

Explainable Enterprise Credit Rating via Deep Feature Crossing Network

Weiyu Guo, Zhijiang Yang, Shu Wu, Fu Chen

Abstract—Due to the powerful learning ability on high-rank and non-linear features, deep neural networks (DNNs) are being applied to data mining and machine learning in various fields, and exhibit higher discrimination performance than conventional methods. However, the applications based on DNNs are rare in enterprise credit rating tasks because most of DNNs employ the “end-to-end” learning paradigm, which outputs the high-rank representations of objects and predictive results without any explanations. Thus, users in the financial industry cannot understand how these high-rank representations are generated, what do they mean and what relations exist with the raw inputs. Then users cannot determine whether the predictions provided by DNNs are reliable, and not trust the predictions providing by such “black box” models. Therefore, in this paper, we propose a novel network to explicitly model the enterprise credit rating problem using DNNs and attention mechanisms. The proposed model realizes explainable enterprise credit ratings. Experimental results obtained on real-world enterprise datasets verify that the proposed approach achieves higher performance than conventional methods, and provides insights into individual rating results and the reliability of model training.

Index Terms—Feature Crossing, Explainable Learning, Credit Rating, Attention Mechanism.



1 INTRODUCTION

The enterprise credit rating task attempts to predict the credit rating of a company by mining related data, which is critical to many financial applications, e.g., loan [1], credit guarantee and venture investment [2]. In practice, this problem is very challenging because: (1) the information of a company is typically multi-source and heterogeneous, which results in sparse, multi-type, and high-dimensional features, and (2) accurate predictions depend on effective high-rank features because such features can add nonlinearity to the data and improve the performance of learning methods. For example, the second-rank feature, “profit \otimes revenue” often indicates the repaying ability of a company and makes sense for the enterprise credit rating task. However, it is time-consuming to obtain hand-craft high-rank features and impossible to enumerate test various combinations in polynomial fitting time, due to the input features with the property of sparse and high-dimensional. Finally, automatically generated effective high-rank features are highly efficient, which is an appealing characteristic in many real-world applications, e.g., medical treatment and fraud detection.

With the rapid development of deep learning, deep neural networks (DNNs) are receiving increasing attention in many fields, and even have become ubiquitous in a variety of applications, e.g., image processing [3], [4] and natural language processing [5], [6], because they can ex-

tract valuable high-rank features automatically from original data without artificial feature engineering. Naturally, several previous studies [7], [8], [9], [10] have applied DNNs to the enterprise credit rating task to automatically learn the low dimensional representations of company credit. However, DNN-based forecasting models typically lack interpretability. Users neither understand the meaning of the high-rank representations generated by DNNs nor catch on the inference process of DNN models. This kind of information asymmetry and opacity shakes the norm of fair lending, which is a fundamental principle of the financial industry. In other words, users in finance field do not trust the predictions provided by a “black box” model. Thus, one significant challenge of using DNN models to predict enterprise credit ratings is that the “reason codes” should be provided to users. For example, users require an easy understood explanation of why they were denied credit, especially, when the basis for denial is the output from an opaque machine learning algorithm.

Recently, several studies [11], [12], [13] have been devoted to model high-rank feature interactions using DNNs to improve the performance and interpretability of forecasting models. Specifically, multiple fully-connected layers are typically used to learn the high-rank feature interactions in an implicit manner, as well as low-dimensional representations of samples. However, such kinds of methods suffer from two limitations. First, fully-connected neural networks are inefficient in terms of learning multiplicative feature interactions [14]. Second, these models learn the feature interactions in an implicit manner, thus they lack good explanation to answer which feature combinations are meaningful. These limitations raise us to seek a new approach that is able to learn high-rank feature combinations explicitly for the task of enterprise credit rating task while offering a channel to penetrate “black box” models.

In this paper, to model the credit ratings of enterprises

- Weiyu Guo, Fu Chen and Zhijiang Yang are with the Department of Information School, Central University of Finance and Economics, Beijing 102206, P.R.China.
E-mail: {weiyu.guo, fu.chen}@cufe.edu.cn, ericyang7816@foxmail.com
- Shu Wu is with the Center for Research on Intelligent Perception and Computing (CRIPAC), National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing 100864, China, and the University of Chinese Academy of Sciences (UCAS), Beijing 100190, P.R.China.
E-mail: shu.wu@nlpr.ia.ac.cn

using deep neural network with output readable explanations, we propose a novel attention mechanism based deep neural network called DeepCross to explicitly learn effective high-rank feature combinations and predict enterprise credit ratings. In our model, to cope with sparse, multi-type, and high-dimensional features, both the categorical and numerical features are first embedded into an identical low-dimensional space, which reduces the dimension of the sparse features and allows different types of features to interact with each other. Then, inspired by self-attention, in considering a raw feature as a field query, the last high-rank representation as the key and value, we afterwards construct a series of novel feature crossing modules to explicitly learn effective patterns of high-rank feature combinations explicitly from the given dataset. As a result, we generate static explanations to penetrate the process of the proposed model's training to investigate the reliability of the model. Finally, by leveraging dual attentions, i.e., attributive and temporal attention, the proposed model adaptively indicates informative features and important time points for samples. Thus, we can provide personalized explanations for a given sample and rating pair.

