Graph Neural Network with Self-attention and Multi-task Learning for Credit Default Risk Prediction

Zihao Li[1], Xianzhi Wang[1], Lina Yao[2], Yakun Chen[1], Guandong Xu[1], and Ee-Peng Lim[3]

[1] University of Technology Sydney, Sydney NSW 2007, Australia
{zihao.li, yakun.chen}@student.uts.edu.au, {xianzhi.wang, guandong.xu}@uts.edu.au
[2] University of New South Wales, Sydney NSW 2052, Australia
lina.yao@unsw.edu.au
[3] Singapore Management University, Singapore 188065
eplim@smu.edu.sg

Abstract. We propose a graph neural network with self-attention and multi-task learning (SaM-GNN) to leverage the full advantages of deep learning for credit default risk prediction. Our approach incorporates two parallel tasks based on shared intermediate vectors for input vector reconstruction and credit default risk prediction, respectively. To better leverage supervised data, we use self-attention layers for feature representation of categorical and numeric data; we further link raw data into a graph and use a graph convolution module to aggregate similar information and cope with missing values during constructing intermediate vectors. Our method does not heavily rely on feature engineering work and the experiments show our approach outperforms several types of baseline methods; the intermediate vector obtained by our approach also helps improve the performance of ensemble learning methods.

Keywords: Credit default risk prediction · Graph neural network · Self-attention · Multi-task learning.

1 Introduction

Credit default risk refers to the possibility of a loss resulting from a borrower's failure to repay a loan or meet contractual obligations. It is a major concern for any bank and financial institution in making loan decisions. Bad loans can cause banks problems with their capital adequacy and, at worst, lead to default. Bad loans also risk impairing the long-term economic growth and lead to greater uncertainty and instability in the banking and financial systems. Therefore, it is highly necessary to assess borrowers' repayment abilities before authorizing a loan, which calls for accurate credit default risk prediction.

Traditional risk assessment highly relies on experts with professional knowledge and relevant experience to assess loan requests against specific business models and rules, which is labor-intensive and prone to personal bias. For example, the widely adopted '5C principle' [1] requires domain professionals to evaluate borrowers' default risk by manually evaluating borrowers on five aspects: *character*, *capital*, *capacity*, *collateral* and *conditions*. It helps to incorporate both qualitative or quantitative measures in the '5C principle'; however, this makes credit default risk assessment even more complicated and time-consuming.

The availability of big financial data related to personal and home credit offers the opportunity for automating credit default risk assessment with predictive models. Such models need to address several challenges for accurate credit default risk prediction. First, it requires incorporating all sources of clues about users' backgrounds, credit histories, investments, etc., to make accurate predictions. Second, there might be insufficient or incomplete information (e.g., non-existence of credit histories) about certain users, making it necessary to leverage the information about other users and past loans to improve the prediction accuracy for those users. Until recently, the related research has been focusing on traditional machine learning techniques, e.g., Support Vector Machine (SVM) [10, 23], decision tree [19], Random Forest (RF) [16], and XGBoost [14]. These models' performance highly depends on the quality of feature engineering, which requires high domain expertise to incorporate diverse sources and forms of data.

Recently, deep learning has shown great potential in addressing the above challenges, thanks to its capability to capture complex, non-linear relations from massive data. It has proven successful in various domains, such as computer vision natural language processing, speech recognition, and recommendation systems [13]. However, there have been limited studies on credit default risk prediction based on deep learning techniques. Moreover, the current studies cannot effectively leverage multiple aspects of clues (e.g., heterogeneous information in applicants' profiles, relations among loan applications) to overcome the challenges posed by incomplete profiles and missing values, which greatly impair the prediction accuracy [26]. To cover this knowledge gap, a loan application graph can be designed based on raw application records. Thus, the similar neighbors can be considered as the auxiliary information of the records contained missing values for credit default risk prediction. As shown in Figure 1, we could construct a loan application graph based on the similarity between each client's historical records first. Hence, the neighbors information can be introduced as the external information to alleviate the missing values problem. And the credit default risk of each record is able to be predicted via node classification techniques.

