applied sciences

MDPI

Article

A Deep Learning Approach for Credit Scoring Using Feature Embedded Transformer

Chongren Wang [1,2,*] and Zhuoyi Xiao [1]

[1] School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan 250014, China
[2] Digital Economy Research Institute, Shandong University of Finance and Economics, Jinan 250014, China
* Correspondence: wangchongren@sdufe.edu.cn

Abstract: In this paper, we introduce a transformer into the field of credit scoring based on user online behavioral data and develop an end-to-end feature embedded transformer (FE-Transformer) credit scoring approach. The FE-Transformer neural network is composed of two parts: a wide part and a deep part. The deep part uses the transformer deep neural network. The output of the deep neural network and the feature data of the wide part are concentrated in a fusion layer. The experimental results show that the FE-Transformer deep learning model proposed in this paper outperforms the LR, XGBoost, LSTM, and AM-LSTM comparison methods in terms of area under the receiver operating characteristic curve (AUC) and the Kolmogorov–Smirnov (KS). This shows that the FE-Transformer deep learning model proposed in this paper can accurately predict user default risk.

Keywords: credit scoring; machine learning; deep learning; transformer

check for updates

Citation: Wang, C.; Xiao, Z. A Deep Learning Approach for Credit Scoring Using Feature Embedded Transformer. *Appl. Sci.* **2022**, *12*, 10995. https://doi.org/10.3390/app122110995

Academic Editor: Habib Hamam

Received: 29 September 2022
Accepted: 28 October 2022
Published: 30 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

1. Introduction

With the development of financial technology, big data and artificial intelligence technology have been paid increasingly more attention by financial enterprises. For financial enterprises, such as banks and P2P lending platforms, the most important risk is credit risk, that is, user default risk. Therefore, an increasing number of enterprises are trying to apply artificial intelligence technology, i.e., deep learning, to user credit risk assessment so as to reduce the loan default rate and to improve the ability of enterprises to resist risks [1,2]; this problem has attracted increasing attention.

Credit scoring is essentially a classification problem in machine learning. With the help of a credit risk assessment model, applicants can be divided into "good" customers and "bad" customers. Financial institutions can make loan approval decisions and risk pricing based on the credit scoring results.

With the development of financial technology, some loan businesses are carried out on online platforms, from basic websites to the current mobile application (APP), which has accumulated massive amounts of user online behavioral data, such as data on user registration behavior, user login behavior, user click behavior, and user authentication behavior. These online behavioral data have important mining value. In recent years, with the maturity of deep learning technology, it has become feasible to mine these data.

Based on the online behavioral data of users and the credit data of financial enterprises, this study proposed an end-to-end transformer credit scoring system, which can accurately predict users' default risk.

The main contributions of this study are as follows:

1. This paper introduces transformer into the field of credit scoring based on user online behavioral data, and the experimental results show that the transformer used in this study outperforms LSTM and traditional machine learning models.

2. We make use of credit feature data and user behavioral data and develop a novel end-to-end deep learning credit scoring framework. The framework is composed of two parts, a wide part and a deep part, and it can automatically learn from user behavioral data and feature data.

The structure of this study is as follows: Section 2 summarizes the literature relevant to this study, Section 3 introduces the relevant theories and the transformer method proposed in this study, Section 4 analyzes the experimental results, and Section 5 summarizes this study.

2. Related Work

At the early stage of the development of credit scoring methods, due to the lack of comprehensive historical data in financial institutions, credit scoring mainly depended on the personal experience of experts. Later, with an increase in credit data, many statistical models and credit scoring methods gradually emerged. Altman [3] built a Z-score credit scoring model based on multivariate discriminant analysis technology, and Parnes [4] verified the superiority of the Z-score credit scoring method through detailed comparative analysis experiments. Logistic regression models are the most representative of statistical models. They are widely used because of their high prediction accuracy, simple calculation, and strong interpretation ability [5].

At present, a large number of scholars are introducing machine learning methods into the field of credit scoring research [6,7]. The traditional machine learning methods in this research field can be divided into individual classifier methods and ensemble learning methods. Individual classifiers that have been studied and applied in credit scoring include decision trees (DTs) [8] and SVM [9]. In addition, some recent studies have also proposed some improved individual classifiers [10]. Munkhdalai et al. [11] proposed a credit scoring approach that combines linear (softmax regression) and non-linear (neural network) methods.

Ensemble learning improves model performance by building and combining base learners, which can be further divided into homogeneous ensemble learning and heterogeneous ensemble learning. Homogeneous ensemble learning methods only use one kind of base learner for ensemble, such as random forest (RF) [12] and extreme gradient boosting (XGBOOST) [13]. Heterogeneous ensemble learning combines several kinds of base learners to improve model performance. Wang et al. [2] proposed a two-stage credit scoring model. The first stage is credit scoring, and the second stage is profit scoring. They used stacked generalization (stacking) to build the model, and the base learner includes LR, DT, and SVM. However, the features of the experimental data in these studies were usually low-dimensional and designed by experts [14].

In recent years, deep learning has shown remarkable results in many application fields, such as text sentiment classification [15], image classification [16], and recommendation systems [17]. Similarly, many studies have applied deep learning to the field of credit scoring, and research has proven the abilities of deep learning algorithms, which can automatically learn features from data. Tomczak and Zi ę Ba [18] proposed a new RBM-like credit risk prediction approach and proved the advantages of this credit scoring method through experiments. Yu et al. [19] proposed a new multi-level deep belief network (DBN) credit risk prediction method based on limit learning machine (ELM), which improved the credit risk prediction performance of this approach. Zhang et al. [20] proposed a hybrid model that combines transformer networks with CatBoost decision trees, and their experimental data came from a bank, but they were low-dimensional feature data.