Our primary contributions are summarized as follows:

- We propose to study the problem of explicitly and automatically learning high-rank feature crossing in enterprise credit rating and meanwhile construct end-to-end DNN based model called DeepCross with good explainability for the target problem.
- A feature crossing approach based on attentive neural network is proposed. It can learn high-rank feature interactions automatically from both categorical and numerical input features, and support the generation of static explanations to investigate the proposed model's training process.
- Dual attention modules, i.e., an attribute and temporal attention modules, are proposed to recognize the informative features and important time points of the given samples. As a result, personalized reasons of an enterprise credit rating can be furthered to insight.
- A series of experiments are conducted to validate the proposed model. The results demonstrate that the proposed model can obtain more precise enterprise credit ratings than conventional approaches and can provide multiple pipelines to insight the predictive process and results for users.

2 RELATED WORK

The goal of this study is to propose a deep feature crossing model to obtain accurate and explainable enterprise credit ratings, thus it is relevant to three lines of study: (1) credit rating approaches for enterprises, (2) feature interactions learning techniques, and (3) attention mechanisms in the deep learning context.

2.1 Enterprise Credit Rating

Enterprise credit rating is an intermediary service in the financial field, which has existed for over 100 years. A mount of approaches has been proposed to handle this problem, and such approaches can be categorized into factor

analysis-based methods [15], statistic-based methods [16], and model-based methods [17].

Typically, factor analysis-based methods [15] are usually to score the credit-related factors of an enterprise based on expert experience, which can be applied flexibly to qualitative analysis of enterprise credit. However, such methods are highly dependent on the subjective judgment of experts and lack the ability analysis enterprise credit quantitatively. Differing from factor analysis-based methods, statistic-based methods quantify the enterprise credit rating based on the company's financial indicators. For example, the Z-Score [16] treats the linear weighted sum of given financial indicators as the credit score of the company. The weights in the Z-Score model are calculated using the historical data of similar companies. However, such methods lack generalizability because the weights and score thresholds are fixed by experts who based on the statistical results of the historical data and their own experience to provide.

With the development of machine learning technologies, model-based approaches, e.g., logistic regression [17] and decision tree [18], have been used for the enterprise credit rating problem. For example, logistic regression [17] is often used (rather than the Z-Score [16]) to handle the large-scale credit rating task. In addition, the decision tree [18] is also popular for the credit rating task because it can generate interpretable decision rules. However, as the features of companies become increasingly complex, the prediction performance of these models based on shallow feature representations is getting harder to be promoted. Due to the strong ability of feature representation abilities of DNNs, recent model-based credit rating approaches have transformed from traditional linear or nonlinear models to deep models [8], [19]. Most of these models leverage recurrent neural networks (RNN) or convolutional neural networks (CNN) to learn the feature representation of credit from raw inputs, and then use multilayer feed-forward networks to predict the credit ratings. Based on this basic paradigm, although a higher accuracy can be achieved than the traditional models that use shallow feature representations, the deep models are typically considered as "black boxes" that cannot provide the required explanations of predictions. Thus, users neither can understand the meaning of the feature representations generated by DNNs not can catch on the inference process of DNN models. Generally, users in the field of finance do not trust predictions obtained using "black box" models.

2.2 Learning Feature Crossing

Feature crossing is a promising way to capture the interactions among raw features, and it is widely used to enhance the performance of many predictive tasks, e.g., click-through rate [11], [12], [20] and financial analysis [8], [19]. The results of feature crossing can indicate the cooccurrence of features and add nonlinearity to data, which can improve the performance of learning methods significantly.

Factorization Machines (FM) [21] and its extensions [22], [23] are well-known examples of learning feature interactions, which were proposed to capture the first-rank and second-rank feature interactions and have been proved

effective for many tasks. However, modeling only low-rank feature interactions limits performance improvements. Thus, some recent studies [11], [12], [20] have modeled high-rank feature interactions using DNNs to improve the power of expression. Most of these deep models follow the paradigm of embedding and stacked DNNs. Based on this paradigm, both categorical and numerical features are represented with low-dimensional vectors, and then feed-forward networks are utilized to learn the representation of high-rank feature interactions from their feature embedding. However, these approaches learn feature interactions in implicit manners, thus they lack explainability.

In contrast, several studies have investigated learning feature interactions in explicit manners. For example, previous studies [12], [13] performed explicit feature interactions by taking the outer product of features at the bit-wise or vector-wise level. However, it is important to explain which combinations are useful, because enumerating all crossing features is both impossible and unnecessary, and pursuing this leads to excessive computational complexity and may generate irrelevant or redundant feature interactions. In addition, tree-based models [24], [25], [26] have been used to conduct meaningful feature interactions. However, in such methods, the training procedure was broken into multiple stages. Moreover, they rely on a certain amount of human experience and lack versatility. Finally, previous studies [11], [27] combined the power of embedding-based models and attention mechanisms [6], [28], [29] to learn high-rank feature interactions and identify useful feature combinations. Differing from existing studies, we explicitly model feature interactions using a self-attention mechanism in an “end-to-end” manner. Moreover, the proposed approach probes the learned feature combinations via lasso-based feature selection [30]. As a result, we can learn compact and explainable feature combination patterns automatically.

