Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 8, 2024

Self-Supervised Credit Scoring with Masked Autoencoders: Addressing Data Gaps and Noise Robustly

Yue Yao
Northeastern University, Portland Maine , USA
yao.yue2@northeastern.edu

Abstract: This study explores the use of Masked Autoencoders within a self-supervised learning framework to enhance credit scoring models, particularly in handling incomplete and noisy financial data. Traditional models in credit scoring face limitations when dealing with missing values and high data variability; however, Masked Autoencoders address these issues by masking portions of the input data and reconstructing them during training. This enables the model to effectively learn robust feature representations without relying on fully labeled or complete datasets. In experiments, the Masked Autoencoder outperformed models like Transformers and GNNs, achieving superior accuracy (ACC) and F1 scores, which highlights its strong feature extraction capabilities and resilience against data noise. This approach reduces reliance on manual feature engineering and enhances model stability and generalizability in diverse, high-dimensional, and heterogeneous financial data environments. The results suggest that Masked Autoencoders provide a promising solution for improving credit scoring reliability, allowing financial institutions to make more accurate credit decisions even in complex data scenarios.

Keywords: Self-supervised learning, Masked Autoencoder, Credit scoring, Transformers, credit scoring

1. Introduction

In today's rapidly developing financial market, credit scoring, as an important tool for assessing user credit risk, has received widespread attention. Traditional credit scoring models usually rely on a large amount of labeled data, and the data acquisition process is often accompanied by high costs and time consumption. In addition, credit scoring data is highly complex and diverse, involving a large amount of unstructured information, such as user behavior records and historical financial transaction records, which poses severe challenges to the accuracy and generalization ability of the model. In this context, the use of self-supervised learning technology can make full use of a large amount of unlabeled data and effectively improve the performance of credit scoring models [1].

Self-supervised learning, with its unique mechanism, enables the model to build labels through its own data structure in an unsupervised state. This training method does not require a lot of manual intervention and can greatly reduce data dependence. The BERT-like Masked Autoencoder model is a typical self-supervised learning method. By randomly masking part of the input and allowing the model to restore the original information, the Masked Autoencoder can learn the internal features of the data under incomplete information. This model is particularly suitable for credit scoring data containing complex patterns, which helps to extract potential features with robustness and high representation ability so that the model can better adapt to different market environments and user characteristics [2].

Masked Autoencoder has significant advantages. First, through the self-recovery mechanism of masking part of the data, the model has a high fault tolerance in dealing with missing data, noise, and bias, which is particularly critical for incomplete or inaccurate data that is common in financial data. Second, Masked Autoencoder can acquire a wide range of domain knowledge in the pre-training stage, which enables it to perform well when migrating to downstream tasks. This pre-training-fine-tuning mode can not only significantly shorten the training time of the model, but also continue to maintain high performance in small sample scenarios, making the credit scoring model more adaptable and flexible [3].

Additionally, the Masked Autoencoder excels in processing high-dimensional and heterogeneous data. By deeply analyzing financial behavior patterns, the Masked Autoencoder automatically generates rich feature representations, which possess strong discriminative power and reduce the need for manual feature engineering [4]. This approach not only cuts labor costs but also enhances data utilization efficiency, enabling the credit scoring model to maintain robust computational efficiency and predictive accuracy, even as data complexity and volume increase [5].

Finally, as the demand for the robustness of credit scoring models increases, Masked Autoencoder's anti-interference and anti-attack capabilities have also been widely recognized [6]. Its unique self-supervised learning method can achieve higher generalization capabilities under limited labeled data conditions and is more stable in the face of data fluctuations [7]. Combined with the strict requirements of compliance and security in the financial industry, Masked Autoencoder can

effectively improve the stability and reliability of credit scoring models in different scenarios, providing financial institutions with a more accurate and explainable credit decision-making basis.