With the development of the Internet industry, people's lives are becoming increasingly Internet-based, resulting in a large amount of user online behavioral data. Considering the large volume, high dimension, and sequential characteristics of user online behavioral data, for these kinds of data, the learning ability of traditional machine learning algorithms is limited; therefore, researchers have begun to use deep learning methods to deeply mine user online behavioral data. Some researchers have attempted to apply deep learning

methods based on user online behavioral data to recommendation systems. Hidasi et al. [21] built a recommendation system using a recurrent neural network (RNN) based on users' online operation behavioral data. The experimental results show that this recommendation method is superior to existing methods. Lang and Rettenmeier [22] introduced a long short-term memory network (LSTM) to predict consumer behavior on e-commerce websites using user behavioral data, and the experimental results show that this approach has good prediction effects.

Similarly, some studies have attempted to apply deep learning methods based on user behavioral data to the field of credit scoring. Wang et al. [1] made use of borrowers' online operation behavioral data and proposed a consumer credit scoring method based on an attention mechanism LSTM. This method only uses user behavioral data, and the research results show that this approach has advantages over existing methods.

To sum up, credit scoring methods based on machine learning and deep learning are increasingly becoming a research hotspot. The research on deep learning methods based on user behavioral data is still relatively scarce, and there are still some research gaps in the research field of deep learning credit scoring models based on user online behavioral data. On the one hand, the LSTM model has long-term dependence and cannot be parallelized, and further research on deep learning algorithms is required. On the other hand, existing studies have only built deep learning credit scoring models based on user behavioral data, and they have not used feature data to build an end-to-end neural network model. Therefore, further research combining user behavioral data and feature data to build deep learning credit scoring models needs to be carried out.

3. Theory and Method

3.1. LSTM

LSTM, which was proposed by Hochreiter and Schmidhuber [23], is widely used to process sequence information, such as text classification [24] and machine translation [25], because it can alleviate long-term dependencies. LSTM can realize the remembering and forgetting of long-term historical states through different gate structures.

As shown in Figure 1, suppose x_t is the parameter information of the new incoming training process, and h_{t-1} is the staged result of the last iteration process. The input x_t, the memory state C_{t-1}, and the intermediate output h_{t-1} in the forget gate determine the forgetting part of the memory state. x_t in the input gate is changed by sigmoid and tan h functions, and then, it determines the reserved vector in the memory state. Finally, the effective information is output by the output gate control, and a performance model with better prediction can be obtained by iterating the error correction many times. However, LSTM can only calculate in sequence, which leads to two problems. On the one hand, the calculation of each time period depends on the calculation results of the previous time period, so the model cannot calculate in parallel. On the other hand, although the gate structure of LSTM alleviates the problem of long-term dependence, LSTM still cannot solve this problem.

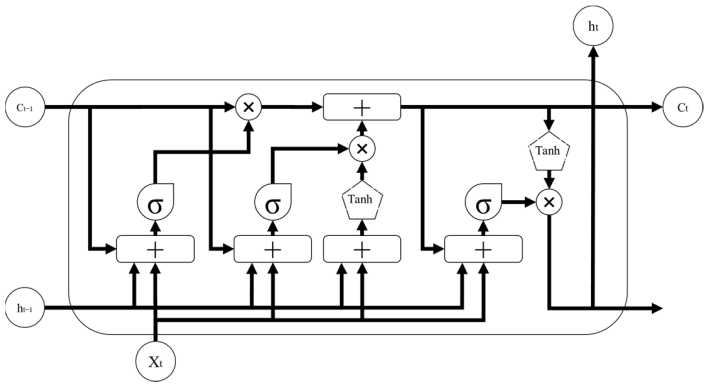

Figure 1. Structure of the LSTM model.

3.2. Transformer

The transformer model proposed by Google was first applied to the task of machine translation [26]. In this research, a transformer is an encoder–decoder structure. The transformer consists of an encoder and a decoder, which are stacked with 6 layers in total. This model does not use a recurrent structure. After passing through the 6-layer encoder in the model, the input data are output to the decoder of each layer in order to calculate the attention. The architecture of a transformer consists of four modules: an input module, an encoding module, a decoding module, and an output module.

A transformer is a deep neural network based on the self-attention mechanism and parallel data processing. It outperforms RNNs and convolutional neural networks (CNNs) in machine translation tasks, and it has become the current mainstream feature extractor. At the same time, the transformer solves two problems of LSTM. On the one hand, it uses an attention mechanism to reduce the distance between any two positions in a sequence to a constant. On the other hand, the transformer can be computed in parallel unlike the sequential structure of LSTM. The transformer is obviously superior to LSTM in terms of comprehensive feature extraction ability. Therefore, in the task of machine translation, the traditional attention-mechanism-based LSTM has migrated to the network structure based on the transformer model.

3.3. Feature Embedded Transformer

In this study, we introduce a transformer into the field of credit scoring and develop an end-to-end deep learning credit scoring framework; we named this framework the feature embedded transformer (FE-Transformer). The architecture of this method is shown in Figure 2. The FE-Transformer neural network is composed of two parts: a wide part and a deep part. The deep part uses the transformer neural network; the output of the transformer neural network and the feature data of the wide part are concentrated in the fusion layer; and finally, the prediction results are output. The FE-Transformer can automatically learn from user behavioral data and feature data.

