published papers, and provides an effective toolkit to deal with different convolution layers. We used the following convolution layers in this study: GCNConv, SAGEConv, GraphConv, GATConv, TransformerConv, GINConv. A neural network model based on graph convolutions can therefore be built by stacking multiple convolutional layers, with each layer being followed by a point-wise non-linearity. In all models the neural network weights are trained by using gradient descent. By deploying a trainable neural network to aggregate information in local node neighbourhoods, GNNs are trained in an end-to-end fashion together with the parameters of classification algorithm (Fig 4).

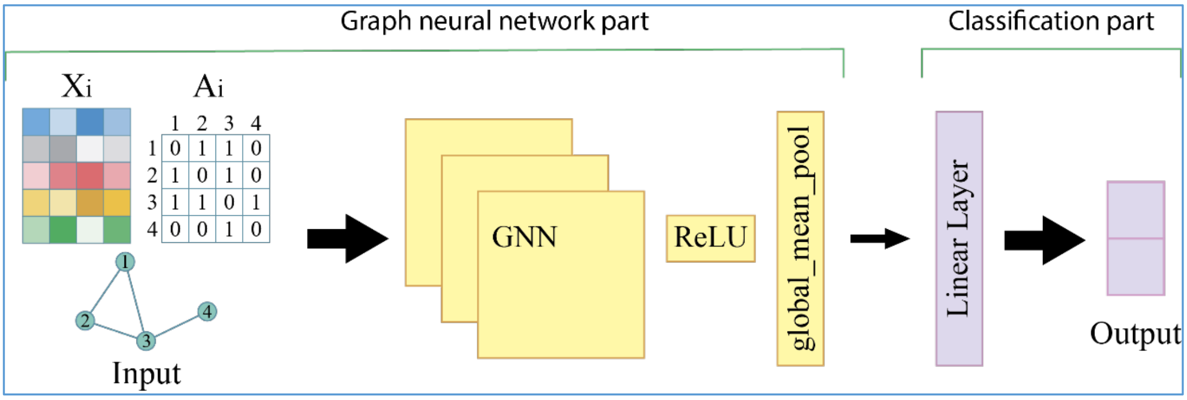
To solve the problem of a social network user classification, each user can be represented as an attributed graph **G_i = (X_i , A_i)**, where the node features **X_i** characterize numerical information, obtained from the user's profile, and the adjacency matrix **A_i** encodes the connection with other users (see Fig. 4). For a given dataset of labelled graphs **D = {G_i = (X_i , A_i), y_i}** with labels **y_i** indicates a professional success label of a specific user. The task is to train a classifier that maps each user to its corresponding label, **g : (X_i , A_i) → y_i.**

Next, we will consider the GNN layers used for building neural network models in more detail.

GCNConv layer-based GNN. GCNConv is a convolution layer introduced in [36]. This GNN model uses an efficient layer-wise propagation rule that is based on a first-order approximation of spectral convolutions on graphs according to the formula:

$$H^{l+1} = \sigma\left(D^{\frac{1}{2}} A' D^{\frac{1}{2}} H^l W^l\right) \tag{1}$$

where $A' = A + I$ is an adjacency matrix of an undirected graph G with self-loops added for each vertex, D is the vertex degree matrix, W^l is a trainable weighted matrix for the layer l , $\sigma(\dots)$ denotes an activation function, such as ReLU, $H^l \in R^{N \times D}$ is an activation matrix for the layer l . For the first layer $l = 0, H^0 = X$, where X is the graph G node's feature vector matrix.



	1	2	3	4
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1
4	0	0	1	0

Fig. 4. The overall structure of GNN-based classification model architecture.

Each user is represented as an attributed graph with node features vector **X_i**, and the adjacency matrix **A_i**. Firstly, an input graph is passed sequentially through multiple graph convolution layers. Then the vertex features are pooled with a global mean pooling (GMP) layer, and passed to traditional linear layer-based binary classification setting with two outputs.

SAGEConv layer-based GNN. SAGEConv is a general inductive framework that leverages node feature information to efficiently generate node embeddings in graphs (introduced in [37]). This layer learns a function that generates embeddings according to the following formula:

$$x'_i = W_1 x_i + W_2 * mean_{j \in N_i} x_j \tag{2}$$

where x_i is a feature vector of the i-th vertex, W_1 and W_2 are trainable weight matrices, N_i is a set of vertices adjacent to vertex i, $mean_{j \in N_i} x_j$ is a vector elementwise mean operator $\{x_i, \forall j \in N_i\}$.