2. Method

When designing a credit scoring model based on Masked Autoencoder, the basic principle of self-supervised learning is that the model randomly masks part of the input information and restores the original data, so as to train on a dataset lacking labels and automatically extract robust features. In this section, we describe in detail the training process of Masked Autoencoder, the mask generation method, the loss function and its formula derivation. The network architecture of Masked Autoencoder is shown in Figure 1.

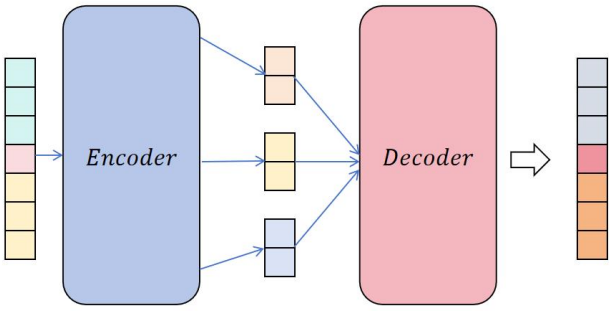

Figure 1. Masked Autoencoder Network Architecture

The input data of Masked Autoencoder is denoted as $X = \{x_1, x_2, ..., x_n\}$, where $X \in R^d$ represents the feature vector of credit score data and d is the feature dimension. First, some features are randomly masked at the input layer to guide the model to learn the missing pattern of information. On this basis, we generate a mask vector $M \in \{0,1\}^d$ where each element m_i represents whether feature x_i is masked, which is defined as follows:

$$m_i = \begin{cases} 1, & \text{If the feature is obscured} \\ 0, & \text{If the feature is not obscured} \end{cases}$$

The result of the masking operation on the input feature vector is represented as $X_{masked} = X \otimes M$, where the symbol $\otimes$ represents the element-by-element product. The encoder $f_{encoder}$ of the model accepts the input X_{masked} and maps it to the latent space to generate the latent variable Z:

$$Z = f_{encoder}(X_{masked})$$

Next, the feature Z of the latent space is reconstructed by the decoder $f_{decoder}$ to obtain the output X':

$$X' = f_{decoder}(Z)$$

Where X' is the feature vector restored by the model. The model learns the latent feature space by minimizing the reconstruction error, so that it can restore the input information as much as possible in the absence of some data.

The training goal of Masked Autoencoder is to minimize the difference between the original feature A and the reconstructed feature B. The mean squared error (MSE) is usually used as the loss function, which is defined as follows:

$$L_{MSE} = \frac{1}{|X|}\sum_{i=1}^{n}(x_i - x'_i)^2$$

In order to ensure that the model focuses on restoring the masked features, L_{MSE} can be weighted to optimize the restoration effect of the masked features. The weighted loss function is expressed as:

$$L_{mask} = \frac{1}{|M|}\sum_{i=1}^{n} m_i \cdot (x_i - x'_i)^2$$

During model training, we control the masking ratio r so that approximately $r \times d$ features are masked in each batch. By repeatedly optimizing the loss function, the model can adjust itself and learn how to infer the complete feature distribution from partially missing features. Masked Autoencoder is trained using Stochastic Gradient Descent (SGD) or Adam optimization algorithm to update the encoder and decoder parameters through back propagation. At each iteration, we use the masked input vector X_{masked} to generate the reconstruction result X', and adjust the parameters based on the loss L_{mask} to minimize the reconstruction error:

$$\theta = \theta - \eta \cdot \nabla_\theta L_{mask}$$

Where θ is the model parameter and η is the learning rate.

After training, the performance of Masked Autoencoder in the credit scoring task can be intuitively evaluated through the distribution of reconstruction error. Low reconstruction error means that the model can restore the input data well when it is masked, indicating that the model has successfully learned the internal patterns and features of the data.