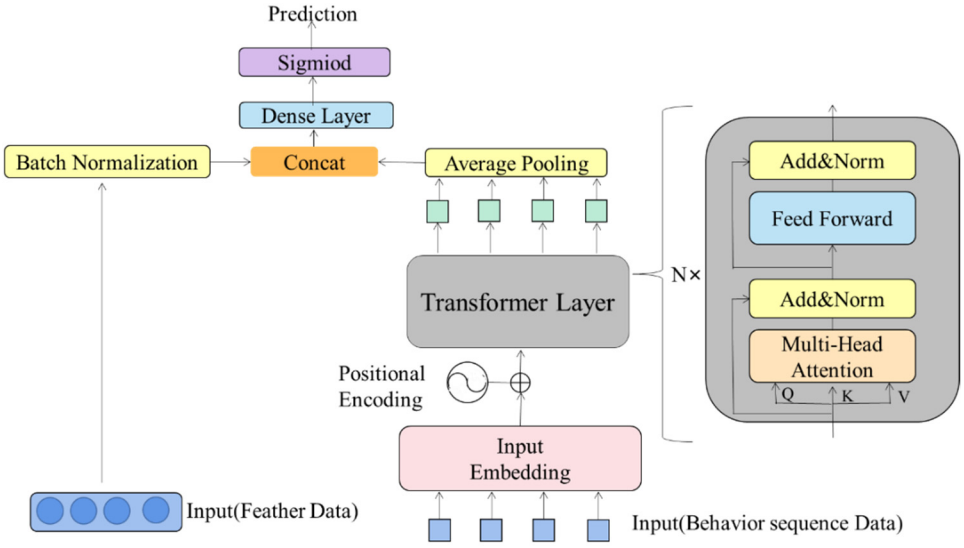

Figure 2. Network architecture of the FE-Transformer.

3.3.1. Input Data and Data Coding

There are two kinds of input data in this model: one is feature data, and the other is behavioral data. Feature data include users' gender, age, credit record, and other credit data. The users' behavioral data mainly include the users' online operation behavioral data, such as click behavior and input behavior. After the feature data are processed, they are used as the input of the model.

For the behavioral data, inspired by NLP, each kind of behavior event can be regarded as a word. The behavioral data of each user are composed of a series of events, which constitute a sequence of events and can be regarded as a sentence. We process the raw online operation behavior record data and convert these behaviors into event sequences in chronological order. Then, we encode the input behavioral data via embedding and position encoding.

An event is the basic unit of model processing. First, the input event needs to be converted into a vector through a word embedding algorithm. In order to understand a sequence of events, the model needs to know the position of the event in the sentence in addition to understanding the meaning of the event. Since the calculation of the transformer abandons the recursion and convolution of the cyclic structure, it cannot simulate the positional information of the events in the sequence, so it is necessary to obtain the positional vectors of the events through positional encoding. The position vector is then added to the event vector to obtain the input to the model. We take the sine function to generate the position vector for each event:

$$PE_{(pos,2i)} = \sin\left(pos/10000^{2i/d_{model}}\right) \quad (1)$$

$$PE_{(pos,2i+1)} = \cos\left(pos/10000^{2i/d_{model}}\right) \quad (2)$$

where pos is the position of the event in the behavior sequence of events, d_{model} is the dimension of positional encoding, $2i$ is the even dimension, and $2i+1$ is the odd dimension ($2i \leq d_{model}, 2i+1 \leq d_{model}$). After data coding, the data are used as the input of the transformer layer.

3.3.2. Transformer Encoding Layer

The transformer encoding layer is composed of one or more layers of stacked encoders. Each layer of the encoder is mainly composed of a multi-head attention layer and a fully connected feed-forward layer. Layer normalization [27] is used in front of each sublayer, and residual connection is used behind each sublayer. The transformer encoding layer structure is shown in Figure 2.

The event vector matrix obtained by the embedding layer is passed into the encoder through the multi-head attention layer and into the fully connected feed-forward layer, and then, the output is passed up to the next encoder. After one or more encoders, the encoding information matrix of all events in the behavior sequence is obtained.

The self-attention mechanism is an improvement in the attention mechanism, and it has the advantages of reducing the network's dependence on external information and being good at capturing internal correlations in data. The transformer architecture introduces a self-attention mechanism, which avoids the use of recursive structures in neural networks and completely relies on the self-attention mechanism to draw the global dependencies between the input and output [28].

The attention layer uses scaled dot-product attention. Compared with general attention, scaled dot-product attention uses the dot product for similarity calculation, which has the advantages of a faster calculation speed and being more space-saving. The basic structure is shown in Figure 3.

The self-attention mechanism is used to calculate the degree of relatedness between events. When calculating, each event in the input is first linearly projected into three different spaces to obtain a query vector (Q), a key vector (K), and a value vector (V). When obtaining self-attention information, the Q vector is used to query all candidate positions. Each candidate position has a pair of K and V vectors. The query process is the processing of dot products between the Q vector and the K vector of all candidate positions [29]. The product result is divided by the scaling factor (the square root of the dimension of the key vector) to improve the convergence speed. The result is normalized using the softmax

function and then weighted to the respective V vector, and the summation determines the final self-attention result. The calculation formula is shown in Formula (3):

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (3)$$

d_k is the number of columns of matrices Q and K, that is, the vector dimension.