In this approach, node features are included in the learning algorithm, and the topological structure of the neighbourhood of each node and distribution of node features in the neighbourhood are also simultaneously learned by the model. SAGEConv outperforms baselines for classification of the unseen nodes category in evolving information graphs [37].

GraphConv layer-based GNN. GraphConv convolution layer is based on work [38], which is a generalization of GNNs based on the k-WL. In this layer, the propagation of the representation to the next layer occurs according to the following formula:

$$x'_i = W_1 x_i + W_2 \sum_{j \in N(i)} e_{j.i} * x_j \tag{3}$$

where x_i is a feature vector of the i-th node, W_1 and W_2 are parameter matrices, $e_{j,i}$ denotes the weight of the edge between the source node j and the target node i, with default value equal to 1. This model is stronger than other models in distinguishing non-isomorphic (sub-)graphs. It is capable of distinguishing more graph properties, and outperforms the state-of-the-art neural architectures in large-scale molecule learning tasks [38].

GATConv layer-based GNN. GATConv is an attention-based architecture used to perform node classification of graph-structured data introduced in [39]. This layer computes hidden representations of each node in a graph by attending its neighbours using a self-attention strategy according to the formula:

$$x'_i = a_{i,i} W x_i + \sum_{j \in N(i)} a_{i,j} W x_j \tag{4}$$

where W is a matrix of weight coefficients, x_i is a feature vector of the i-th node, $N(i)$ is neighbourhood of the i-th node, $\alpha_{i,j}$ are attention coefficients determined by the formula:

$$a_{i,j} = \frac{\exp\left(LeakyReLU\left(a^T [W x_i \| W x_j]\right)\right)}{\sum k \in N(i) \cup \{i\} \exp\left(LeakyReLU(a^T [W x_i \| x_k])\right)} \tag{5}$$

where a is an attention mechanism that shows the importance of vertex j for vertex i, T is a transposition operation and $\|$ is a concatenation operation. The attention-based architecture performs the operation efficiently, since it is parallelizable, and can be applied to graph nodes having different degrees by specifying arbitrary weights to the neighbours.

TransformerConv layer-based GNN. TransformerConv is a novel Unified Message Passaging Model (UniMP) that can incorporate feature and label propagation, and adopts a Graph Transformer network, taking feature embedding and label embedding as input information for propagation presented in [40]. For the given layer, labels are propagated to the next layer according to the following rule:

$$x'_i = W_i x_i + \sum_{j \in N(i)} a_{i,j} W_2 x_j \tag{6}$$

where x_i is the i-th node feature vector, W_1, W_2 are weight coefficient matrices, $N(i)$ is the i-th node neighbourhood. Here $\alpha_{i,j}$ is a multi-head attention coefficient, calculated by the formula:

$$a_{i,j} = softmax\left(\frac{(W_3 x_i)^T (W_4 x_j)}{\sqrt{d}}\right) \tag{7}$$

where W_3, W_4 are trainable weight matrices, T is a transposition operation, $softmax(...)$ is a multivariate logistic function, d is each head's size. To train the network without overfitting in self-loop input label information, this model introduces a masked label prediction strategy, in which some percentage of input label information is masked at random, and then predicted.

GINConv layer-based GNN. GINConv is a Graph Isomorphism Network (GIN)-based simple neural architecture [34], that shows a discriminative power equal to the power of Weisfeiler-Lehman graph isomorphism test. Propagation of labels to the next layer for GINConv layer occurs according to the following rule:

$$x'_i = MLP((1+\in) * x_i + \sum_{j\in N(i)} x_j) \tag{8}$$

where ϵ is a trainable parameter or fixed scalar, x_i is the i-th node feature vector, $N(i)$ is the i-th node neighbourhood, MLP is multilayer perceptron. This model was empirically validated on a number of graph classification benchmarks, and it was demonstrated that this model also achieved state-of-the-art performance [41].