Overall, we propose a novel Graph neural network with Self-attention and Multi-task learning (SaM-GNN), which comprehensively incorporates self-attention, graph neural networks, and multitask learning for accurate credit default risk prediction. In a nutshell, we make the following contributions in this paper:

- We construct an undirected graph for loan applications and combine self-attention and graph convolution networks for representation learning in our model. The self-attention mechanism enables effective feature representation, and graph convolution networks allow for aggregating similar information via

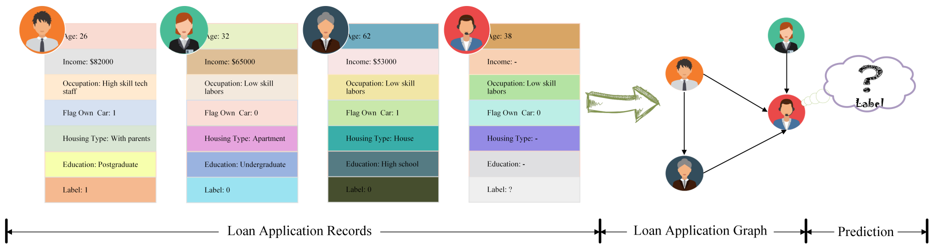


Fig. 1. Toy example. Each application record can be represented as the node in loan application graph. And the credit default risk prediction task can be transformed into the node classification task.

a graph structure to improve the model's robustness to missing values in the input.
- We design two parallel tasks for multi-task learning based on shared feature representation: a decoder module for input reconstruction and a classification module for credit default risk prediction. The two tasks are jointly trained to optimize feature representation and prediction results simultaneously.
- We conducted experiments on two real-world credit default risk prediction datasets. Our experimental results show a significant performance improvement of our approach over state-of-the-art methods. Besides, the feature representations of loan applications output by our approach helps improve the performance of existing ensemble methods.
- We will make our data and code public, including detailed parameter configurations for all the methods, to ensure reproducibility[4].

The rest of the paper is organized as follows. Section 2 overviews existing methods for credit default risk prediction. Section 3 introduces our self-attention graph neural network for credit default prediction and multi-task training strategy. Section 4 reports our experiments to evaluate our model on two real-world datasets, and finally, Section 5 concludes the paper.

2 Related Work

There has been many studies based on single-model machine learning methods like decision tree for credit default risk prediction in various contexts [3, 8, 19]. But until now, most existing machine learning techniques for credit default risk prediction are based on ensemble models, which take either bagging and boosting approaches. Bagging methods (e.g., Random Forest [4]) train multiple classifiers simultaneously and then combine them to make the final prediction [18]. In contrast, boosting methods (e.g., XGBoost [6], LightGBM [11]) apply individual models in a chain, where each model takes as input the result of the previous model [14, 15]. As a typical model of bagging strategy, Random Forest selects

[4] GitHub link has been anonymized for peer review and will be made public later.

samples and attributes randomly to construct and integrate decision trees. Uddin *et al.* [21] leverage Random Forest in micro-enterprises credit default risk modeling for accuracy and interpretability. Zhu *et al.* [25] conclude that Random Forest has better accuracy than other machine learning methods like logistic regression, decision trees, and SVM. Li *et al.* [15] use the XGBoost algorithm to identify customers who do not pay back from good customers. Li *et al.* [14] further design a stacking framework for XGBoost, SVM, and Random Forest for P2P default risk prediction and experimentally show that the model fusion algorithm has better adaptability and accuracy. Aleksandrova *et al.* [3] evaluate several popular machine learning algorithms for P2P credit scoring; they conclude that ensemble classifiers (e.g., XGBoost, GBM, and Random Forest) outperform non-ensemble models (e.g., logistic regression, decision tree, and multilayer perceptron). Differing from Aleksandrova's conclusion, Coser *et al.* [8] point out that logistic regression and Random Forest obtain the best results among various models for default risk prediction.

There has been limited work on deep learning for credit default risk analysis [9]. Wang *et al.* [22] use LSTM for P2P lending risk prediction, and Maria *et al.* [26] build connections between borrowers based on their geographic locations or economic activities and develop a multi-layer personalized PageRank model for credit default risk prediction. Since there is limited historical lending data, it is difficult to make an accurate prediction. Suryanto *et al.* [20] apply transfer learning to alleviate the issue of insufficient historical data. In summary, existing deep learning models rarely comprehensively use the full-spectrum multi-source, heterogeneous data for risk assessment. And none of them establishes effective approach to learning effective representations of loan applications or explicit connections among the applications. They commonly face challenges in dealing with incomplete profiles of applicants and missing information in loan applications. In this regard, we apply self-attention with multi-task learning for application record representation, and also consider graph neural network to capture the connection among similar applications and alleviate the missing values problem. To the best of our knowledge, we are the first to apply graph neural networks and multi-task learning simultaneously to improve the robustness of credit default risk prediction.

3 Proposed Method

Our proposed model (shown in Fig. 2) works as follows: it first generates embedding of categorical data (e.g., gender, suite type, education) and applies self-attention mechanism to the embedding and numeric data (e.g., income total and goods price) for feature representation; Then, the resulting representations are concatenated and updated via graph convolution constructed based on the similarity between loan applications to alleviate the impact of missing values. Finally, a decoder module and a classification module (consisting of multiple fully connected layers) are jointly trained to generate feature representation and simultaneously predict credit default risk.