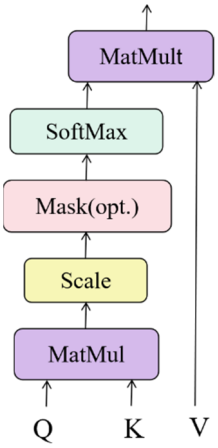

Figure 3. Scaled dot-product attention.

Multi-head self-attention enables the model to jointly learn the representation information of different locations from different representation subspaces. It is equivalent to a collection of different self-attention heads. As shown in Figure 4, after Q, K, and V are subjected to different linear projections, the scaled dot-product attention calculation is performed so that different parts of the input can be paid attention to and different semantic information can be learned. After multiple operations in parallel, the attention information in all subspaces is finally merged. The calculation process of each multi-head module shown in Equations (4) and (5) indicates that the results of multiple self-attention heads are spliced and converted into an output vector of a specific dimension.

$$\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \quad (4)$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \quad (5)$$

W_i^Q, W_i^K, and W_i^V are the weight matrices after the linear transformation of Q, K, and V, respectively; $W^O \in R^{d_{model} \times d_k}$ is the weight matrix for the multi-head self-attention mechanism; and h is the number of self-attention heads.

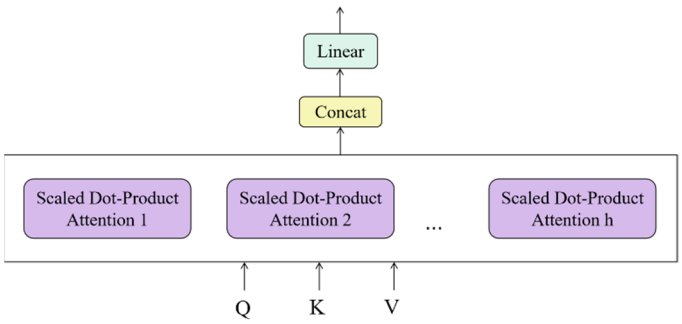

Figure 4. Multi-head attention.

The multi-head self-attention mechanism is the key to the transformer model, as it enriches the relationship between events and can even understand the semantic and syntactic structure information of sequences of events.

3.3.3. Concatenate Layer and Output Layer

The output of the transformer encoding layer is connected to an average pooling layer and output as a vector. In the concatenate layer, the vector output by the transformer encoding layer and the feature data are concatenate. In order to make the dimension of the data consistent, a batch normalization layer is added behind the feature data.

On this basis, following the full connection layer is the output layer. The output layer uses the sigmoid activation function to obtain the output, and the output result is the user's possibility of default. The formula of the output layer is as follows:

$$y = sigmoid(Wx + b) \tag{6}$$

In the process of model training, we choose cross-entropy as the loss function: cross-entropy represents the gap between the actual category of the model and the probability of the category predicted by the model. The smaller the value of the cross-entropy loss, the closer the model prediction probability and the real value. The loss function is calculated as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_1) + (1 - y_i)\log(1 - p_i)] \tag{7}$$

where y_i represents the real label of the sample, p_i represents the prediction probability of the model, and N represents the number of samples.

Finally, we select the back propagation (BP) algorithm to update the model parameters.

3.4. Evaluation Metrics

In order to test the validity of the model, we choose two commonly used indicators of credit scoring to evaluate the performance of the model: area under the receiver operating characteristic curve (AUC) and Kolmogorov–Smirnov (KS).

Let TP be the real status of the customer classified as non-default and who is judged to be non-default. FN is the real status of the customer classified as non-default and who is judged to be default. TN is the real status of the customer who is judged to be default. FP is the actual status of the customer classified as default and who is judged to be non-default. Define the True Positive Rate (TPR) as the number of TPs divided by the total number of positive customers, and define the False Positive Rate (FPR) as the number of FPs divided by the total number of negative customers; the formulae of TPR and FPR is as follows:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \times 100\% \tag{8}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \times 100\% \tag{9}$$

Taking TPR as the abscissa and FPR as the ordinate, we draw the receiver operating characteristic (ROC) curve of the model. The closer the ROC curve is to the upper left corner, the better the performance of the classifier. However, since "close to the upper left corner" is only an intuitive description of the graph, it is generally only chosen to calculate the AUC value to better quantify the degree of proximity. AUC also considers the model's ability to discriminate between defaulting customers and non-defaulting customers, avoiding the problem of model evaluation criteria failure caused by sample imbalance. The larger the AUC value, the stronger the ability of the model to Identify defaults.

The Kolmogorov–Smirnov (KS) is a commonly used credit score evaluation index, and it is mainly used to measure the model's ability to distinguish default users. After the model predicts the default probability of all samples, we sort the samples according to

the predicted default probability, calculate the cumulative TPR value and the cumulative FPR value under each default rate, and then calculate the sum of the two values under each default rate. Then, after obtaining the absolute value of the difference, we take the maximum value of these absolute values as the KS value. The larger the value of KS, the better the ability of the model to distinguish between defaulting borrowers and on-time borrowers.