3. Results and Discussion

All models of graph neural networks with different convolution layers presented in the previous section were trained on a training set, and the analysis of their performance was carried out on a test set. The entire dataset containing 5446 data records was split as follows: 70% (3812 records) were used as the training set and 30% (1634 records) were used as the test set. To achieve statistical significance of the results, 10 iterations were performed for all training and classification tasks, and average values of the neural network accuracy were used for 10 iterations in further evaluation of the results. All input datasets were balanced (the same count of data in two categories). Thus, the representativeness of "Accuracy" metric was achieved, which allowed it to be used as the main performance metric of graph neural networks used in this study.

All graph neural network models with different convolution layers were trained on 50 epochs by using Adam optimizer with the learning rate = 0.00001, and with the batch size equal to 128. All neural network models were trained on a computer with 64 Gb RAM, Intel Core i9-10920 CPU and NVIDIA® Quadro RTX™ 6000 GPU. A plot of accuracy obtained during training of different models is demonstrated in Fig 5a. It should be noted that almost all models (excluding GAT layer-based model) trained quite quickly, and after the 10-th epoch the accuracy value was almost constant. The results of the average accuracy, maximal accuracy and standard deviation for 10 experiments, as well as one epoch average training times obtained by using different convolution layers are shown in Table 1.Numerical characteristics of neural network training results. One epoch training time and accuracy for the test dataset in neural networks, based on six different GNN layers, averaged for 10 experiments. The GinConv layer-based neural network with the best training result is highlighted in bold.

Convolutional Layer	Milliseconds per epoch	Test Data		
		Max Accuracy	*Average Accuracy*	*Standard Deviation*
GCNConv	1564	0.869	0.858	0.006593
SAGEConv	1421	0.88	0.863	0.009686
GraphConv	1464	0.869	0.86	0.007231
GATConv	1876	0.84	0.823	0.015703
TransformerConv	2248	0.853	0.837	0.010255
GINConv	**1386**	**0.8898**	**0.876**	**0.007034**

The analysis of the accuracy values of the test dataset with different types of convolutional layers showed that the model of a graph neural network with GINConv convolution layer gave the best results relative to other models (89%). We performed calculations with different numbers (1,2,3,4) of each layer, and got the maximum accuracy for GINConv with 3 layers. The data presented in this Table was obtained by using 3 layers for each type of layer, in each experiment. It should be noted, that a further increase in the number of layers did not lead to improvement in the neural network training performance. Here we have given the average training time for only one training epoch, since the total training time of the neural network depends on the number of epochs, and can be easily calculated by multiplying the training time of one epoch by the number of training epochs.

Moreover, a receiver operating characteristic curve (ROC-curve) was built for each model of a graph neural network (Fig. 5b). The maximal value of the area Under the ROC curve (AUC) was also obtained by using graph neural network with GINConv layer (0.93).

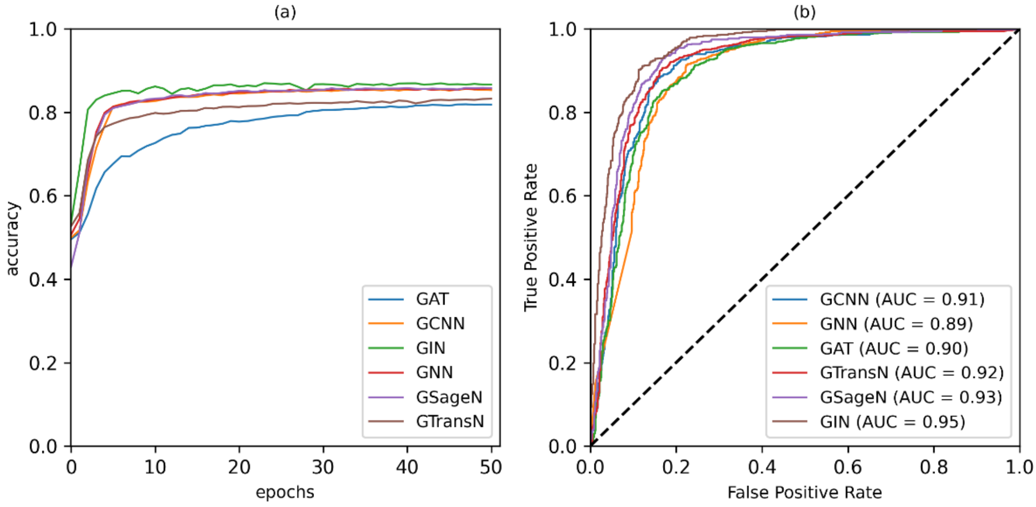


Fig. 5. Accuracy of different models in the test dataset during model training (a), and ROC-curves of the trained models (b)

A confusion matrix was calculated for the training set to visualize and summarize the performance of a classification algorithm by using graph neural network with GINConv layer (Table 2).