2.3 Attention Networks

Attention was first proposed in the context of neural machine translation [31] and has been proved effective in a variety of tasks, e.g., question answering [32], text summarization [33], and recommender systems [29]. Recently, the self-attention mechanisms [6] have been used frequently to construct transformer models, from natural language processing [5] to computer vision [34], [35], which leverage multi-heads self attention [6] to well capture the relationships within the features. Unlike previous methods that use attention techniques to improve model accuracy, the proposed model employs the latest deep learning-based attention techniques [6], [29] to alleviate the lack of interpretability of deep learning methods. We employ attention techniques to explicitly take feature crossing and adaptively select features.

3 DEEP FEATURE CROSSING NETWORK

In this section, we first give an overview of DeepCross, the proposed deep feature crossing network, which can automatically learn high-rank feature crossing for the enterprise credit rating task, and identify the meaningful feature combinations and time points. We then present a comprehensive description of how low-dimensional representations

are learned explicitly for high-rank combination features without manual feature engineering, which are then used to generate accurate enterprise credit ratings.

3.1 Overview

The goal of the proposed approach is to map the original sparse and high-dimensional feature vector into low-dimensional spaces and model the high-rank feature interactions explicitly. As shown in Fig. 1, the proposed method takes the sparse sequential data as input, followed by an embedding layer that projects both numerical and categorical features into the same low-dimensional feature space. Then, we feed the embedding of all features into a series of stacked feature crossing blocks (Fig. 2), which are implemented using a self-attentive neural network and a principal component analysis (PCA) layer. For each feature crossing block, high-rank features are automatically combined using a self attentive mechanism, and the useful feature combinations are selected using a PCA layer to avoid irrelevant and redundant feature interactions, and reduce computational complexity. By stacking multiple feature crossing blocks, different ranks of feature combinations can be generated, which are the low-dimensional representations.

The outputs of the feature crossing blocks are then used to estimate the credit rating of a company. To obtain more accurate and explainable estimations, a feature attention module and a temporal attention module are stacked after the feature crossing stage. The feature and temporal attention modules are used to model the correlations of feature combinations and their temporal dependence, respectively.

3.2 Input and Embedding Layer

We treat the raw attributes of a company as a sequential data, and represent the t -th element in the sequence as a sparse vector $v_t = [a_1; a_2; \dots; a_n]$, which is the concatenation of all feature fields including both numerical and categorical attributes. Here, n is the total number of feature fields, a_i is the attribute representation of the i -th feature field, a_i is a one-hot vector if the i -th feature field is categorical (e.g., a_1 in Fig. 1), otherwise a_i is a scalar value, and the i -th feature field is numerical (e.g., a_n in Fig. 1).

The feature representations of the categorical features may be sparse and high-dimensional. A common method is to project them into a low-dimensional space. Inspired by word embeddings [36], [37], we first treat each field of the categorical attributes as a lexicon and each possible value in this field as a word. Then, we can learn the low-dimensional vector representation for each categorical attribute with an embedding layer. Specifically, we represent a given categorical feature a_j with a low-dimensional vector:

$$e_j = a_j \cdot L_j \quad (1)$$

where $L_j \in \mathbb{R}^{c_j \times d}$ is an embedding matrix for field j , and a_j is an one-hot vector. c_j represents the number of categories in the field j , and d is the dimensions of the embedding vector. In addition, considering that some categorical features can be multi-valued, e.g., the business scope of a company may cover several lines and different business lines have different contributions to the company. Therefore, if required, we process the a_j to be a probability

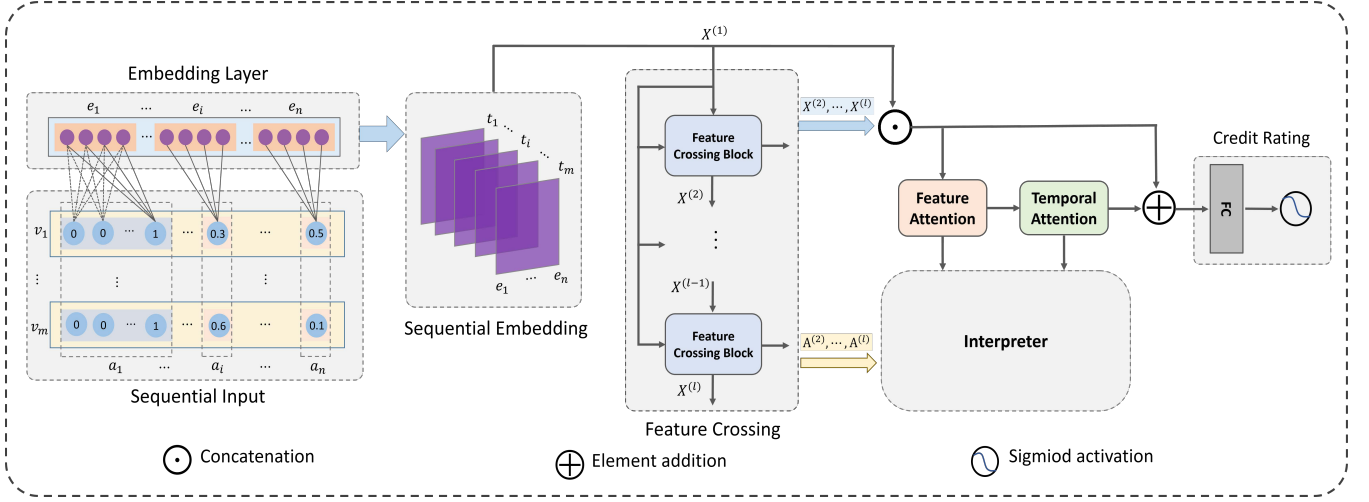


Fig. 1: Overview of proposed model. The embedding layer projects both numerical and categorical features into the same low-dimensional feature space. The details of feature crossing block are illustrated in Fig. 2, which automatically and explicitly learn the meaningful feature combinations for the task of enterprise credit rating. The feature attention module and temporal attention module are used to model the correlations of feature combinations and their temporal dependence, respectively.

distribution vector, and the value of each dimension denotes the importance of the related category.