4. Experimental Results

4.1. Experimental Set

Our experimental environment is a server with a Ubuntu 16 operating system, and the programming language is Python. The Python libraries used in this experiment mainly include Numpy, Pandas, Scikit-learn, Matplotlib, and Keras. Numpy is a scientific computing library of Python, and it provides the function of matrix operation; Pandas is a library mainly used for data processing and data analyses; Scikit-learn is a machine learning library; and Matplotlib is a drawing library. The deep learning framework used in this experiment is Keras (Tensorflow as the back end).

The dataset used in this research comes from an anonymous P2P lending company in China. The dataset includes feature data and behavioral data, with a total of 100,000 borrowers. The label of the data is whether the borrower defaults. If the borrower defaults, the label is 1; otherwise, the label is 0. The dataset includes five months of user loan data. To verify the stability of the model in predicting future loans, we first sort the loans according to the loan date, and then, we divide the data with the loan date in the last month into test sets. The test set data account for about 20% of the dataset, and the remaining data are divided into training sets. The training set is mainly used for training the model, and the test set is mainly used for testing model performance.

Then, data preprocessing and data coding are carried out. For user behavioral data, considering that the length of each user's behavior sequence is different, this paper converts all sequences into fixed length sequences. After a series of experiments, the length of the time series is fixed to 100, the sequences whose length exceeds 100 intercept the first 100 events, and the sequences whose length is less than 100 are filled with 0. For the FE-Transformer credit scoring model proposed in this paper, the number of transformer coding layers is set to 2, and the number of headers of the multi-head attention mechanism is set to 4. In order to alleviate the overfitting problem, Dropout [30], which is a method of dropping neural units with a certain probability from the network while training the neural network, is added to the transformer coding layer, and the dropout ratio is set to 0.3. The model training adopts mini-batch random gradient descent, the learning rate is set to 0.001, the parameter update adopts adaptive motion estimation (Adam) rules, and the early stopping strategy is adopted in the process of deep learning model training to alleviate overfitting problem.

In order to evaluate the FE-Transformer credit scoring method proposed in this paper and to prove the superiority of this approach, we conducted a detailed comparative analysis, and the comparison methods are as follows:

Logistic regression (LR): Logistic regression is the most representative of statistical models, and the input of this model is feature data.

XGBoost: XGBoost [31] is an ensemble learning algorithm, and the input data processing method of the XGBoost model is the same as that of the LR model.

LSTM: In the deep part of the model, LSTM is adopted. Firstly, user behavioral data are converted into event sequences as the input of LSTM; then, the output of LSTM is fused with the feature data; and finally, the sigmoid function is used for classification.

AM-LSTM: Using the method proposed by Wang et al. [2], the attention mechanism is added to the LSTM approach.

FE-Transformer: The approach proposed in this study.

In order to demonstrate the performance advantages of the FE-Transformer approach proposed in this study, we conducted three types of experiments, which used different

datasets: one dataset only comprises feature data, one dataset only comprises behavioral data, and one dataset comprises all data.

The first type of experiment only used feature data to examine the effect of the machine learning models on the traditional credit data. Considering that the feature data only contain feature data and have low dimensions, they are not suitable for training deep learning models, so we chose two traditional models, namely, logical regression and XGBoost. The second type of experiment used the dataset with only behavioral data. For the deep learning models of LSTM, AM-LSTM, and the transformer, user event sequences can be directly used as model input, but for the traditional machine learning models of LR and XGBoost, sequence data cannot be used as input, so we manually extracted features and selected the frequency of each event as the feature. The third type of experiment used the dataset with all the data, and the five models LR, XGBoost, LSTM, AM-LSTM, and FE-Transformer were selected for the experiment.

4.2. Performance Analysis

The results of the models only using the feature data are shown in Table 1. As can be seen in the experimental results, the AUC and KS values of the XGBoost model are higher than those of LR, indicating that the performance of the ensemble learning algorithm is superior to that of the single linear model.

Table 1. Results of models only using credit data.

Models	Training Set		Test Set	
	KS	**AUC**	**KS**	**AUC**
LR	0.23	0.622	0.23	0.621
XGBoost	0.248	0.634	0.241	0.63

The results of the models only using the behavioral data are shown in Table 2 and Figure 5. For the models only using the behavioral data, the performance of the deep learning models (LSTM and transformer) exceed that of the traditional machine learning algorithms (LR and XGBoost). This is because the deep learning algorithm can extract higher-level feature information. At the same time, consistent with the results of existing research, the effect of the AM-LSTM model is better than that of the basic LSTM model. The transformer model used in this study performs better than LSTM, AM-LSTM, and traditional machine learning models, and it achieves the highest AUC and KS values.

Table 2. Results of models only using behavioral data.

Models	Training Set		Test Set	
	KS	**AUC**	**KS**	**AUC**
LR	0.092	0.57	0.09	0.54
XGBoost	0.1	0.58	0.095	0.553
LSTM	0.203	0.631	0.198	0.62
AM-LSTM	0.243	0.66	0.238	0.661
Transformer	0.26	0.679	0.25	0.672

The input data of the models using all data include the behavioral data and feature data. The results of the models using all data are shown in Table 3 and Figure 6. From the experimental results, it can be seen that the performance of the LR and XGBoost models is better when using all data than when only using feature data. This indicates that user behavioral data can improve the prediction effect of the credit scoring model. The performance of the deep learning models (LSTM and the transformer) exceeds that of the traditional machine learning models (LR and XGBoost). The performance of the FE-Transformer model proposed in this study is better than that of the other machine

learning models, and it also achieved the highest AUC (0.72) and the highest KS values (0.32) on the test dataset.