Table 1. Confusion matrix of the trained graph neural network with GINConv convolution layers

	Predicted Negative	Predicted Positive
Actual Negative	675	166
Actual Positive	36	757

By using the confusion matrix, in addition to the Accuracy metric, for a graph neural network with GINConv convolution layer metrics, we calculated Sensitivity, Specificity, Precision, and F1-score metrics to better characterize the performance of the trained neural network (Table 3).

Table 2. Performance metrics of the trained graph neural network with GINConv convolution layers

Metrics	Values
Accuracy	0.88
Mis-Classification	0.12
Sensitivity	0.95
Specificity	0.8
Precision	0.8
F1-score	0.87

We also carried out several more detailed numerical experiments to refine the features of the neural network with GINConv convolution layers. To determine the optimal number of layers in a model showing the best performance for a graph neural network with GINConv convolution layer, we trained this model with different numbers of convolution layers (1,2,3,4,5).

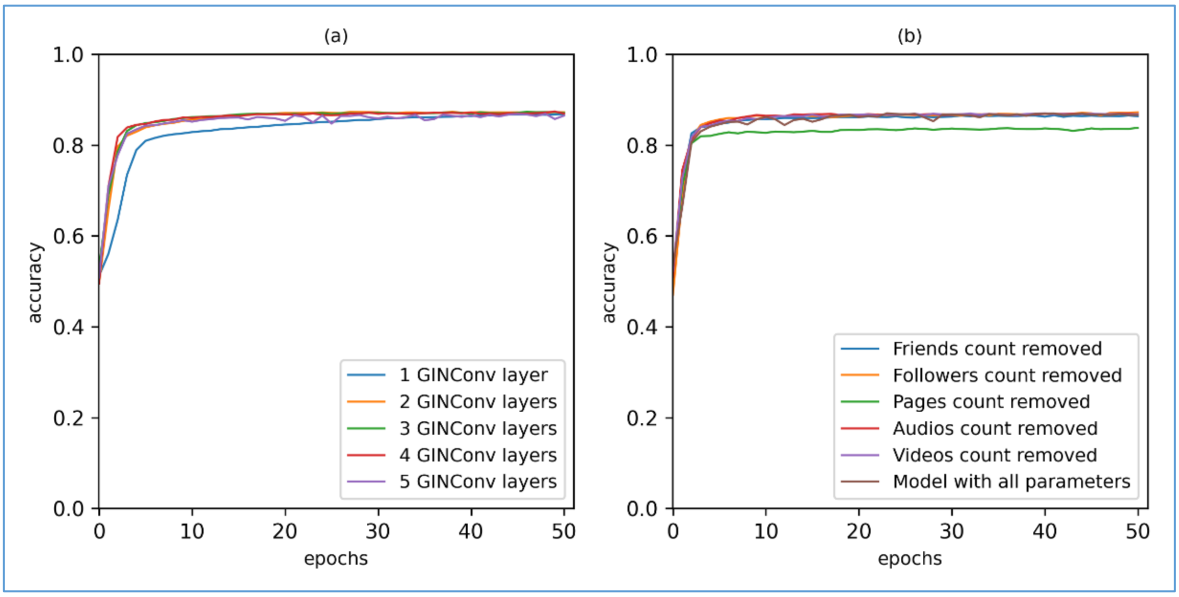


Fig. 6. Accuracy for neural network based on GINConv layer in different experimental setups (a- the influence of the number of GINConv layers on the training process, b-the influence of exclusion of specific input parameters on training accuracy).

In Fig. 6a we presented the model's accuracy for 50 epochs with different numbers of GINConv layers. It can be seen from these graphs that two GIN layers are enough in the neural network to achieve high accuracy values, and the further increase in the number of layers does not lead to the increase in the neural network quality.

To identify the importance of each parameter (friends, followers, pages, audios, videos), which were the components of the feature vectors of graph nodes, we performed neural network training with the excluded parameters. The input feature vectors were reduced to four parameters (each time one metric was removed from the vector). Fig. 6b shows graphs of the model accuracy for 50 epochs in these simulations.

