Proceedings of Machine Learning Research vol 245:1–8, 2024 2024 International Conference on Machine Learning and Intelligent Computing

Predicting Student Performance with Graph Structure Density-Based Graph Neural Networks

Xiaochen Lai* LAIXIAOCHEN@DLUT.EDU.CN
School of Software, Dalian University of Technology, Dalian, China

Wentao Hao WENTAO.HAO@FOXMAIL.COM
School of Software, Dalian University of Technology, Dalian, China

Zheng Zhang ZHANGZHENG@MAIL.DLUT.EDU.CN
School of Software, Dalian University of Technology, Dalian, China

Xiaohan Bai BAI_XH00@163.COM
School of Software, Dalian University of Technology, Dalian, China

Editors: Nianyin Zeng and Ram Bilas Pachori

Abstract

Educational data mining is a key area in data mining, focusing on predicting student performance. This research offers a new perspective for EDM, aiding educators in timely interventions based on performance predictions to reduce student failure and improve teaching quality. A novel approach: Graph Structure Density-based Sampling-Aggregation (GSDSA), is presented. GSDSA effectively reduces computational costs and improves training efficiency by limiting the number of neighbor nodes and using the Jaccard coefficient to measure node similarity. Moreover, it also considers relationships among students, employing the Pearson correlation coefficient to analyze none-graph-structured data, enhancing its processing capabilities. Experimental results demonstrate that GSDSA surpasses various graph neural networks and machine learning algorithms in classification accuracy.

Keywords: Graph neural networks, Educational data mining, Student performance prediction

1. Introduction

Educational Data Mining (EDM) represents a significant research direction within Data Mining (DM). EDM aims to analyze educational data through statistical and machine learning methods, modeling and analyzing students' multidimensional data and academic performance to enhance teaching quality, improve learning strategies, and ultimately better educational outcomes.

Predicting student performance is an important application dimension of EDM. Learning Management Systems (LMS), Massive Open Online Courses (MOOCs), and other web-based educational platforms provide data on student behavior. Analyzing these data can help identify students at risk of failing early, assisting educators in adjusting teaching strategies and developing new educational methods (Waheed et al., 2020).

Several algorithms have been used for predicting student academic performance, including Artificial Neural Networks (ANN), Support Vector Machines (SVM), Logistic Regression (LR) and Decision Trees (DT). Pallathadka et al. (2023) employed Naive Bayes (NB), SVM to classify and predict student performance. Waheed et al. (2020) employed ANN, SVM, and LR to predict whether students could pass based on demographic information and behavior data on online course platforms. Musso et al. (2020) proposed a machine learning model using ANN, focusing on learning

strategies, social support, motivation, socio-demographics, health status, and performance features to predict students' grades and dropout rates. Roslan and Chen (2023) employed DT, NB to predict students' performance in English and Mathematics based on past academic performance, demographics, and psychological attributes. Xu et al. (2019) utilized DT, Neural Networks, and SVM to forecast student performance based on internet usage behaviors, including timing, frequency, traffic, and duration. Bernacki et al. (2020) employed J-48 DT and J-Pip DT algorithms to predict student grades based on logs in Learning Management Systems for early warning purposes.

Existing EDM techniques primarily rely on individual students' demographic information, daily performance, and past grades for predictions, neglecting the correlations among students with similar profiles. Graph structures can represent both students' academic performance and their interconnections. This paper introduces a Graph Structure Density-based Sampling-Aggregation Graph Neural Network Model (GSDSA). During the sampling process, GSDSA employs the Jaccard coefficient to assess the similarity of students' academic statuses, prioritizing the sampling of student nodes with higher similarity, which reduces sampling variance and enhances classification accuracy.

2. Graph Neural Network

2.1. Graph Structure Data

Graph Neural Networks (GNN) are a type of neural network specialized in processing graph structure data. Graph structure data consists of nodes, edges, and their attributes.

Nodes can represent any entity, such as each node representing a student, with node attributes including characteristics of the student like demographic information, regular grades, past performance, etc.