To realize the feature crossing can be realized between categorical and numerical features, we also represent the numerical features in a low-dimensional feature space, which is same as the feature space of categorical features. Specifically, we initialize a learnable matrix $B \in \mathbb{R}^{k \times d}$ as a basis matrix, in which the i -th row represents the basis of the i -th numerical feature in a d -dimensional feature space. Then, a given numerical feature a_i can be expressed as follows:

$$e_i = a_i \cdot b_i \quad (2)$$

where $b_i \in B$ is the basis vector of the numerical feature a_i .

By doing with the embedding layer, the sequential data are transformed as the sequential embeddings, i.e., multiple 2D feature maps (Fig. 1).

3.3 Feature Crossing Module

After both the numerical and categorical features are projected into the same low-dimensional space, we further model high-rank feature combinations in the representation space. Here, the key problem is to determine which features should be combined to form meaningful higher-rank features. Traditionally, this problem has been partly tackled by domain experts who create meaningful combinations based on their experience. In fact, human experts can only design some low-rank combinations, e.g., cost-benefit ratio, because enumerating and imagining all high-rank feature combinations is impossible to human. Thus, we tackle this problem using a neural network module inspired by self-attention mechanism and PCA.

Recently, the self-attentive network [6] has achieved remarkable performance in self-driven feature correlation modeling. It demonstrates superior performance when modeling arbitrary word dependency in machine translation [38], [39] and the long-range dependencies of pixels in image analysis [40]. Here, we extend this technique to learn the correlations between different ranks of features

and generate effective higher-rank feature combinations. Specifically, we utilize the key-value attention mechanism

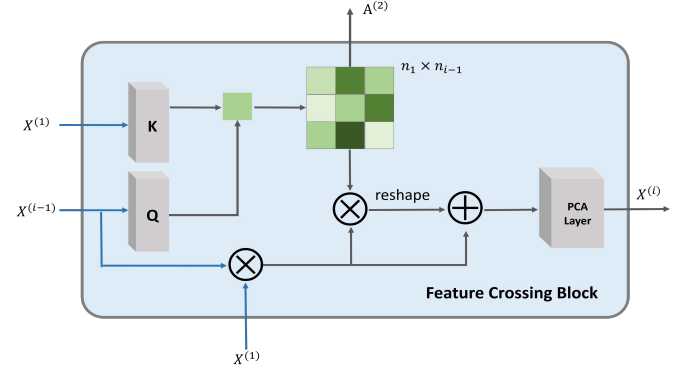


Fig. 2: Feature crossing block, where \oplus indicates the element wise addition and \otimes is a Cartesian product. Grey cubes represent the convolutional networks.

[6] to dynamically determine which feature combinations are meaningful. As shown in Fig. 2, taking the generation of l -th rank feature combinations as an example, we explain how to identify meaningful high-rank feature combinations from the candidate set, which is a set of all outer products of features. We define the first-rank features $X^{(1)} \in \mathbb{R}^{t \times n_1 \times d}$ and $i-1$ -th rank features $X^{(i-1)} \in \mathbb{R}^{t \times n_{i-1} \times d}$ as the input of a specific feature crossing module. We then perform a vector-level crossing product operation between $X^{(1)}$ and $X^{(i-1)}$:

$$X^{(t,i)} = X^{(t,1)} \otimes X^{(t,i-1)} \quad (3)$$

where $X^{(t,i)}$ and $X^{(t,i-1)}$ are feature maps of the t -th time point in $X^{(i)}$ and $X^{(i-1)}$, respectively. By adopting the key-value attention mechanism, the correlations of features between $X^{(1)}$ and $X^{(i-1)}$ can be expressed as follows:

$$a_{m,k} = \frac{\exp(\Psi(x_m^{(i-1)}, x_k^{(1)}))}{\sum_{j=1}^{n_{i-1}} \exp(\Psi(x_j^{(i-1)}, x_k^{(1)}))} \quad (4)$$

$$\Psi(x_m^{(i-1)}, x_k^{(1)}) = \langle f(W_{query}, x_m^{(i-1)}), f(W_{key}, x_k^{(1)}) \rangle$$

where $\Psi(*, *)$ is an attention function that is used to define the correlation between the m -th feature $x_m^{(i-1)} \in X^{(t,i-1)}$ and the k -th feature $x_k^{(1)} \in X^{(t,1)}$. Here, we adopt the inner product of the input vectors, i.e., $\langle *, * \rangle$, as the attention function. $f(*, *)$ represents a convolution layer with a filter kernel size of 1×1 . The convolution layer aggregates the temporal inputs into a feature map. W_{query} and $W_{key} \in \mathbb{R}^t$ are the learnable parameters of the convolution layers, respectively. Finally, to learn the effectiveness of the i -th rank features $X^{(i)}$, we update the representation of the crossing feature $x_{m,k}^{(t,i)} \in X^{(t,i)}$ in d -dimensional feature space with residual connections guided by attention coefficients $a_{m,k}$:

$$\tilde{x}_{m,k}^{(t,i)} = g(a_{m,k} \cdot x_{m,k}^{(t,i)} + x_{m,k}^{(t,i)}) \quad (5)$$

where $g(*)$ is a non-linear activate function. Here, we adopt the leaky rectified linear units [41] as the activate function, which negative slope is 0.1.