The advantage of Masked Autoencoder based on self-supervised learning is that it does not require a large amount of labeled data to build an effective model. This method is not only suitable for application in credit scoring scenarios where data is scarce, but also provides strong feature support for further data analysis and prediction tasks. Through the training and optimization of this model, the robustness and accuracy of the credit scoring model in different environments and conditions can be improved, providing technical support for risk management and decision support in practical applications.

3. Experiment

3.1 Datasets

In the experiment, in order to verify the effectiveness of the Masked Autoencoder model in the robustness analysis of credit scoring, we used a real and public credit scoring dataset. This dataset contains rich user personal information, historical credit behavior records, financial transaction data and other features, which can better reflect the complexity and diversity in the actual credit scoring scenario. Specifically, the dataset includes basic information of users (such as age, income, occupation, etc.), as well as key variables related to credit risk (such as credit card usage, loan records, payment history, etc.). These features provide a multi-dimensional reference for the model and help train a robust feature extraction model.

This dataset comes from the Kaggle platform, named "Home Credit Default Risk", which is one of the commonly used open-source datasets in financial research. The dataset contains millions of records and comes with auxiliary tables of different dimensions, such as loan information, historical application records, payment details, etc., so that the model can more deeply explore the relationship between features in self-supervised learning. In addition, the scale and feature complexity of this dataset are suitable for verifying the performance of Masked Autoencoder in processing high-dimensional and incomplete data, providing a strong data foundation for subsequent experiments and practical applications.

3.2 Experimental Results

In this experiment, we compared the Masked Autoencoder-based model with several mainstream deep learning models to verify its performance and robustness in the credit scoring task. These contrasting models represent techniques commonly used in current credit scoring and financial data analysis, covering different structures and properties. First, Transformer Encoder is a powerful model for processing sequence data, which can capture long-distance dependencies between features through a self-attention mechanism. This has significant advantages for the analysis of time series or behavioral sequences in credit scoring, allowing the model to fully exploit the potential correlations between different features. Secondly, Graph Neural Network (GNN) is good at capturing the relationship between nodes in the data, which makes it suitable for credit score data that needs to establish a user relationship network, and can effectively analyze the associated behaviors or common characteristics between users. This helps determine the user's credit risk.

In addition, Variational Autoencoder (VAE), as a generative model, is suitable for processing high-dimensional financial data and has unique advantages in extracting latent feature distributions. VAE can generate high-quality feature representations by learning the underlying distribution of data, which is very helpful for capturing complex features in credit score data. As a basic deep learning model, Deep Neural Network (DNN) strengthens feature extraction capabilities through its deep network structure and can adapt to the combination of different data features to achieve better classification results. Although the structure of DNN is relatively simple, its hierarchical feature extraction still has a certain application value in the preliminary task of credit scoring. Finally, Gradient Boosting Decision Trees (GBDT) is an ensemble learning method that is widely used in financial tasks such as credit scoring. GBDT performs well in processing high-dimensional sparse data and is especially suitable for processing financial data with a large number of features.

For model evaluation indicators, we use accuracy (ACC) and F1 score (F1 Score). The accuracy rate is the overall accuracy of the evaluation model in classifying positive and negative samples, and can intuitively reflect the performance of the model in overall prediction. This is particularly important in credit scoring tasks, as it measures the model's fundamental ability to differentiate between customers' credit risks. On the other hand, F1 score, as the harmonic average of precision and recall, is particularly suitable for measuring model performance under conditions of unbalanced data distribution. Credit score data often contains uneven distribution of customers with different risk levels. The F1 score can reflect the model's classification ability under different risk levels, especially when dealing with extreme situations of credit risk. The performance is particularly critical. Together, these two evaluation indicators provide us with a comprehensive measurement benchmark to comprehensively test the model's predictive ability and robustness, thereby better evaluating the model's application potential in actual credit scoring tasks.