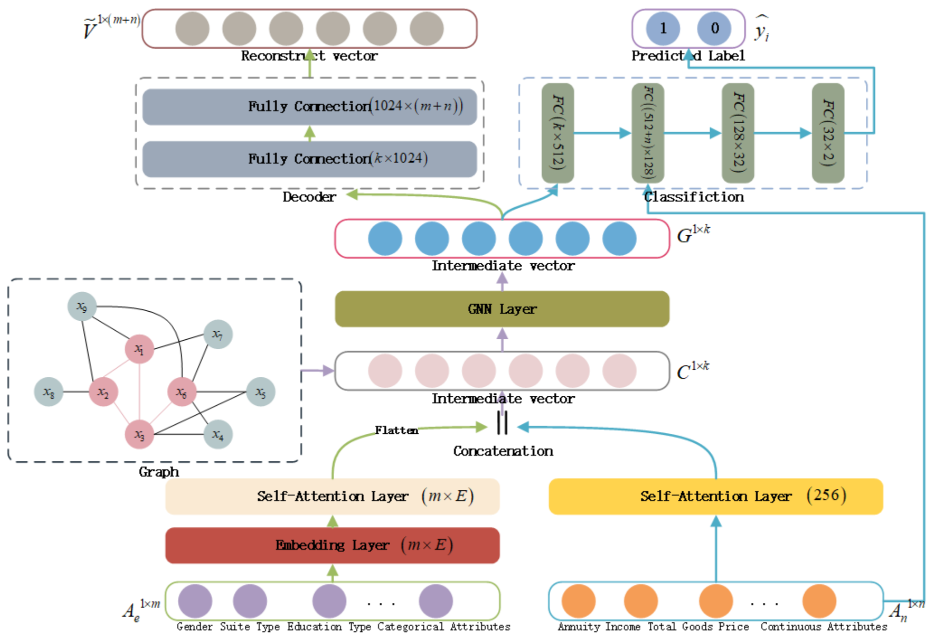


Fig. 2. The framework of SaM-GNN. $A_e^{1\times m}$. $A_n^{1\times n}$ are categorical attributes and numeric attributes, respectively, where m, n are the corresponding numbers. E is the embedding size, and $k = m \times E + 256$. C and G are intermediate vectors before and after similar information aggregation via graph convolution block. $\tilde{V}$ is the reconstructed input, and $\hat{y}$ is the predicted label.

3.1 Self-Attention Module

This module aims for generating feature representation for categorical and numeric inputs. Let $X_e = \{x_{e_0}, x_{e_1}, ..., x_{e_m}\}$ be categorical attributes and $X_n = \{x_{e_0}, x_{e_1}, ..., x_{e_n}\}$ be numeric attributes in the input. Suppose $\mathbf{E_i} \in \mathbb{R}^{1\times d}$ is the embedding of categorical attribute x_{e_i}, where d is the embedding size. The feature representations of categorical attributes are as follows:

$$\begin{cases} \mathbf{Q_e} & = \mathbf{E_e W_q} + \mathbf{b_q} \\ \mathbf{K_e} & = \mathbf{E_e W_k} + \mathbf{b_k} \\ \mathbf{V_e} & = \mathbf{E_e W_v} + \mathbf{b_v} \\ \text{Attention}(\mathbf{Q_e}, \mathbf{K_e}, \mathbf{V_e}) & = \text{softmax}(\frac{\mathbf{Q_e K_e^T}}{\sqrt{d_k}})\mathbf{V_e} \end{cases} \tag{1}$$

where $\mathbf{E_e} \in \mathbb{R}^{m\times d}$ is the embedding of categorical attributes. $\mathbf{W_q}, \mathbf{W_k}, \mathbf{W_v} \in \mathbb{R}^{d\times d}$ and $\mathbf{b_q}, \mathbf{b_k}, \mathbf{b_v} \in \mathbb{R}^{1\times d}$ are learning weight matrices. $1/\sqrt{d_k}$ is the scaling factor. Similarly, the feature representations of numeric attributes is as follows:

$$\begin{cases} \mathbf{Q_n} & = \mathbf{X_n W'_q} + \mathbf{b'_q} \\ \mathbf{K_n} & = \mathbf{X_n W'_k} + b'_k \\ \mathbf{V_n} & = \mathbf{X_n W'_v} + b'_v \\ \mathrm{Attention}(\mathbf{Q_n}, \mathbf{K_n}, \mathbf{V_n}) & = \mathrm{softmax}(\frac{\mathbf{Q_n^T} \cdot \mathbf{K_n}}{\sqrt{d_k}}) \mathbf{V_n}^T \end{cases} \tag{2}$$

where $\mathbf{X_n} \in \mathbb{R}^{1\times n}$ is the numerical attributes value after normalization. $\mathbf{W'_q}, \mathbf{W'_k}, \mathbf{W'_v} \in \mathbb{R}^{n\times n}$ and $\mathbf{b'_q}, \mathbf{b'_k}, \mathbf{b'_v} \in \mathbb{R}^{1\times n}$ are learning weight matrices. $1/\sqrt{d_k}$ is the scaling factor.