Edges denote the relationships between nodes, which can be directed or undirected. The attributes of an edge can simply be set to indicate its existence or not, or can include weight, type, etc.

2.2. Propagation Rules and Applications of GNN

Graph neural network message propagation mainly includes two steps: information aggregation and feature update. During the information aggregation process, each node aggregates information from its neighboring nodes, using various methods for aggregation. Subsequently, nodes update their own features using the aggregated information. Assuming a node v at the t-th propagation state is $\mathrm{h}_{\mathrm{v}}^{\mathrm{t}}$, the set of neighboring nodes of v is N(v) , the set of edges connected to v is E(v) , f is the aggregation function, g is the output function, the message passing and update in a GNN can be represented as:

$$\mathrm{m}_{\mathrm{v}}^{\mathrm{t}+1} = \mathrm{f}(\mathrm{h}_{\mathrm{v}}^{\mathrm{t}}, \mathrm{N}(\mathrm{v}), \mathrm{E}(\mathrm{v})) \tag{1}$$

$$\mathrm{h}_{\mathrm{v}}^{\mathrm{t}+1} = \mathrm{g}(\mathrm{h}_{\mathrm{v}}^{\mathrm{t}}, \mathrm{m}_{\mathrm{v}}^{\mathrm{t}+1}) \tag{2}$$

The initial state of node v is represented by its features X(v), $\mathrm{h}_{\mathrm{v}}^{\mathrm{t}+1}$ represents the state of node v after t rounds of propagation. Each node aggregates information from each of its neighbors $\mathrm{m}_{\mathrm{v}}^{\mathrm{t}+1}$, combined with its own state $\mathrm{h}_{\mathrm{v}}^{\mathrm{t}}$ to update its state.

Graph Neural Networks can include multiple layers of information aggregation and feature update, each layer capable of capturing information from a broader neighborhood. This allows the model to consider relationships between more distant nodes. Graph Neural Networks compute directly on the graph structure, preserving the graph's structural information and enabling learning

from this structure. In the field of graph data, they effectively capture the relationships and interaction patterns between nodes, learning complex representations for nodes and edges. This strong generalization capability has led to their wide application in areas such as social network analysis, bioinformatics, recommendation systems, chemical and drug discovery, and knowledge graphs.

3. Research Methodology

3.1. GSDSA

During the sampling process, sampling all neighbors would lead to exponential increases in sampling costs as the number of network layers increases linearly. To reduce sampling complexity, various sampling methods are currently used in graph neural networks, such as random walk sampling, GraphSage (Hamilton et al., 2017), FastGCN (Chen et al., 2018), etc. The network structure proposed in this paper, as shown in Figure 1, samples a fixed number of neighbor nodes for each node, then aggregates these nodes to generate node embeddings, and finally makes predictions using an MLP (Multilayer Perceptron). R represents the classification result of the node.

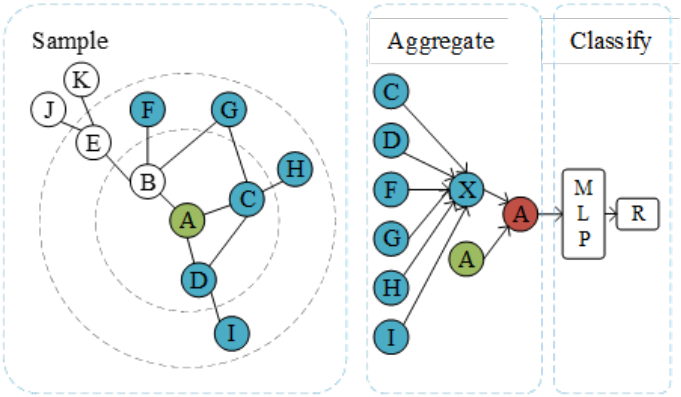

Figure 1: GSDSA frame diagram.