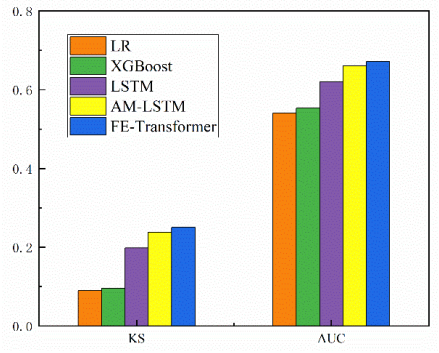

Figure 5. Performance comparison of models only using behavioral data.

Table 3. Results of models using all data.

Models	Train Set		Test Set	
	KS	**AUC**	**KS**	**AUC**
LR	0.25	0.670	0.251	0.658
XGBoost	0.262	0.679	0.26	0.665
LSTM	0.273	0.7	0.26	0.682
AM-LSTM	0.31	0.707	0.313	0.71
FE-Transformer	0.33	0.731	0.32	0.72

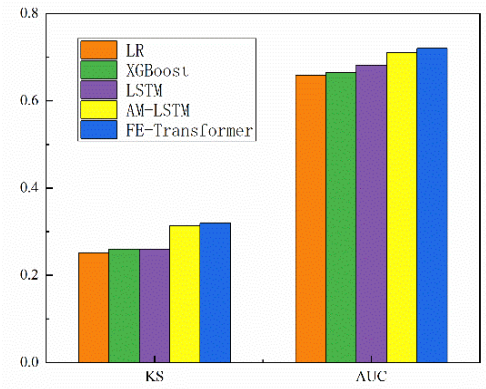

Figure 6. Performance comparison of models with all data.

4.3. Parameter Analysis

In this section, we analyzed the influence of different hyper-parameters on the performance of the FE-Transformer model. We selected two important parameters for analysis, the number of heads and the number of transformer layers. As can be seen in Figure 7, the experimental results show that, with an increase in parameters, the performance of the model increases first and then decreases. When the number of heads is set to 4, the KS and AUC of the FE-Transformer achieve the highest values, and when the number of transformer layers is set to 2, the KS and AUC of the FE-Transformer achieve the highest values. The reason for this may be that, when the hyper-parameters value is very small, the model training is not enough, so the performance of the model is general, and when the hyper-parameter values are too large, overfitting problems occur, which affect the performance of the FE-Transformer.

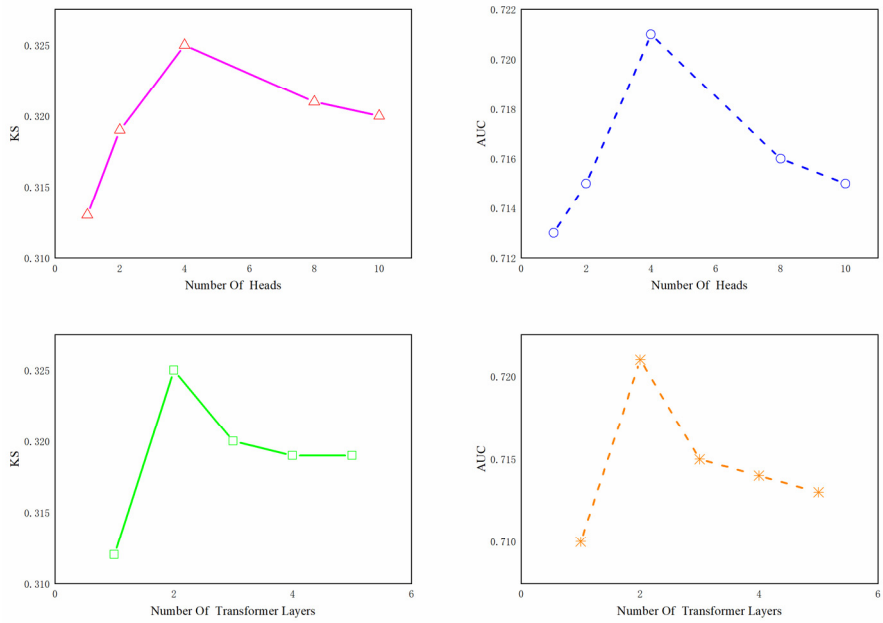

Figure 7. Influence of different hyper-parameters on the performance of FE-Transformer model.

For the deep learning model containing all data, when we fuse the feature data with the data output from the deep learning model, we added a batch normalization layer. Batch normalization can normalize the data and improve the generalization ability of neural networks [32]. In order to verify the impact of batch normalization on the performance of the model, we conducted a comparative experiment on whether to conduct batch normalization. The results are represented in Table 4 and Figure 8. For the three deep learning models LSTM, AM-LSTM, and FE-Transformer, the performance of the models with batch normalization significantly exceeds that of the models without batch normalization. The reason for this may be that batch normalization can make the output of the deep learning model be consistent with the dimension of the feature data, which is conducive to the use of the gradient descent algorithm to optimize the model.

Table 4. Performance comparison of deep learning models with normalization and without normalization.