We concatenate the representations of categorical attributes and numeric attributes to construct the intermediate vector $\mathbf{C} \in \mathbb{R}^{1\times(m\times d+n)}$.

$$\begin{aligned} \mathbf{C} = \overset{m}{\underset{i=0}{||}} & \mathrm{Attention}(\mathbf{Q_{e_i}}, \mathbf{K_{e_i}}, \mathbf{V_{e_i}}) \\ & ||\mathrm{Attention}(\mathbf{Q_n}, \mathbf{K_n}, \mathbf{V_n}) \end{aligned} \tag{3}$$

where $||$ represents concatenation operation.

3.2 Graph Convolution Module

We construct an undirected graph by regarding each loan application as a node and adding weighted edges between nodes. Considering there are numerous attributes in raw data, we first select a subset of attributes to decrease the computational complexity. And then for every pair of nodes, we add an edge between them if they bear the same value on at least one attribute. We set the edge's weight to the number of attributes on which the two nodes share the same values.

To be specific, we define two thresholds, θ_m and θ_u, to select the attributes to participate in the graph construction. We choose an attribute if and only if 1) the proportion of loan applications that have missing values on the attribute in all the applications is smaller than θ_m and 2) the number of distinct values for the attribute is fewer than θ_u. These two thresholds should be appropriately configured to avoid it being excessively large (which introduces more noise) or small (which limits the auxiliary information usable to guide graph construction). We will study the impact of these thresholds in our experiments (Section 4.7).

Once the graph is constructed, we can apply graph convolution to aggregate neighbors' information and update the intermediate vector $\mathbf{C}$ [12]:

$$\mathbf{C}^{(l+\mathbf{1})} = \mathrm{ReLU}(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \mathbf{C}^{(l)} \mathbf{W}^{(l)}) \tag{4}$$

where $\mathbf{C}^{(l+\mathbf{1})}$ is the intermediate vector after $l+1$ layers of graph convolution; $A \in \mathbb{R}^{N\times N}$ is the adjacency matrix for graph $\mathcal{G}$, $\tilde{A} = A + I$, and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $\tilde{D}$ is the degree matrix of $\tilde{A}$; $\mathbf{W}$ is the learned weight. We denote the last layer of graph convolution as $\mathbf{C^{(L)}}$

Finally, we define a λ to determine how much information to aggregate from neighbors:

$$\mathbf{G} = \lambda \mathbf{C} + (1-\lambda)\mathbf{C^{(L)}} \tag{5}$$

where $\mathbf{G}$ is the final representation of the intermediate vector; $\mathbf{C}, \mathbf{C}^{(\mathbf{L})}$ are the intermediate vector before and after graph convolution; λ is a learned parameter that controls how much raw information to remember.

3.3 Decoder Module

The decoder module (shown in the top-left of Fig. 2) uses two fully connected layers to reconstruct the input and to capture a better representation of raw data.

$$\tilde{\mathbf{V}} = \mathrm{ReLU}(\mathbf{W_{d_2}}\mathrm{ReLU}(\mathbf{W_{d_1}}\mathbf{G^T} + \mathbf{d_1}^\mathbf{T}) + \mathbf{d_2}^\mathbf{T}) \tag{6}$$

where $\tilde{\mathbf{V}}$ is the reconstructed inputs, $\mathbf{W_{d_1}}, \mathbf{W_{d_2}}, \mathbf{d_1}, \mathbf{d_2}$ are the weight matrix for linear transformation.

3.4 Classification Module

The classification module uses fully connected layers to make predictions on whether or not to authorize a loan or not (the top-right of Fig. 2 shows the detailed specification of the module, e.g., the number of layers and the number of neurons of each layer). Although the intermediate vector $\mathbf{G}$ can capture semantic information from raw data (based on a stack of layers in our framework), it is prone to losing the shallow information in original data. To make up for the information loss, we concatenate the original numeric input (i.e., a numeric vector) and the intermediate vector as the combined input for predicting the risk probability. The prediction result can be represented as:

$$\hat{\mathbf{y}} = \sigma(\mathbf{MLP}(\mathbf{G}||\mathbf{A_n})) \tag{7}$$

where σ denotes the sigmoid activation function.