3.1.1. SAMPLING

Current sampling methods mainly focus on the features of the nodes themselves, neglecting the similarity relationships among nodes in the graph structure. This paper proposes a sampling method based on graph structure density. Given a hyperparameter k, which is the number of neighbor nodes sampled each time, the steps for sampling a neighborhood of a node v are as follows:

Find the complete set of first-order and second-order neighbors, $\mathrm{N}_{1,2}(\mathrm{v})$, of node v.

For all first-order and second-order neighbors $\mathrm{u} \in \mathrm{N}_{1,2}(\mathrm{v})$, calculate the structural density between them and node v using the Jaccard coefficient. The Jaccard coefficient is calculated using (3) where N(v) represents the set of first-order neighbors of node v.

$$\mathrm{Jaccard}(\mathrm{v},\mathrm{u}) = \frac{|\,\mathrm{N}(\mathrm{v}) \cap \mathrm{N}(\mathrm{u})\,|}{|\,\mathrm{N}(\mathrm{v}) \cup \mathrm{N}(\mathrm{u})\,|} \tag{3}$$

Clearly, if the Jaccard coefficient between v and u is greater than 0, then v is at most a two-hop neighbor of node n. Therefore, in the first step, only the first-order and second-order neighbors of the node are taken.

Nodes within $\mathrm{N}_{1,2}(\mathrm{v})$ are sorted by the magnitude of the Jaccard coefficient. If there are nodes with equal Jaccard coefficients, the features of the nodes are introduced, and the Pearson correlation coefficient is used to further distinguish them. The formula for calculating the Pearson coefficient between two nodes is as follows.

$$\mathrm{Pearson}(\mathrm{v},\mathrm{u}) = \frac{\sum(\mathrm{v_i}-\bar{\mathrm{v}})(\mathrm{u_i}-\bar{\mathrm{u}})}{\sqrt{\sum(\mathrm{v_i}-\bar{\mathrm{v}})^2\sum(\mathrm{u_i}-\bar{\mathrm{u}})^2}} \tag{4}$$

There $\mathrm{v_i}$ represent the ith feature of node v. Additionally, before computing the Pearson coefficient, the features of the nodes need to be normalized. From the sorted set $\mathrm{N}_{1,2}(\mathrm{v})$, the top k nodes are selected as the nodes for this sampling.

3.1.2. AGGREGATOR

Aggregating the sampled neighborhood is a crucial part of graph neural networks. This paper employs an average pooling aggregation function to aggregate the sampled nodes. Pooling aggregators can reduce data dimensions, thereby lowering computational requirements. Compared to max pooling, average pooling focuses on maintaining the representativeness of the overall feature while extracting sample features, preserving more feature information. After aggregation, the information is concatenated and updated with the node itself, followed by normalization.

$$\mathrm{h}^{\mathrm{t}}_{\mathrm{S(v)}} = \mathrm{mean}(\{\sigma(\mathrm{W_{pool}h}^{\mathrm{t-1}}_{\mathrm{u}} + \mathrm{b}), \forall \mathrm{u} \in \mathrm{S(v)}\}) \tag{5}$$

$$\mathrm{h}^{\mathrm{t}}_{\mathrm{v}} = \sigma(\mathrm{CONCAT}\left\{\mathrm{h}^{\mathrm{i-1}}_{\mathrm{v}}, \mathrm{h}^{\mathrm{i}}_{\mathrm{S(v)}}\right\}) \tag{6}$$

$$\mathrm{h}^{\mathrm{t}}_{\mathrm{v}} \leftarrow \mathrm{h}^{\mathrm{t}}_{\mathrm{v}} / \parallel \mathrm{h}^{\mathrm{t}}_{\mathrm{v}} \parallel^2 \tag{7}$$

3.2. Student Datasets Transformation into Graph

3.2.1. DATA PREPROCESSING

Before constructing graph-structured data, it is necessary to preprocess the student dataset. The dataset contains some missing values, necessitating the removal of incomplete data, such as information on students who dropped out midway. Textual types in the dataset are converted to Boolean types. Categorical features in the dataset are subjected to One-hot encoding to transform them into numerical variables and eliminate non-existent ordinal relationships, such as the information on students' places of birth. For numerical features, normalization is carried out individually by feature, keeping all data within the range of 0 and 1. Additionally, students are discretized into three categories based on their grades: excellent, passing, and failing, to serve as the outcome of the student classification.