As the feature rank increases, the number of corresponding combinations increase exponentially. This problem leads that enumerating all possible high-rank features will increase the memory and computation exponentially. In fact, only a few high-rank features are effective for target task. Therefore, we utilize a convolution network to construct PCA (Fig. 2) on the candidate feature combinations. Here, a point-wise convolution neural network, which filters are conducted lasso regularization [30] in the training stage, is used to explicitly extract the meaningful features and reduce computational costs. The implementation of the PCA layer is expressed as follows:

$$\arg \min_W L(y, f(W, X^{(i)})) + \sum_{k=1}^{c_o} \|W_{:,k}\|_1 \quad (6)$$

where $L(*, *)$ is the loss function for target label y . $f(*, *)$ represents a convolution neural network which the number of input channels is $c_i = n_1 \times n_{i-1}$, and the number of output channels is a hyper-parameter $c_o = n_i$. $W \in \mathbb{R}^{c_i \times c_o}$ is the learnable parameters of 1×1 convolution kernels. Due to the imposition of lasso on learnable parameter $W_{:,k}$, most of the elements in $W_{:,k}$ trend to zero after effective model training, thus we can indicate which feature combinations in $X^{(i)}$ are useful for the target task by locating the non-zero values of W . Finally, the representations of meaningful feature combinations $[X^{(2)}, \dots, X^{(l)}]$ and their conditioned on attention weights $[A^{(2)}, \dots, A^{(l)}]$ are generated using a battery of stacked $l-1$ feature crossing modules. Then we collect the different rank feature combinations as follows:

$$\tilde{X} = X^{(1)} \odot X^{(2)} \odot \dots \odot X^{(l)} \quad (7)$$

where \odot is the concatenation operator. As a result, we can obtain mixed multi-rank features about target task, and each of these features has explicit combinatorial semantics.

3.4 Feature Attention Module

Once the combination features are generated in the same low-dimensional space, we further use a module of feature attention to learn the influence of different features on final enterprise credit rating. In order to adaptively calculate the attention scores of each feature, which indicate the correlations between result of enterprise credit rating and

features, we assign each feature $\tilde{X}_{:,i,:} \in \tilde{X}$ with a learnable parameter matrix $W_i \in \mathbb{R}^{t \times d}$, and calculate the attention scores of features for each input sample as follows:

$$a_i = \frac{\exp(\tilde{X}_{:,i,:} \otimes W_i)}{\sum_{j=1}^n \exp(\tilde{X}_{:,j,:} \otimes W_j)} \quad (8)$$

where \otimes is an arithmetic operation that first performs element-wise multiplication, and then calculates the sum of all the results. Here, we realize this arithmetic operation using a convolution neural network, where the filter kernel size of $t \times d$, the number of its output channels is n as same with the number of features. t is the number of time points.

To preserve the information of previously learned combination features, we add standard residual connections to the end of this module. Formally, the output of this module is expressed as follows:

$$\tilde{X}_{:,i,:} = g(a_i \cdot \tilde{X}_{:,i,:} + \tilde{X}_{:,i,:}) \quad (9)$$

where $g(*)$ represents the rectified linear units (ReLU), which can add the nonlinearity into the proposed model.

3.5 Temporal Attention Module

The temporal attention module attempts to learn which time points are more informative in the sequential data, and facilitates the following credit rating in consideration of the features of critical moments. In addition, the feature fluctuations of in adjacent time points are often key patterns for credit rating forecasting, thus, we realize this attention using sliding kernels on the input sequence. Here, let $\tilde{X}^{(t,:)}$ be the features of center time point t and s be the width of the sliding kernel. We calculate the attention weights for each time point in sequence as follows:

$$\begin{aligned} \mathbf{X}_{s,t} &= [\tilde{X}_{t-\frac{s+1}{2},:,}, \dots, \tilde{X}_{t, :,}, \dots, \tilde{X}_{t+\frac{s+1}{2}, :,}] \\ a_t &= \frac{\exp(\mathbf{X}_{s,t} \otimes W_s)}{\sum_{j=1}^T \exp(\mathbf{X}_{s,j} \otimes W_s)}, t \in [1, T] \end{aligned} \quad (10)$$

where \otimes is an arithmetic operation that first performs element wise multiplication, and then calculates the sum of all the results. $W_s \in \mathbb{R}^{s \times n \times d}$ is a learn-able tensor. We realize arithmetic operation \otimes using 3D-convolution neural network, where convolution kernel is W_s and the number of output channels is one.

To preserve the information of previously learned combination features, we add standard residual connections to the end of this module. Formally, we obtain the following:

$$\tilde{X}_{t, :,} = g(a_t \cdot \tilde{X}_{t, :,} + \tilde{X}_{t, :,}) \quad (11)$$

where $g(*)$ is adopted as the ReLUs.