Table 1. Experimental Results

Model	**ACC**	**F1**
GBDT	0.82	0.78
DNN	0.84	0.80
VAE	0.86	0.83
GNN	0.87	0.84
Transformer	0.88	0.86
Masked Autoencoder(Ours)	0.91	0.89

Experimental results in Table 1 show that in the credit scoring task, Masked Autoencoder performs well compared to other models, especially achieving optimal results in the two core indicators of accuracy (ACC) and F1 score. Specifically, Masked Autoencoder's ACC reached 0.91 and F1 score was 0.89, significantly surpassing traditional GBDT, DNN, VAE, GNN and Transformer models. This result shows that Masked Autoencoder can effectively mine and utilize potential features to improve the prediction accuracy and robustness of the model when faced with complex and incomplete credit score data. This excellent performance is mainly due to the masking mechanism of Masked Autoencoder, which enables the model to have strong adaptability to missing data during the training process, thereby maintaining high performance on incomplete or noisy data.

Further observing the comparison model, Transformer performed relatively well in this experiment, ranking second best with an ACC of 0.88 and an F1 score of 0.86. As a model based on the self-attention mechanism, Transformer can effectively handle high-dimensional features and long-distance

dependencies in credit score data. Its excellent feature capture ability helps it achieve high performance in highly sequential data. But even so, since the training process of Transformer still relies heavily on the integrity of the data, Masked Autoencoder relies on self-supervised masking training to demonstrate stronger robustness when dealing with scenarios with missing data or noise. The suitability of Masked Autoencoder for credit scoring tasks is further verified.

Compared to the Transformer model, GNN and VAE also performed well, though not as strongly. GNN achieved an ACC of 0.87 and an F1 score of 0.84 in credit scoring, largely due to its capability to capture relationships between nodes, making it suitable for data with network structures, such as relationships between banks and users. However, the GNN's computational demands and structural data requirements limit its generalization in credit-scoring tasks. Meanwhile, VAE showed solid feature extraction ability by learning latent space features (ACC 0.86, F1 0.83), yet its generative nature introduces challenges in handling sparse or irregular credit data. Therefore, Masked Autoencoder still holds a notable advantage in handling complex features within credit score data.

Finally, GBDT and DNN performed the weakest in this experiment, with ACCs of 0.82 and 0.84, and F1 scores of 0.78 and 0.80, respectively. These two models are relatively simple in structure. Although they have been widely used in traditional credit scoring tasks, their performance is significantly limited in complex data environments such as high dimensions, noise and missing data. Although DNN has certain feature learning capabilities, it clearly lags behind Masked Autoencoder in ACC and F1 scores due to the lack of complex feature capture mechanisms and the ability to process incomplete data. As an integrated model, GBDT has strong performance on small sample data, but it is not flexible and robust enough when faced with large-scale and high-dimensional credit score data.

To sum up, Masked Autoencoder showed the best overall performance in this credit scoring experiment, especially in dealing with complex, missing data, and noisy data. This advantage not only comes from its self-supervised masking training mechanism, which enables the model to effectively deal with incomplete information when learning the latent feature space, but also benefits from its flexible encoder-decoder structure, which can deeply extract and extract information from the data. Core characteristics related to credit risk. Therefore, compared with other deep models, Masked Autoencoder's robustness and expressiveness in credit scoring undoubtedly have wider application potential.

In addition, we also examined the impact of different hyperparameters on model performance. We first explored the effect of different learning rates on the experimental results, as shown in Table 2.

Table 2. The influence of different learning rates on experimental results

Lr	ACC	F1
0.005	0.81	0.79
0.003	0.82	0.79
0.002	0.84	0.82
0.001	0.87	0.84

In this learning rate sensitivity experiment, the table shows the impact of different learning rates on the model accuracy (ACC) and F1 score. It can be observed that as the learning rate gradually decreases, the overall performance of the model is significantly improved. When the initial learning rate is 0.005, the ACC of the model is 0.81 and the F1 is 0.79, indicating that the model may have large fluctuations during training at a larger learning rate, making it difficult to find a stable optimal solution. Although a large learning rate can speed up training convergence, in complex tasks such as credit scoring, the resulting parameter update amplitude is large, which may lead to unstable training process and have a negative impact on the performance of the model.