3.2.2. QUANTIFY SIMILARITY

Student datasets are usually not in graph form, thus necessitating their conversion into graphs first. This paper constructs an undirected graph based on student features. Currently, graph structures in Graph Neural Networks are generally obtained by calculating the similarity between samples, which can be determined using various methods such as cosine similarity, Euclidean distance, Manhattan distance, Pearson correlation coefficient, etc. The Pearson correlation coefficient, capable

of identifying linear relationships between variables, offers intuitive understanding and ease of calculation. Therefore, this paper utilizes the Pearson correlation coefficient to calculate similarities between students. Specifically, each student in the graph is represented as a node. If two student nodes, v and u, have a Pearson correlation coefficient Pearson(v,u)>0.7, it is considered that these two students have a strong linear correlation.

The Pearson correlation coefficient can only identify linear relationships when calculating similarity. If two nodes, v and u, have features that indeed exhibit a linear relationship but have significantly different means, then the two nodes are not actually similar. This paper introduces a constraint condition that two nodes are considered similar only if they exhibit a strong linear relationship and satisfy $\frac{1}{c} \mid \sum u_i - \sum v_i \mid < 0.15$, at which point an undirected edge is added between the two nodes. Here, c represents the number of features of the nodes. Similarity is calculated pairwise for all students, and if the condition is met, an undirected edge is added, ultimately generating a graph-structured dataset.

Graph Neural Networks can include multiple layers of information aggregation and feature update, each layer capable of capturing information from a broader neighborhood. This allows the model to consider relationships between more distant nodes. Graph Neural Networks compute directly on the graph structure, preserving the graph's structural information and enabling learning from this structure. In the field of graph data, they effectively capture the relationships and interaction patterns between nodes, learning complex representations for nodes and edges. This strong generalization capability has led to their wide application in areas such as social network analysis, bioinformatics, recommendation systems, chemical and drug discovery, and knowledge graphs.

4. Experiments

4.1. Datasets

This paper validates experimental results using two types of datasets: graph datasets and student datasets, with three of each type. For graph datasets, Cora, CiteSeer, and PubMed are chosen; these are three common citation graph datasets widely used in graph neural network research. For student datasets, SAPD (Students' Academic Performance Dataset), DAP (Data of Academic Performance evolution for Engineering Students), and Oulad (Open University Learning Analytics Dataset) are selected. The main information about these two types of datasets is presented in Tables 1 and 2.

Table 1: Citation graph dataset.

Dataset	Nodes	Edges	Features	Classes
Cora	2708	5429	1433	7
CiteSeer	3327	4732	3703	6
PubMed	19717	44338	500	3

4.2. Comparison Method

This paper conducts comparative experiments using three graph neural network algorithms—GAT (Graph Attention Networks) (Veličković et al., 2017), GCN (Graph Convolutional Networks) (Kipf and Welling, 2016), GraphSage—and three traditional machine learning algorithms—MLP, SVM,

Table 2: Student dataset.

Dataset	Samples	Features	Classes
SAPD	480	16	3
DAP	12411	41	3
Oulad	4954	12	3

RF. Among these, the three graph datasets are experimented with GSDSA and the three mentioned graph neural network algorithms, while the three student datasets are experimented with the three traditional machine learning algorithms and the four graph neural network algorithms.

4.3. Experimental Setup

The preprocessed datasets are divided into three parts, with the paper designating the first 60% of the data as the training set, the middle 20% as the validation set, and the last 20% as the test set. Training employs the Adam optimizer, with an initial learning rate of 0.1, which decreases to a minimum of 0.001. The number of hidden layers is set to 1, batch size to 16, and the maximum number of training epochs to 500, utilizing an early stopping strategy.