3.6 Credit Rating and Model Training

The output of the temporal attention module is still a set of feature vectors $\{\tilde{X}^{(i)}\}_{i=1}^l$, which includes all time points of the feature maps learned via the $l-1$ feature crossing modules, feature attention module, and temporal attention module. For the final credit rating prediction, we simply concatenate all feature vectors that belong to the same time point to a vector as follows:

$$e_t = \tilde{X}_{t,1,:} \odot \dots \odot \tilde{X}_{t,n,:} \quad (12)$$

Enterprise credit reflects the operation situation of the enterprise, which is influenced by both long-term and short-term operations. To balance efficiency and performance, we use GRU [42] to model the dependency of the credit rating on long-term and short-term operations because GRU overcomes the vanishing gradients problem of RNNs and is faster than LSTM [43]. The inputs of GRU in the proposed model are the ordered features. The formulations of GRU are expressed as follows:

$$\begin{aligned} u_t &= \sigma(W_r \cdot [h_{t-1}, e_t]) \\ z_t &= \sigma(W_z \cdot [h_{t-1}, e_t]) \\ \tilde{h}_t &= \tanh(W_{\tilde{h}} \cdot [u_t * h_{t-1}, e_t]) \\ h_t &= z_t * \tilde{h}_t + (1 - z_t) * h_{t-1} \end{aligned} \quad (13)$$

where $\sigma(*)$ is the sigmoid activation function, e_t is the input feature at the t -th time point. W_r , W_z and $W_{\tilde{h}}$ are the learnable parameters of GRU. h_t is the t -th hidden states, and $*$ represents the element-wise product. Thus, we utilize the last hidden states h_T of GRU as the low-dimensional feature representation of the company, and we predict the company's credit rating as follows:

$$\tilde{y} = \sigma(W_{fc} \cdot h_T) \quad (14)$$

where $W_{fc} \in \mathbb{R}^{k \times n}$ is the project matrix that maps the low-dimensional vector h_T to the credit ratings \tilde{y} . n is the dimension of h_T , and k is the number of ratings.

To train the proposed model effectively, we leverage the L_q loss function [44] to supervise the learning process of our model. The L_q loss which supervised characteristic is somewhere in between regression loss (e.g., MAE loss) and classification loss (e.g., cross entropy loss). We do this for two reasons: 1) the credit rating task can be realized by either classification or regression in practice; and 2) MAE loss typically has good generalization but less fitting ability, while cross entropy loss is the opposite. Formally, our training process is expressed as follows:

$$\begin{aligned} \arg \min_{\mathbb{W}} L_q(y_j, \tilde{y}_j) + \sum_{i=2}^l \sum_{k=1}^{c_o} \|W_{:,k}^{(i)}\|_{l1} \\ L_q(y_j, \tilde{y}_j) = \frac{1 - (y_j \cdot \log \tilde{y}_j)^q}{q} \end{aligned} \quad (15)$$

where \mathbb{W} represents the learnable parameters of the proposed model, which are updated by minimizing the total loss using gradient descent. $W_{:,k}^{(i)} \in \mathbb{W}$ is the learnable parameters of the PCA layer in the i -th feature crossing module (Section 3.3). In addition, \tilde{y}_j is the j -th element in the predictive vector \tilde{y} . $q \in (0, 1]$ is a hyper parameter that tunes the supervised characteristic of learning between classification and regression. Note that, the loss function is equivalent to a cross entropy loss when $q \rightarrow 0$, and becomes MAE loss when $q = 0$.

4 EXPLAINABLE ENTERPRISE RATING

Through the interpreter shown in Fig.1, we attempt to generate two kinds of explanations, i.e., static explanations for the rating model and individual explanations for each rating result. These explanations can rationalize the pro-

posed model from several perspectives, i.e., training data, model reliability, and individual sample.

Owing to the imposition of lasso regularization on the parameters of PCA layers, most of the weights about feature combinations trend to zero after completing the training. Thus, given a data set and a trained model, we can identify the meaningful feature combinations for the target task as the non-zero elements in these weights. Specifically, we identify the non-zero values from the learned parameters of the PCA layers via recursive tracking. Here, assume the useful combination patterns $\Omega = \{S_i^l | l \in [1, 2, \dots, L], i \in \Lambda^l\}$ have been obtained, where Λ^l is the combination pattern set in which the associated parameters are non-zero in the l -th PCA layer. In addition, $S_i^l = \{s_j | 1 \leq j \leq l, s_j \in I\}$ is a feature combination pattern of the l -th rank, where I is the set of raw features, i.e., the first-rank features. By analyzing the static explanations Ω , financial experts can investigate the trained model to determine whether bias caused by the training sets is evident, and they can explore new financial indicators for the enterprise credit rating task.

To obtain comprehensible rating predictions, we further generate individual explanations for each prediction, which provides specific feature combinations and their weights by mining attention cues of the feature crossing, temporal attention and feature attention modules. Specifically, given attention score vectors \mathbf{p} and \mathbf{q} generated from feature attention module and temporal attention module, respectively, we generate the individual explanations by following algorithm:

$$\begin{aligned} \mathbf{E} &= \mathbf{r} \cdot (\mathbf{p} \otimes \mathbf{q}) \\ (e_1, e_2, \dots, e_k) &= \text{topK}(\mathbf{E}) \end{aligned} \quad (16)$$

where $\mathbf{E} \in \mathbb{R}^{t \times n}$ can be treated as the weights of the feature combinations at different time points. \mathbf{r} is a distribution vector of ratings, which is treated as the probabilities on different rating classes. (e_1, e_2, \dots, e_k) is a set which elements indicate the locations of top-k weights in matrix \mathbf{E} . Through the e_i , we can locate which feature combinations at which time points are more important to the given prediction, and we score its significance with $\mathbf{E}(e_i)$. Note that we can further identify the constituents of the given feature combination e_i by parsing the useful combination patterns Ω recurrently. As a result, the interpreter can provide different explanations for different input and prediction pairs.