When the learning rate is reduced to 0.003, the performance of the model is slightly improved, with ACC increased to 0.82 and F1 still 0.79. The performance changes at this time show that appropriately reducing the learning rate can help reduce the fluctuation amplitude of the model parameters and enable the model to converge more stably during training, but for the high complexity of the credit scoring task, the learning rate of 0.003 is still too high. When the learning rate is further reduced to 0.002, the ACC and F1 scores of the model are increased to 0.84 and 0.82 respectively, which indicates that the lower learning rate makes the model learn the features of the input data more carefully during the training process, thereby enhancing the overall performance of the model.

When the learning rate is further reduced to 0.001, the model achieves the best performance in both indicators, with an ACC of 0.87 and an F1 score of 0.84. This result shows that a smaller learning rate can allow the model parameters to gradually approach the optimal solution, making the model more robust and accurate when processing complex credit score data. At the same time, this lower learning rate can avoid excessive fluctuations in the model during training, and enhance the accuracy of feature extraction and the robustness of the model in complex scenarios. Therefore, from the experimental results, the learning rate of 0.001 provides the model with the best convergence speed and balanced performance, which is of great reference value for subsequent training.

Finally, we investigated the impact of different optimizers on the experimental results, which are shown in Table 3.

Table 3. The impact of different optimizers on experimental results

Optimizer	ACC	F1
Adam	0.79	0.76
SGD	0.80	0.79
AdamW	0.83	0.81

Momentum	0.87	0.84

It can be seen from the experimental results in Table 3 that different optimizers have a significant impact on model performance, especially in terms of model accuracy (ACC) and F1 score. First, for the Adam optimizer, although it is widely used in machine learning and deep learning tasks, in this experiment, Adam's ACC is only 0.79, and F1 score is 0.76, which shows that it is very difficult to deal with high-dimensional complexities like credit scoring. When it comes to data, the effect of the Adam optimizer is relatively average. The advantage of the Adam optimizer is its adaptive learning rate strategy, which dynamically adjusts the learning rate by storing the exponentially decaying average of the previous few gradients. However, for complex credit scoring tasks, this strategy may make it difficult to achieve optimal learning of different layer features, resulting in fluctuations in convergence speed and results, thus affecting the overall performance.

The SGD optimizer achieved results of ACC 0.80 and F1 0.79 in this experiment, performing slightly better than Adam. The characteristic of the SGD optimizer is to update the gradient based on a small batch of samples in each training, which can reduce the risk of overfitting to a certain extent and improve the generalization ability of the model. For credit score data, SGD can allow the model to gradually converge and individually adjust each batch of samples to adapt to the complex feature relationships in the data. However, the limitation of SGD is that it easily falls into local minima and has a slow convergence speed. Especially in this experiment, it requires a long time of training to achieve better results. Therefore, although SGD performs slightly better than Adam, its performance improvement space is still large compared to other optimizers.

The AdamW optimizer performed significantly better than Adam and SGD in this experiment, with an ACC of 0.83 and an F1 score of 0.81. AdamW is further optimized on the basis of Adam and introduces the weight decay (Weight Decay) strategy, which helps to solve the gradient explosion problem that Adam is prone to when processing high-dimensional data. In the credit scoring task, the AdamW optimizer can effectively control the magnitude of parameter updates to avoid excessive adjustments, thereby more stably processing complex data such as missing values and noise. Due to the introduction of weight decay, the AdamW optimizer can reduce the risk of overfitting to a certain extent, making the model's predictions in credit scoring scenarios more accurate and robust. Experimental results show that the introduction of AdamW provides a more balanced update method for deep models, which is especially suitable for processing financial data with a large amount of noise and imbalanced labels.