The paper uses accuracy rate (ACC) to measure the final classification outcome.

4.4. Results and Analysis

The experimental results are shown in Tables 3 and 4.

Table 3: Citation graph dataset experiment ACC (%).

Method	Cora	CiteSeer	PubMed
GSDSA	**85.36**	**73.14**	**89.04**
GAT	**86.81**	69.21	85.47
GCN	84.41	65.4	**88.49**
GraphSage	86.29	**71.18**	87.76

Table 4: Student graph dataset experiment ACC (%).

Method	DAP	Oulad	SAPD
GSDSA	**89.41**	**80.61**	**85.26**
GAT	**89.71**	76.87	78.96
GCN	83.53	69.79	72.22
GraphSage	87.61	**79.48**	**81.38**
MLP	84.05	76.31	79.14
SVM	75.13	68.62	60.84
RF	80.48	75.02	77.46

From the experimental results, it is observable that in the experiments with graph datasets, GSDSA ranked second in performance on the Cora dataset and first on the other two datasets. Similarly, in the experiments with student datasets, GSDSA's performance was also notable, achieving first place in two datasets and second in another. Moreover, the experiments with student datasets revealed that algorithms based on graph neural networks generally surpassed the performance of the other three machine learning algorithms, showcasing the superior classification capabilities of graph neural networks on student datasets.

5. Conclusions

This paper introduces a Graph Structure Density-based Sampling-Aggregation Graph Neural Network Model (GSDSA), which calculates node relatedness using the Jaccard coefficient to enhance the utilization of graph structure information. It restricts the number of neighbor nodes sampled during the sampling process, and in each node's feature aggregation process, it only considers first-order and second-order neighbors, eliminating the need to consider the entire graph's information. This reduces computational costs and increases training efficiency. Comparative experiments have validated the model's superior classification performance on graph datasets. This paper applies the GSDSA model to student performance prediction. The current field of student performance prediction mainly considers the characteristics of the students themselves, neglecting the connections between students. After preprocessing and calculating the Pearson correlation coefficient, this paper constructs a graph-structured dataset from the student data, using the GSDSA model to predict student performance. This effectively utilizes the structural density information of graph data, showing desirable results in the experiments as well.

For future work, the sampling method could be further improved by considering both the graph structure and the nodes' own features, or by introducing an attention mechanism to further differentiate the importance of neighboring nodes and extract more key information from the graph.

References

Matthew L Bernacki, Michelle M Chavez, and P Merlin Uesbeck. Predicting achievement and providing support before stem majors begin to fail. *Computers & Education*, 158:103999, 2020.

Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: Fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Mariel F Musso, Carlos Felipe Rodríguez Hernández, and Eduardo C Cascallar. Predicting key educational outcomes in academic trajectories: a machine-learning approach. *Higher education*, 80(5):875–894, 2020.

Harikumar Pallathadka, Alex Wenda, Edwin Ramirez-Asís, Maximiliano Asís-López, Judith Flores-Albornoz, and Khongdet Phasinam. Classification and prediction of student performance data using various machine learning algorithms. *Materials today: Proceedings*, 80:3782–3785, 2023.

Muhammad Haziq Bin Roslan and Chwen Jen Chen. Predicting students' performance in english and mathematics using data mining techniques. *Education and Information Technologies*, 28(2): 1427–1453, 2023.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Hajra Waheed, Saeed-Ul Hassan, Naif Radi Aljohani, Julie Hardman, Salem Alelyani, and Raheel Nawaz. Predicting academic performance of students from vle big data using deep learning models. *Computers in Human behavior*, 104:106189, 2020.

Xing Xu, Jianzhong Wang, Hao Peng, and Ruilin Wu. Prediction of academic performance associated with internet usage behaviors using machine learning algorithms. *Computers in Human Behavior*, 98:166–173, 2019.