GNN-based neural networks give higher classification accuracy (0.89) compared to the classical fully connected neural networks (0.77), which we used in the previous article [26]. Such an increase in the accuracy of the GNN-based classifier is explained by the fact, that in our classical neural network-based work, connections of the analysed user with other users of VKontakte social network (i.e. the place of analysed user in the social graph) were not taken into account. The GNN layers allowed us to organically include these very important characteristics into models, which made it possible to obtain such significant increase in the classification problem accuracy.

4. Conclusion

In this work we performed the research of the person's professional success on the basis of data from social networks by using graph neural networks. Various architectures of graph neural networks based on various types of layers were built, and their success was also analysed on the test dataset. These models were designed to solve the problem of classifying professionally successful and unsuccessful people based on the numerical characteristics of their personal pages and local connections (friends, friends of friends). Carried out a detailed analysis of the prediction performance based on various graph architectures with different numbers and types of layers (GCNConv, SAGEConv, GraphConv, GATConv, TransformerConv, GINConv). The highest prediction accuracy (88%) was obtained by using GINConv layer-based three-layer graph neural network. Graph neural networks have definitely shown a significant advantage over feed-forward networks. This is because GNNs are designed to work with data that can be represented as a graph, while a social network is a graph.

The study can serve as a basis for a neural network-based psychometric model for predicting the person's life activity through integration of "Social success " component [1]. The social success of a person is an ability to create broad contacts and communicate intensely, as well as an ability to influence other persons. The following indicators were defined as the metrics of social success: the number and quality of interpersonal relationships, degree and breadth of social influence, social significance of a person in a group, structure of interpersonal relationships, ability of a person to unite a group, etc.

A future focus area of this work is seen as the use of explainability methods for graph convolutional neural networks, because deep models are not amenable to interpretability [42]. Recently, the explainability of deep models has achieved significant results for texts and images. The explainability methods for graph neural networks are also experiencing rapid development, and several explanation methods are proposed to explain graph neural network models, including GNNExplainer, XGNN, etc.[43]. These methods will allow us to understand why the neural network gives a particular output, and will help us to highlight the basic graph structures, on the basis of which it classifies social network users to different categories of social success.

Acknowledgment

Funding: The study (all theoretical and empirical tasks of the research presented in this paper) was supported by a grant from the Russian Science Foundation (Project No. 19-18-00253, "Neural network psychometric model of cognitive-behavioural predictors of life activity of a person on the basis of social networks").

Declarations

Author contribution. The first and second authors created a computational model, implementing the ideas presented by the third author. The third author verified the simulation results from the psychological point of view. All the authors discussed the results and contributed to the final manuscript.
Funding statement. The study (all theoretical and empirical tasks of the research presented in this paper) was supported by a grant from the Russian Science Foundation (Project No. 19-18-00253, "Neural network psychometric model of cognitive-behavioural predictors of life activity of a person on the basis of social networks").
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

The datasets and program codes used in this study are available from the corresponding author on reasonable request.

References

[1] R. Buettner, "Predicting user behavior in electronic markets based on personality-mining in large online social networks," *Electron. Mark.*, vol. 27, no. 3, pp. 247–265, 2017, doi: 10.1007/s12525-016-0228-z.

[2] 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, pp. 1–11, Mar. 2022, doi: 10.26555/ijain.v8i1.800.

[3] A. Ulizulfa, R. Kusumaningrum, K. Khadijah, and R. Rismiyati, "Temperament detection based on Twitter data: classical machine learning versus deep learning," *Int. J. Adv. Intell. Informatics*, vol. 8, no. 1, pp. 45–57, Mar. 2022, doi: 10.26555/ijain.v8i1.692.

[4] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J. Big Data 2021 81*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[5] F. J. P. Montalbo and A. A. Hernandez, "Classifying Barako coffee leaf diseases using deep convolutional models," *Int. J. Adv. Intell. Informatics*, vol. 6, no. 2, pp. 197–209, Jul. 2020, doi: 10.26555/ijain.v6i2.495.

[6] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "DeepInf: Social influence prediction with deep learning," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 2110–2119, Jul. 2018, doi: 10.1145/3219819.3220077.

[7] L. Gao, H. Wang, Z. Zhang, H. Zhuang, and B. Zhou, "HetInf: Social Influence Prediction With Heterogeneous Graph Neural Network," *Front. Phys.*, vol. 9, p. 729, Jan. 2022, doi: 10.3389/FPHY.2021.787185.