3.5 Joint Learning

We apply a joint learning strategy for input vector reconstruction and credit default risk prediction. The loss function includes mean square loss and cross-entropy loss for the two learning tasks, respectively:

$$\begin{aligned} L = \frac{1}{|N|}\sum_{i \in N}\big(&- \alpha(y_i log\hat{y}_i + (1-y_i)log(1-\hat{y}_i)) \\ &+ \beta\frac{1}{m+n}\sum_{j=1}^{m+n}(\mathbf{V_{ij}} - \tilde{\mathbf{V}}_\mathbf{ij})^2\big) + \lambda||\boldsymbol{\Theta}||_2 \end{aligned} \tag{8}$$

where $\alpha, \beta \in [0,1]$ balances the loss between the classification task and the reconstruction task. $y_i = 1$ indicates a positive case while $y_i = 0$ indicates otherwise. $\hat{y}_i$ is the predicted probability, i.e., the network's output after the softmax layer. $\mathbf{V_{ij}}, \tilde{\mathbf{V}}_\mathbf{ij}$ represent the original and reconstructed input vectors, respectively. $\boldsymbol{\Theta}$ denotes the set of trainable parameters. The last term is $L2$ regularization to mitigate overfitting.

4 Experiments

In this section, we report our experiments for evaluating our approach against several competitive baselines. Besides, we provide a further evaluation of our model under different configurations and parameter settings.

4.1 Datasets

We conduct experiments on two public datasets, which are representative of high-dimensionality and high-volume features of credit default risk data, respectively.

– **Home Credit Default Risk dataset**[5] covers various information about applicants, such as family information, income and expenditure, credit records, loan records, and repayment history. There are 307,511 train samples and 48,744 test samples, each having 477 numeric attributes and 55 categorical attributes in the raw data. The ratio of positive to negative samples in the train set is approximately 1:11. Values are missing for half of those attributes. We randomly draw 10% of the train data as the validation set.
– **Lending Club dataset**[6] contains millions of loan records with 151 attributes from 2007 to 2018. Following previous work [3], we remove meaningless attributes (e.g., URL, member_id), together with those attributes with more than 30% missing values, and then fill the remaining missing values with zeros. Each sample has 7 categorical attributes and 16 numeric attributes after preprocessing. The ratio of positive to negative samples is approximately 1:4. We use the most recent 10% records for testing and the rest for training.

4.2 Baseline Methods and Evaluation Metric

We select several recent competitive methods, which reflect the state-of-the-art research, to compare with our approach:

– **Logistic Regression [17]**: a simple and efficient linear model for binary classification.
– **Decision Tree [25]**: a technique that classifies samples following an ordering of attributes with a tree structure.
– **Random Forest [17]**: an ensemble learning method that trains a multitude of decision trees and determines the label via majority voting.
– **XGBoost [3]**: a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework.
– **Fully Connected Deep Network [3]**: a network with fully connected layers. We choose ReLU as the activation function and cross-entropy loss for optimization.

[5] https://www.kaggle.com/c/home-credit-default-risk/overview
[6] https://www.kaggle.com/wordsforthewise/lending-club

- **Convolution Neural Network [24]**: a network that feeds the concatenation of the representation of categorical data (obtained by convolution operations) and numeric attributes to fully connected layers for making predictions.
- **Wide & Deep Neural Network [7]** applies a linear model to improve the sparsity of categorical features and robustness of models via cross-product feature transformations. For the deep module, a feed-forward neural network is applied for feature representation, while the wide module is endeavored to improve the memory of the network.

For SaM-GNN, we set the embedding size $E = 5$, $\alpha = 0.5, \beta = 0.5$ for the loss function. For graph construction, we set $\theta_m = 0.3$ and $\theta_u = 20$ for attributes selection. The learning rate of the Adam optimizer is initialized to 0.001, which decays by 0.1 after every 50 epochs. Batch size and L2 penalty are set to 500 and 10^{-5}, respectively. For the classification module in SaM-GNN, we apply cross-entropy loss, use ReLU as the activation function, and set the dropout rate to 0.35 for all the fully connected layers except the last. The number of neurons in the layers are 512, 256, 128, 64, 32, 2 for Home Credit Default Risk dataset and 32, 16, 16, 2 for Lending Club dataset. More details about parameter configurations of methods can be found in our public code repository (see Section 1).

Following previous studies [2], we use *Area under the ROC Curve (AUC)* as the evaluation metric to evaluate models' performance. AUC has the characteristic of signifying the probability that positive samples receive higher scores than negative samples. Therefore, it can effectively reduce false alarms (or false positive rate) and decrease potential financial loss, making it especially suitable for the credit default risk prediction problem.