5 EXPERIMENTS

In this section, we evaluate the effectiveness of the proposed approaches on real-world datasets, and attempt to answer the following questions:

- How does the proposed model perform on the problem of enterprise credit rating problem? Is it efficient for large-scale, sparse, high-dimensional, and multi-type data?
- Are the proposed model and its outputs explainable? How to generate and understand the explanations using our proposed methods?
- What are the influences of different model configurations on predictive performance?

5.1 Experimental Setup

5.1.1 Datasets

We evaluate the proposed approaches using two real-world datasets, i.e., the CH-Stocks and US-Stocks¹ datasets.

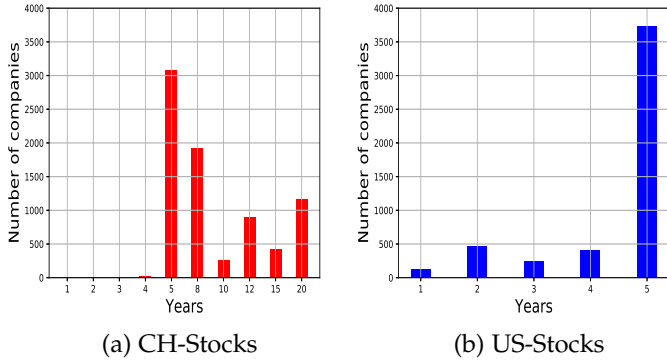


Fig. 3: Statistical distribution of the number of companies w.r.t. the time span.

CH-Stocks contains the historical data of 7968 Chinese listed companies which was crawled from multiple data sources. For each company, we collected its historical data, which contains 24 first-rank financial features (Appendix A), from its IPO to third fiscal quarter of 2019. According to the experience of investment analysts, the revenue situation typically indicates the credit rating of a company. In this experiment, we randomly split the dataset into a training set (70%) and a testing set (30%), and indirectly predict the credit rating of the company by classifying whether its revenue increased in the next fiscal quarter. The testing data contained 2451 Chinese listed companies, of which 1493 of were positive samples.

US-Stocks contain over 200 financial indicators for all the stocks of in the US stock market yearly from 2014 to 2018. The dataset was developed to understand whether it is possible to predict the future performance of a company by looking at the financial information released in financial reports. In this experiment, we treated a company which average stock price increases in next year as a positive rating. We randomly split the dataset into a training set (70%) and a testing (30%) set, and indirectly predict the credit rating of the company by classifying whether its stock price increased in subsequent year.

The list of listed companies changes with time, thus, the time horizon of the data varies for each company. As shown in Fig. 3, most companies in the experimental datasets have a continuous record of about 5 years. As a result, we primarily trained and tested the proposed model with five consecutive years on both CH-Stocks and US-Stocks datasets.

5.1.2 Competing Methods

We compared the proposed model to both deep learning-based models and conventional shallow models. Decision tree (DT), support vector machines (SVM), factorization machine (FM), gradient boosted decision trees (GBDT),

Z-Score, and logistics regression(LR) are shallow models, which are often used to evaluate the credit ratings of companies. In this experiment, we concatenated all features of a company over all time periods we used as the input data of the compared shallow models (except Z-Score). AutoInt and IFR-CNN are recently proposed deep learning-based models. AutoInt was proposed for click-through rate prediction, and IFR-CNN was designed for bankruptcy prediction. However, these deep learning-based models can be transformed to perform enterprise credit rating prediction.

Z-Score is a classic enterprise credit rating model that treats the linear weighted sum of given financial indicators as the company’s credit score, and then ratings the company by setting experience threshold scores. In this experiment, we used the Altman Z-Score model, which only considers five financial indicators with coefficients $[0.517, -0.460, 18.640, 0.388, 1.158]$, and treats the companies with Z-Score values greater than 0.9 as the positive samples.

LR is a very popular model in the field of credit assessment. It is similar to the Z-Score model as it is also a linear weighted model. However, differing from the Z-Score model, the weights of LR are learned from the training data. In this experiment, the training algorithm was realized using scikit-learn² with L_1 penalty and the “liblinear” solver. Here, we set the tolerance for the stopping criteria to 0.0001, and kept the default values for other hyper-parameters.

SVM³ is a classic kernel-based model that is effective in high-dimensional spaces, even in cases where the number of dimensions is greater than the number of samples. Therefore, it is popular in the field of credit assessment field. In this experiment, the training algorithm was realized using “libsvm”, which is “nu-SVC”, the “rbf” kernel type, and the defaults configurations for other settings.

DT is a representative algorithm for classification that is commonly used in finance. In this experiment, the maximum tree depth was set to 5, and the training algorithm was realized using scikit-learn. Note that the criterion for building the tree is evaluating the Gini impurity.

FM is a general framework that uses factorization techniques to model second-rank feature interactions and provide high prediction accuracy. In this experiment, we used libFM⁴ and the adaptive SGD learning method with learning rate 0.1, iteration times 500, and the defaults configurations for other settings to learn the second-rank feature interactions of the financial indicators and predict the credit ratings of companies.