[8] L. Shang, Y. Zhang, D. Zhang, and D. Wang, "FauxWard: a graph neural network approach to fauxtography detection using social media comments," *Soc. Netw. Anal. Min.*, vol. 10, no. 1, pp. 1–16, Dec. 2020, doi: 10.1007/S13278-020-00689-W.

[9] P. Zhu *et al.*, "SI-News: Integrating social information for news recommendation with attention-based graph convolutional network," *Neurocomputing*, vol. 494, pp. 33–42, Jul. 2022, doi: 10.1016/J.NEUCOM.2022.04.073.

[10] C. Zhang, S. Wang, D. Zhan, M. Yin, and F. Lou, "Inferring Users' Social Roles with a Multi-Level Graph Neural Network Model," *Entropy 2021, Vol. 23, Page 1453*, vol. 23, no. 11, p. 1453, Nov. 2021, doi: 10.3390/E23111453.

[11] Q. Tan, N. Liu, and X. Hu, "Deep Representation Learning for Social Network Analysis," *Front. Big Data*, vol. 2, p. 2, Apr. 2019, doi: 10.3389/FDATA.2019.00002.

[12] A. V. Mantzaris, D. Chiodini, and K. Ricketson, "Utilizing the simple graph convolutional neural network as a model for simulating influence spread in networks," *Comput. Soc. Networks*, vol. 8, no. 1, pp. 1–17, Dec. 2021, doi: 10.1186/S40649-021-00095-Y.

[13] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An End-to-End Deep Learning Architecture for Graph Classification," *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, pp. 4438–4445, Apr. 2018, doi: 10.1609/AAAI.V32I1.11782.

[14] W. Fan *et al.*, "Graph neural networks for social recommendation," *Web Conf. 2019 - Proc. World Wide Web Conf. WWW 2019*, pp. 417–426, May 2019, doi: 10.1145/3308558.3313488.

[15] X. Wei, R. Yu, and J. Sun, "View-GCN: View-based graph convolutional network for 3D shape analysis," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1847–1856, 2020, doi: 10.1109/CVPR42600.2020.00192.

[16] Y. Xie, M. Gong, Y. Gao, A. K. Qin, and X. Fan, "A Multi-Task Representation Learning Architecture for Enhanced Graph Classification," *Front. Neurosci.*, vol. 13, p. 1395, Jan. 2020, doi: 10.3389/FNINS.2019.01395.

[17] T. T. Mueller, J. C. Paetzold, C. Prabhakar, D. Usynin, D. Rueckert, and G. Kaissis, "Differentially Private Graph Classification with GNNs," Feb. 2022, doi: 10.48550/arxiv.2202.02575.

[18] T. Le, M. Bertolini, F. Noé, and D. A. Clevert, "Parameterized Hypercomplex Graph Neural Networks for Graph Classification," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12893 LNCS, pp. 204–216, 2021, doi:10.48550/arXiv.2103.16584.

[19] D. Quercia, R. Lambiotte, D. Stillwell, M. Kosinski, and J. Crowcroft, "The personality of popular facebook users," *Proc. ACM Conf. Comput. Support. Coop. Work. CSCW*, pp. 955–964, 2012, doi: 10.1145/2145204.2145346.

[20] M. Saqr, U. Fors, and J. Nouri, "Using social network analysis to understand online Problem-Based Learning and predict performance," *PLoS One*, vol. 13, no. 9, p. e0203590, Sep. 2018, doi: 10.1371/JOURNAL.PONE.0203590.

[21] M. Saqr and A. Alamro, "The role of social network analysis as a learning analytics tool in online problem based learning," *BMC Med. Educ.*, vol. 19, no. 1, pp. 1–11, May 2019, doi: 10.1186/S12909-019-1599-6.

[22] A. M. Bhandarkar, A. K. Pandey, R. Nayak, K. Pujary, and A. Kumar, "Impact of social media on the academic performance of undergraduate medical students," *Med. J. Armed Forces India*, vol. 77, pp. S37–S41, Feb. 2021, doi: 10.1016/J.MJAFI.2020.10.021.

[23] N. Melão and J. Reis, “Selecting talent using social networks: A mixed-methods study,” *Heliyon*, vol. 6, no. 4, p. e03723, Apr. 2020, doi: 10.1016/j.heliyon.2020.e03723.