4.3 Comparisons with Baselines and Ablation Study

Table 1 shows the performance (with respect to AUC) of different methods. Generally, deep neural networks and ensemble methods outperform traditional models (Decision Tree, Logistic Regression), which has limited feature representation ability for high-dimensional data, while Boosting methods (XGBoost) outperform the Bagging method (Random Forest). Neural network-based models generally perform better than shallow models (traditional models, Bagging method) thanks to their strong feature representation and non-linear learning ability. Regarding Home Credit Default Risk dataset, SaM-GNN, outperform all the other methods. XGBoost and Random Forest achieve a remarkable improving after incorporating the intermediate vector generated by SaM-GNN, demonstrating the effectiveness of SaM-GNN in improving existing methods. As for Lending Club dataset, SaM-GNN and its variant without the Decoder module achieve a significant improvement (over 26%) over most of the other methods; they outperform the third-best method (i.e., XGBoost + Intermediate Vector) by a large margin of 0.1 in AUC, which reconfirms the superiority of our approach.

Table 1. Performance (AUC) of different models. The best three results are highlighted in boldface. ↑ and ↓ denote improvement and drop in performance, respectively. The numbers besides up/down arrows indicate the percentages by which the models improve their original versions.

Method	Home Credit	Lending Club
Logistic Regression	0.71739	0.69017
Decision Tree	0.72383	0.69925
Random Forest	0.74657	0.70380
XGBoost	0.76869	0.71616
Fully Connected Neural Network	0.76908	0.70448
Convolution Neural Network	0.77070	0.70961
Wide & Deep Neural Network	0.75824	0.70207
SaM-GNN	**0.78605**	**0.96982**
SaM-GNN w/o Decoder Module	0.77395 (↓ 1.54%)	0.96347 (↓ 0.65%)
SaM-GNN w/o Decoder & Graph Convolution	0.77026 (↓ 2.01%)	0.71253 (↓ 26.05%)
Random Forest + Intermediate Vector	0.75270 (↑ 0.82%)	0.86833 (↑ 23.38%)
XGBoost + Intermediate Vector	0.77289 (↑ 0.55%)	0.87326 (↑ 21.94%)

Our ablation study (based on comparisons between SaM-GNN and its variants, as shown in Table 1) demonstrates the effectiveness of considering the auxiliary information, i.e., similar credit application records, via graph convolution. While the Decoder module avails SaM-GNN's performance on Home Credit Default Risk dataset, it does not significantly impact the performance on Lending Club dataset—SaM-GNN obtains similar results regardless of whether it incorporates the Decoder module; this suggests the Decoder module is more effective on challenging tasks than easy ones—the classification task on Lending Club Dataset is not liable to overfit even without the Decoder module, given that the dataset contains millions of loan records but only 23 attributes.

4.4 Impact of Sampling Methods

Our datasets have imbalanced distributions over classes, with the ratios of positive samples to negative samples being around 1:11 and 1:4 for Home Credit Default Risk dataset and Lending Club dataset, respectively. Therefore, we test the effectiveness of three sampling strategies [3] in overcoming the class imbalance issue:

- **Upsampling**: Upsampling the positive samples and balance the data distribution. Let δ_{up} as the desired ratio of the number of samples in the minority class over the number of samples in the majority class after resampling.
- **Downsampling**: Downsampling the negative samples and balance the data distribution. Let δ_{down} as the ratio of the number of negative samples to the original number after resampling.
- **SMOTE [5]**: Selecting k nearest neighbors in the feature space for the minority class samples, drawing a line between the neighbors in the feature space, and drawing a new sample at a point along that line. Let δ_{smote} as

Table 2. Performance (AUC) of SaM-GNN under different sampling strategies. δ is the ratio of positive to negative samples after resampling. The best result under each ratio setting is highlighted in boldface.