Models	With Normalization		Without Normalization	
	KS	**AUC**	**KS**	**AUC**
LSTM	0.26	0.682	0.175	0.61
AM-LSTM	0.313	0.71	0.21	0.6
FE-Transformer	0.32	0.72	0.24	0.63

Finally, to analyze the impact of behavioral data on credit scores, we chose the XGBoost model using only the behavioral data for analysis. For this model, the input of the model is the frequency of each event. After building the XGBoost model, we extracted the Top 15 important features. The feature importance score represents the usefulness of the input feature to the user's credit default prediction; the results are shown in Figure 9. In consideration of commercial confidentiality requirements, we desensitized the event names. The results show that the feature importance of different events varies greatly, and some events have a significant prediction effect on user default risk.

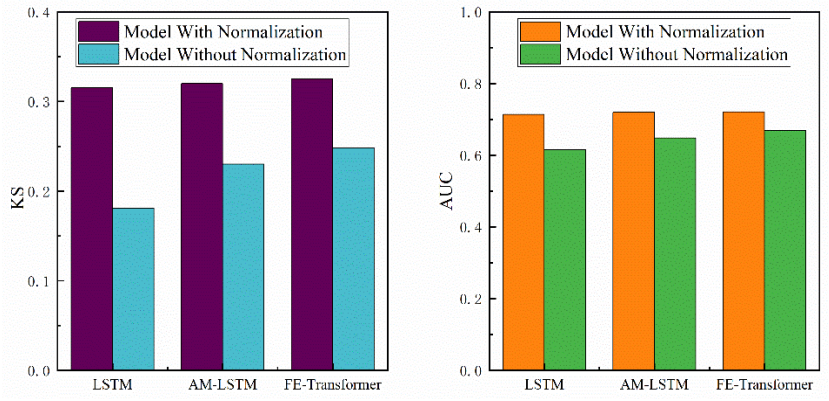

Figure 8. Performance comparison of models with normalization and models without normalization.

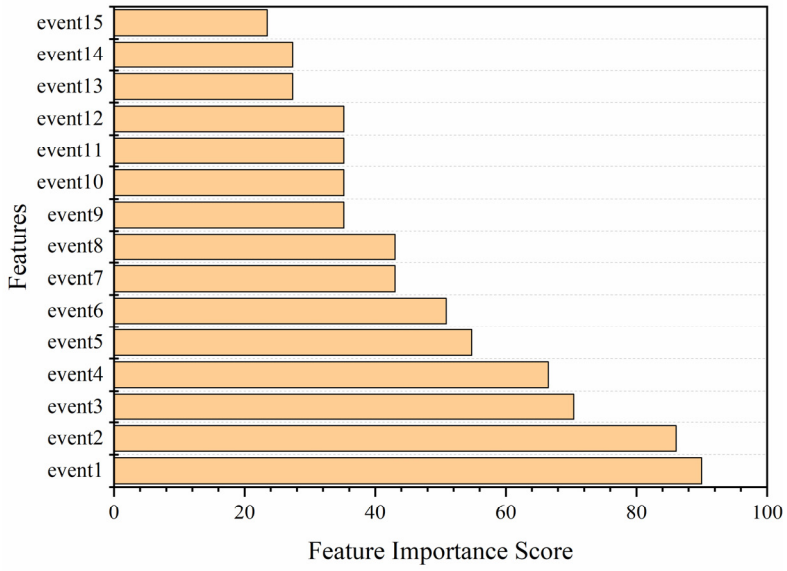

Figure 9. Feature importance of XGBoost model.

The FE-Transformer model proposed in this research outputs the predicted user default probability. The probability value is between 0 and 1. Based on this probability value, the user's credit score can be calculated. The credit score is used as the basis for loan approval and pricing. If the APP of financial institutions is upgraded, the events of user behavior will change. Therefore, after the APP is upgraded, the deep learning model needs to be updated.

To sum up, the experimental results show that the FE-Transformer model proposed in this study outperforms the LR, XGBoost, LSTM, and AM-LSTM comparison methods in terms of AUC and KS. This shows that the FE-Transformer deep learning model proposed in this research can accurately predict user default risk, which is conducive to reducing the loan default rate of financial enterprises, reducing the credit risk of financial enterprises, and maintaining the healthy and sustainable development of financial enterprises.

5. Conclusions

With the development of big data and artificial intelligence technology, deep learning models have become the research focus of credit scoring. We study the credit scoring methods of financial enterprises and propose a FE-Transformer neural network model.

The main conclusions of this study are as follows:

On the one hand, user online behavioral data provide a novel credit scoring data source. The research results show that user online behavioral data can help improve the effect of user default prediction models. On the other hand, the performance of the

FE-Transformer model proposed in this paper is better than that of the other comparison methods, and this proves the effectiveness and feasibility of this method in the field of credit scoring. The user default probability output by the model can provide the basis for loan approval decisions and the risk pricing of financial institutions, and it can help financial institutions improve their credit risk management levels and abilities.

For future research, several issues can be considered. On the one hand, due to the difficulty of data acquisition, this experiment only uses the datasets of one enterprise, and we will continue to look for other enterprise datasets for research. On the other hand, the credit scoring model in this study is a static model, and the dynamic update of credit scoring models is a research hotspot. On the basis of this study, the dynamic update of the model proposed in this research can be further studied.