[24] K. Toteva and E. Gourova, “Social network analysis in professional e-recruitment,” *Adv. Intell. Soft Comput.*, vol. 101, pp. 75–80, 2011, doi: 10.1007/978-3-642-23163-6_11.

[25] S. Chala and M. Fathi, “Job seeker to vacancy matching using social network analysis,” *Proc. IEEE Int. Conf. Ind. Technol.*, pp. 1250–1255, Apr. 2017, doi: 10.1109/ICIT.2017.7915542.

[26] F. M. Gafarov, K. S. Nikolaev, P. N. Ustin, A. A. Berdnikov, V. L. Zakharova, and S. A. Reznichenko, “A Complex Neural Network Model for Predicting a Personal Success based on their Activity in Social Networks,” *Eurasia J. Math. Sci. Technol. Educ.*, vol. 17, no. 10, p. em2010, Aug. 2021, doi: 10.29333/EJMSTE/11175.

[27] E. Kay, J. A. Bondy, and U. S. R. Murty, “Graph Theory with Applications,” *Oper. Res. Q.*, vol. 28, no. 1, p. 237, 1977, doi: 10.2307/3008805.

[28] P. Frasconi, M. Gori, and A. Sperduti, “A general framework for adaptive processing of data structures,” *IEEE Trans. Neural Networks*, vol. 9, no. 5, pp. 768–786, 1998, doi: 10.1109/72.712151.

[29] A. Sperduti and A. Starita, “Supervised neural networks for the classification of structures,” *IEEE Trans. Neural Networks*, vol. 8, no. 3, pp. 714–735, 1997, doi: 10.1109/72.572108.

[30] M. Gori, G. Monfardini, and F. Scarselli, “A new model for earning in raph domains,” *Proc. Int. Jt. Conf. Neural Networks*, vol. 2, pp. 729–734, 2005, doi: 10.1109/IJCNN.2005.1555942.

[31] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” *IEEE Trans. Neural Networks*, vol. 20, no. 1, pp. 61–80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.

[32] “Convolutional networks on graphs for learning molecular fingerprints | Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2.” (accessed Jan. 02, 2020) doi : 10.48550/arXiv.1509.09292.

[33] J. Zhou *et al.*, “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57–81, Jan. 2020, doi: 10.1016/J.AIOPEN.2021.01.001.

[34] D. Bacciu, F. Errica, A. Micheli, and M. Podda, “A gentle introduction to deep learning for graphs,” *Neural Networks*, vol. 129, pp. 203–221, Sep. 2020, doi: 10.1016/J.NEUNET.2020.06.006.

[35] Y. Zhou, H. Zheng, X. Huang, S. Hao, D. Li, and J. Zhao, “Graph Neural Networks: Taxonomy, Advances, and Trends,” *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 1, Jan. 2022, doi: 10.1145/3495161.

[36] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, Sep. 2016, doi : 10.48550/arXiv.1609.02907.

[37] N. K. Ahmed *et al.*, “Inductive Representation Learning in Large Attributed Graphs,” no. Nips, pp. 1–11, 2017, [Online]. doi : 10.48550/arXiv.1710.09471.

[38] C. Morris *et al.*, “Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks,” *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 01, pp. 4602–4609, Jul. 2019, doi: 10.1609/AAAI.V33I01.33014602.

[39] P. Veličković, A. Casanova, P. Liò, G. Cucurull, A. Romero, and Y. Bengio, “Graph Attention Networks,” *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, Oct. 2017, doi : 10.48550/arXiv.1710.10903.

[40] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, “Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification,” *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2, pp. 1548–1554, Aug. 2021, doi: 10.24963/IJCAI.2021/214.

[41] K. Xu, S. Jegelka, W. Hu, and J. Leskovec, “How Powerful are Graph Neural Networks?,” *7th Int. Conf. Learn. Represent. ICLR 2019*, Oct. 2018, doi : 10.48550/arXiv.1810.00826.

[42] P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, “Explainability methods for graph convolutional neural networks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 10764–10773, Jun. 2019, doi: 10.1109/CVPR.2019.01103.

[43] H. Yuan, H. Yu, S. Gui, and S. Ji, “Explainability in Graph Neural Networks: A Taxonomic Survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, Dec. 2020, doi : 10.48550/arXiv.2012.15445.