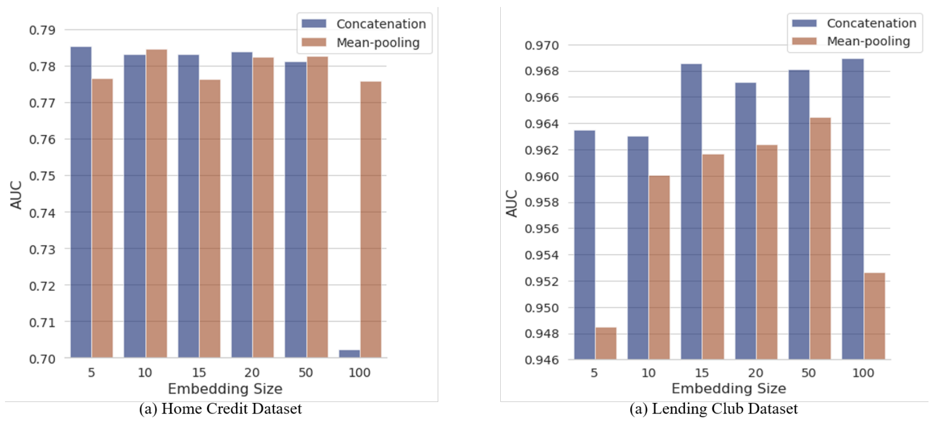
Dataset	Home Credit			Lending Club		
Ratio (δ)	0.5	0.75	1.0	0.5	0.75	1.0
Upsamping	**0.78665**	**0.78529**	0.76584	**0.96390**	**0.95482**	**0.95394**
Downsampling	0.77398	0.77425	**0.77825**	0.93928	0.91154	0.89367
SMOTE	0.76134	0.76130	0.76307	0.93250	0.95238	0.94642

the desired ratio of the number of samples in the minority class over the number of samples in the majority class after resampling.

Specifically, we study the performance of SaM-GNN under varying δ (the ratio of positive to negative samples after resampling). Our results (Table 2) suggest that among the three sampling methods, *upsampling* consistently results in the best AUC on both datasets. Also, for both datasets, the performance of sampling methods tends to fluctuate under varying values of the ratio (δ), indicating the best configurations of δ should be determined empirically rather than by following certain rules.

4.5 Impact of Vector-fusion Methods

Vector-fusion methods are for fusing the feature representations of categorical and numeric inputs to construct the intermediate vector. We study the impact of two commonly used vector-fusion methods, *concatenation* and *mean-pooling*, on the performance of SaM-GNN.



(a) Home Credit Dataset

(a) Lending Club Dataset

Fig. 3. Performance (AUC) of SaM-GNN with different vector-fusion methods on Home Credit Dataset and Lending Club Dataset.

Our results (Figure 3) show that *concatenation* (i.e., what we use in SaM-GNN) generally leads to better performance than *mean-pooling*—while both methods result in similar results on Home Credit Default Risk dataset, *concatenation* consistently outperforms *mean-pooling* on Lending Club dataset. *Mean-pooling* tends to favor larger embedding sizes, obtaining two out of the top-three best results under embedding-size=50 and 100 on both datasets; this is because larger embedding sizes help improve the network's feature representation ability when *mean-pooling* is applied. In comparison, the optimal embedding size for *concatenation* is more data-specific. More specifically, *concatenation* requires smaller embedding sizes (e.g., 5, 15, 20) for the smaller dataset (Home Credit Default Risk) yet larger embedding sizes (e.g., 20, 50, 100) for the larger dataset (Lending Club) to deliver the best results. This makes sense as Lending Club contains more training samples with lower-dimensionality, allowing for using a 'wider' neural network to improve performance without overfitting.

4.6 Impact of Multi-task Learning Parameters

We study the performance of our model (i.e., SaM-GNN without the graph convolution module) under varying parameters for multi-task learning α and β while keep the other hyperparameters the same as SaM-GNN, The results on Home Credit Default Risk dataset (Table 3) show the AUC values above the diagonal are generally greater than below, indicating it improves the performance to force the network to pay more attention to obtaining a better intermediate vector representation. We omit to show the results on Lending Club dataset, on which the impact of multi-task learning is less evident.

Table 3. Performance (AUC) of SaM-GNN (without considering the graph convolution module) under different parameter settings of multi-task learning on Home Credit Default Risk dataset. The best result in each row/column is highlighted in boldface.

α \ β	0.0	0.2	0.4	0.6	0.8	1.0
0.2	0.74548	0.76145	**0.77629**	0.76835	0.76600	0.75980
0.4	0.75684	0.76086	0.76550	0.76078	0.75654	**0.76702**
0.6	0.74604	0.75617	0.76626	**0.77077**	0.76352	0.76245
0.8	0.75136	0.75099	0.75422	0.76565	**0.76814**	0.76579
1.0	**0.76426**	**0.77107**	0.75140	0.76066	0.75738	**0.77991**

4.7 Impact of Graph Construction Parameters

We further study the impact of θ_m and θ_u, which affect attribute selection during graph construction and, in turn, determine the final graph structure. To this end, we test the performance of SaM-GNN (without the Decoder module) under $\theta_u \in \{20, 30, 50, 100, 200, 500, 1000\}$ for Home Credit Default Risk dataset, $\theta_u \in \{5, 10, 20, 30, 50\}$ for Lending Club dataset, and $\theta_m \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$. Intuitively, the graph contains more edges under smaller values of θ_u and θ_m.