Author Contributions: Conceptualization, C.W. and Z.X.; methodology, C.W. and Z.X.; experiment, C.W.; writing—original draft preparation, C.W. and Z.X.; writing—review and editing, C.W.; project administration, C.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Key R&D Plan funded by the Science and Technology Department of Shandong Province, China (No. 2019GSF108222).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, C.; Han, D.; Liu, Q.; Luo, S. A Deep Learning Approach for Credit Scoring of Peer-to-Peer Lending Using Attention Mechanism LSTM. *IEEE Access* **2018**, *7*, 2161–2168. [CrossRef]
2. Wang, C.; Liu, Q.; Li, S. A two-stage credit risk scoring method with stacked-generalisation ensemble learning in peer-to-peer lending. *Int. J. Embed. Syst.* **2022**, *15*, 158–166. [CrossRef]
3. Altman, E.I. Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *J. Financ.* **1968**, *23*, 589–609. [CrossRef]
4. Parnes, D. Applying Credit Score Models to Multiple States of Nature. *J. Fixed Income* **2007**, *17*, 57–71. [CrossRef]
5. Bolton, C. *Logistic Regression and Its Application in Credit Scoring*; University of Pretoria: Pretoria, South Africa, 2010.
6. Lessmann, S.; Baesens, B.; Seow, H.-V.; Thomas, L.C. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *Eur. J. Oper. Res.* **2015**, *247*, 124–136. [CrossRef]
7. Bhatia, S.; Sharma, P.; Burman, R.; Hazari, S.; Hande, R. Credit scoring using machine learning techniques. *Int. J. Comput. Appl.* **2017**, *161*, 1–4. [CrossRef]
8. Mandala, I.G.N.N.; Nawangpalupi, C.B.; Praktikto, F.R. Assessing Credit Risk: An Application of Data Mining in a Rural Bank. *Procedia Econ. Financ.* **2012**, *4*, 406–412. [CrossRef]
9. Harris, T. Credit scoring using the clustered support vector machine. *Expert Syst. Appl.* **2015**, *42*, 741–750. [CrossRef]
10. Abellán, J.; Castellano, J.G. A comparative study on base classifiers in ensemble methods for credit scoring. *Expert Syst. Appl.* **2017**, *73*, 1–10. [CrossRef]
11. Munkhdalai, L.; Ryu, K.; Namsrai, O.-E.; Theera-Umpon, N. A Partially Interpretable Adaptive Softmax Regression for Credit Scoring. *Appl. Sci.* **2021**, *11*, 3227. [CrossRef]
12. Malekipirbazari, M.; Aksakalli, V. Risk assessment in social lending via random forests. *Expert Syst. Appl.* **2015**, *42*, 4621–4631. [CrossRef]
13. Xia, Y.; Liu, C.; Li, Y.; Liu, N. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Syst. Appl.* **2017**, *78*, 225–241. [CrossRef]
14. Kang, Y.; Chen, L.; Jia, N.; Wei, W.; Deng, J.; Qian, H. A CWGAN-GP-based multi-task learning model for consumer credit scoring. *Expert Syst. Appl.* **2022**, *206*, 117650. [CrossRef]
15. Ghorbanali, A.; Sohrabi, M.K.; Yaghmaee, F. Ensemble transfer learning-based multimodal sentiment analysis using weighted convolutional neural networks. *Inf. Process. Manag.* **2022**, *59*, 102929. [CrossRef]
16. Bae, J.-H.; Yu, G.-H.; Lee, J.-H.; Vu, D.T.; Anh, L.H.; Kim, H.-G.; Kim, J.-Y. Superpixel Image Classification with Graph Convolutional Neural Networks Based on Learnable Positional Embedding. *Appl. Sci.* **2022**, *12*, 9176. [CrossRef]
17. Liu, Q.; Mu, L.; Sugumaran, V.; Wang, C.; Han, D. Pair-wise ranking based preference learning for points-of-interest recommendation. *Knowl.-Based Syst.* **2021**, *225*, 107069. [CrossRef]

18. Tomczak, J.M.; Zięba, M. Classification restricted Boltzmann machine for comprehensible credit scoring model. *Expert Syst. Appl.* **2015**, *42*, 1789–1796. [CrossRef]
19. Yu, L.; Yang, Z.; Tang, L. A novel multistage deep belief network based extreme learning machine ensemble learning paradigm for credit risk assessment. *Flex. Serv. Manuf. J.* **2015**, *28*, 576–592. [CrossRef]
20. Zhang, Z.; Wang, Z. Research on Credit Scoring Based on Transformer-CatBoost Network Structure. In Proceedings of the 2022 IEEE 12th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 15–17 July 2022; pp. 75–79.
21. Hidasi, B.; Quadrana, M.; Karatzoglou, A.; Tikk, D. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 241–248. [CrossRef]
22. Lang, T.; Rettenmeier, M. Understanding consumer behavior with recurrent neural networks. In Proceedings of the Workshop on Machine Learning Methods for Recommender Systems, Houston, TX, USA, 27–29 April 2017.
23. Hochreiter, S.; Schmidhuber, J.U.R. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
24. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338. [CrossRef]
25. Guo, D.; Zhou, W.; Li, H.; Wang, M. Hierarchical lstm for sign language translation. *Proc. AAAI Conf. Artif. Intell.* **2018**, *32*, 6845–6852. [CrossRef]
26. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.U.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2017; p. 30.
27. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
28. Zhang, X.; Gao, T. Multi-head attention model for aspect level sentiment analysis. *J. Intell. Fuzzy Syst.* **2020**, *38*, 89–96. [CrossRef]
29. Jing, H.; Yang, C. Chinese text sentiment analysis based on transformer model. In Proceedings of the 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), Zhuhai, China, 14–16 January 2022; pp. 185–189.
30. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
31. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
32. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6 July 2015; pp. 448–456.