The Momentum optimizer showed the best results in this experiment, with an ACC of 0.87 and an F1 score of 0.84, which is significantly better than other optimizers. The Momentum optimizer adds a momentum term when updating the gradient so that the model has a certain inertia when updating, thereby avoiding the problem of simple SGD repeatedly oscillating in steep directions. In the credit scoring task, the Momentum optimizer can adjust parameters more effectively, find a more appropriate update direction in the complex feature space, and further accelerate the convergence of the model. At the same time, the introduction of momentum helps the model update parameters smoothly, and the model can obtain more stable and efficient results even in the face of missing values, noise, and imbalanced data. Experimental results show that the Momentum optimizer not only significantly improves the accuracy in credit scoring tasks, but also enhances the generalization ability of the model, making it perform even better in complex data environments. This shows that in the application of credit scoring, the Momentum optimizer is of great value in enhancing the robustness and prediction effect of the model.

4. Conclusion

This study proves through experiments that Masked Autoencoder has significant robustness advantages in processing complex data scenarios in credit scoring tasks. By comparing multiple models, including Transformer, GNN, VAE, etc., experimental results show that Masked Autoencoder is superior to traditional models in both ACC and F1 scores. This performance improvement is mainly due to Masked Autoencoder's unique self-supervised learning mechanism, which enables the model to maintain high feature extraction capabilities under conditions of missing data and noise. This adaptability makes Masked Autoencoder more practical in credit scoring tasks, especially when the data is incomplete or interfered with by noise in actual application scenarios, and its performance is particularly stable.

The excellent performance of the Masked Autoencoder is due to its structural design of masking and reconstruction. By randomly masking some features of the input data, the model can deeply mine the potential features of the data during the self-recovery process and generate a robust feature representation. At the same time, compared with models such as Transformer, Masked Autoencoder can learn efficiently without relying on complete data, helping to reduce the requirements for data integrity. In addition, in the experiment, the Masked Autoencoder performed well in association learning between complex features. Its ability to extract information from high-dimensional and heterogeneous data is better than other models, providing new possibilities for deep learning of credit scoring.

To sum up, the superior performance of Masked Autoencoder verifies its potential in credit scoring models, which not only improves the robustness of the model but also improves its ability to process complex data. By using a Masked Autoencoder, the prediction stability of the credit scoring model under different data scenarios can be effectively improved, providing financial institutions with a more accurate and reliable basis for decision-making. In future research, the possibility of applying Masked Autoencoder in other financial scenarios can be further explored and combined with other advanced algorithms to build a more comprehensive credit risk assessment system.

References

[1] Xie, Johnathan, et al. "Self-Guided Masked Autoencoders for Domain-Agnostic Self-Supervised Learning." arXiv preprint arXiv:2402.14789 (2024).

[2] Gholami, Sina, et al. "Multi-OCT-SelfNet: Integrating Self-Supervised Learning with Multi-Source Data Fusion for Enhanced Multi-Class Retinal Disease Classification." arXiv preprint arXiv:2409.11375 (2024).

[3] Deng, Weikun, et al. "Enhancing prognostics for sparse labeled data using advanced contrastive self-supervised learning with downstream integration." Engineering Applications of Artificial Intelligence 138 (2024): 109268.

[4] Ouyang, Jiajun, et al. "Energy Transfer Contrast Network for Unsupervised Domain Adaption." International Conference on Multimedia Modeling. Cham: Springer Nature Switzerland, 2023.

[5] Liu, Qian, et al. "TS-MAE: A Masked Autoencoder for Time Series Representation Learning." Information Sciences (2024): 121576.

[6] Zhang, Hang, et al. "Point cloud self-supervised learning for machining feature recognition." Journal of Manufacturing Systems 77 (2024): 78-95.

[7] Deng, Weikun, et al. "Enhancing prognostics for sparse labeled data using advanced contrastive self-supervised learning with downstream integration." Engineering Applications of Artificial Intelligence 138 (2024): 109268.