Our results show our model favors denser graphs derived from the datasets—SaM-GNN achieves the best AUC (0.78002 and 0.96652) under the smallest θ_u values (20 and 5) for Home Credit Default Risk and Lending Club datasets, respectively. θ_m has a slighter impact on the results when compared with θ_u. We omit to show more details due to the limited space.

5 Conclusion

In this paper, we propose a self-attention graph neural network with multi-task learning for credit default risk prediction. The network features self-attention and graph convolution to represent heterogeneous data with missing values, along with multi-task learning of classification and decoder modules for model training. Extensive experiments on two real credit default risk prediction datasets demonstrate the superiority of our approach to existing models. Besides, feature representations from our approach help improve the performance of Bagging and Boosting methods.

References

1. Abrahams, C.R., Zhang, M.: Fair lending compliance: Intelligence and implications for credit risk management. John Wiley & Sons (2008)
2. Addo, P.M., Guegan, D., Hassani, B.: Credit risk analysis using machine and deep learning models. Risks **6**(2), 38 (2018)
3. Aleksandrova, Y.: Comparing performance of machine learning algorithms for default risk prediction in peer to peer lending. TEM Journal **10**(1), 133–143 (2021)
4. Breiman, L.: Random forests. Machine Learning archive **45**(1), 5–32 (2001)
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
6. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794 (2016)
7. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al.: Wide & deep learning for recommender systems. In: 1st workshop on deep learning for recommender systems. pp. 7–10 (2016)
8. Coser, A., Maer-Matei, M.M., Albu, C.: Predictive models for loan default risk assessment. Economic Computation and Economic Cybernetics Studies and Research **53**, 149–165 (2019)
9. Duan, J.: Financial system modeling using deep neural networks (dnns) for effective risk assessment and prediction. Journal of The Franklin Institute-engineering and Applied Mathematics **356**(8), 4716–4731 (2019)
10. Huang, C.L., Chen, M.C., Wang, C.J.: Credit scoring with a data mining approach based on support vector machines. Expert systems with applications **33**(4), 847–856 (2007)
11. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: a highly efficient gradient boosting decision tree. In: 31st International Conference on Neural Information Processing Systems. vol. 30, pp. 3149–3157 (2017)

12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
13. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)
14. Li, G., Shi, Y., Zhang, Z.: P2p default risk prediction based on xgboost, svm and rf fusion model. In: 1st International Conference on Business, Economics, Management Science. pp. 470–475 (2019)
15. Li, Y.: Credit risk prediction based on machine learning methods. In: 14th International Conference on Computer Science & Education. pp. 1011–1013. IEEE (2019)
16. Malekipirbazari, M., Aksakalli, V.: Risk assessment in social lending via random forests. Expert Systems with Applications **42**(10), 4621–4631 (2015)
17. Qiu, Z., Li, Y., Ni, P., Li, G.: Credit risk scoring analysis based on machine learning models. In: 6th International Conference on Information Science and Control Engineering (2019)
18. Schapire, R.E.: The strength of weak learnability. Machine Learning **5**(2), 197–227 (1990)
19. Sohn, S.Y., Kim, J.W.: Decision tree-based technology credit scoring for start-up firms: Korean case. Expert Systems with Applications **39**(4), 4007–4012 (2012)
20. Suryanto, H., Guan, C., Voumard, A., Beydoun, G.: Transfer learning in credit risk. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 483–498 (2019)
21. Uddin, M.S., Chi, G., Al Janabi, M.A., Habib, T.: Leveraging random forest in micro-enterprises credit risk modelling for accuracy and interpretability. International Journal of Finance & Economics (2020)
22. Wang, Y., Ni, X.S.: Risk prediction of peer-to-peer lending market by a lstm model with macroeconomic factor. In: ACM Southeast Conference. pp. 181–187 (2020)
23. Wang, Y., Wang, S., Lai, K.K.: A new fuzzy support vector machine to evaluate credit risk. IEEE Transactions on Fuzzy Systems **13**(6), 820–831 (2005)
24. Zhou, X., Zhang, W., Jiang, Y.: Personal credit default prediction model based on convolution neural network. Mathematical Problems in Engineering (2020)
25. Zhu, L., Qiu, D., Ergu, D., Ying, C., Liu, K.: A study on predicting loan default based on the random forest algorithm. Procedia Computer Science **162**, 503–513 (2019)
26. Óskarsdóttir, M., Bravo, C.: Multilayer network analysis for improved credit risk prediction. Omega-international Journal of Management Science **105**, 102520 (2021)