GBDT is a boosting learning technique for both regression and classification problems that produces a prediction model in the form of an ensemble of weak decision trees. In this experiment, the training algorithm was realized using scikit-learn. We set the number of estimators to 100, and the number of random states was set 10. Default values were used for other hyper-parameters.

AutoInt⁵ automatically learns the high-rank feature interactions using multi-head self-attention, which can map both the numerical and categorical features into the same

2. scikit-learn.org/stable

3. www.csie.ntu.edu.tw/~cjlin/libsvm

4. www.libfm.org

5. github.com/shichence/AutoInt

1. www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018

low-dimensional space. In this experiment, we set the number of heads and blocks to 2 and 3, respectively, and the block shape was set to [64, 64, 64]. To facilitate fair comparison, we employed AutoInt to obtain the feature representation of a sample, and then used our feature attention module and temporal attention module to generate prediction.

IFR-CNN transforms the bankruptcy prediction to be the task of image classification by generating matrices of financial ratios. In this experiment, we realize the matrix generation approach of IFR-CNN, and generate the matrices of financial ratios by using the data of fiscal quarter or year. Then, a binary classifier based on googlenet [45] realized by torchvision⁶ is trained and tested by using the generated matrices. Its task is predicting the next situation based on current data. In other words, it only leverages data from a single time point to make predictions.

5.2 Evaluation Criteria

TABLE 1: Confusion matrix of classifiers in this experiment. $y = 1$ indicates that the model gives the company a positive outlook, $y = 0$ represents negative outlook. $g = 1$ indicates that the actual credit rating of the company is positive, and $g = 0$ is a negative outlook.

Confusion Matrix		Ground Truth	
		$g = 1$	$g = 0$
Prediction	$y = 1$	True Positive(TP)	False Positive(FP)
	$y = 0$	False Negative(FN)	True Negative(TN)

In binary classification problems, the classifiers are likely to obtain four types of predictions, i.e., true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). Therefore, we first defined the confusion matrix for our experiments as shown in Table 1. To achieve a comprehensive evaluation, the predictive performance of the compared models was evaluated using several indicators, e.g., accuracy, type I and II errors, and area under the ROC curve (AUC). Based on the confusion matrix defined in Table 1, the accuracy, type I and II errors can be defined as follows:

$$\begin{aligned}
 Acc &= \frac{TP + TN}{TN + FP + FN + TP} \\
 Err_1 &= \frac{FP}{TN + FP} \\
 Err_2 &= \frac{FN}{FN + TP}
 \end{aligned} \tag{17}$$

where, Acc indicates the accuracy of labels prediction by a given model. Err_1 and Err_2 are type I error and type II error, respectively. They can investigate the performance of binary classification with the viewpoint of different categories.

To avoid the evaluation error caused by sample imbalance, the AUC is utilized to further evaluate the quality of models in this experiment, where ROC is a comprehensive indicator reflecting continuous variables of True Positive Rate and False Positive Rate continuous variables. The AUC value is between 0.5 and 1, and a higher value is better.

6. github.com/pytorch/vision

5.3 Quantitative Analysis

5.3.1 Evaluation of Effectiveness

We summarize the quantitative results of company credit ratings obtained by different models in Table 2. The following observations can be obtained: (1) LR and SVM which are a machine learning based linear model significantly outperformed the statistics analysis-based model, i.e., Z-Score model, because they can adaptively fit the distribution of the given dataset, which may be more suitable to the company credit rating task on large-scale data. Besides, the performance of Z-Score model on US-Stocks dataset was better than CH-Stocks dataset. The reason for this phenomenon is that the weights of Z-Score model in this experiment obtained from corporate statistics in advanced economies, which may be not suite for the Chinese situation. (2) GBDT and DT, which explore high-rank feature engineering, consistently outperformed the first-rank approaches by a large margin on all datasets, which indicates that using only first-rank features may be insufficient in company credit rating prediction. (3) Benefiting from the feature engineering capabilities of DNNs and the attention mechanism, the DeepCross and AutoInt typically achieved better performance than other models. (4) The proposed model, i.e., DeepCross, obtained the best performance, which indicates that using feature crossing modules to explore deeper-rank feature interactions is crucial. Note that the proposed model shares the same structures as AutoInt (except the setup of the feature crossing modules). (5) The deep learning-based model IFR-CNN did not consistently show advantages compared to some of the shallow models. The reason for this phenomenon may be that only leveraging data from a single time point is not sufficient for the company credit rating prediction task. The company credit changes with time and it may be a sequential process.

In summary, the proposed model outperformed all compared models. Compared to the most competitive baseline i.e., AutoInt, the proposed model could explore deeper-rank feature interactions with similar resource consumption and is more efficient during online inference. This advantage is gained through the stacked feature crossing modules, which first perform explicit feature crossing via the vectorized Caresian product, and then perform PCA using a one-dimensional convolutional network.

5.3.2 Influence of Different Rank

The proposed model learns high-rank feature combinations by stacking multiple feature crossing modules. We investigated the performance of the proposed model in terms of parameter l , which is the rank of feature combinations. As shown in Fig. 4, the performance typically increased as we increased the rank of the proposed model because higher-rank feature crossing means that more feature combinations are used for prediction. However, the results obtained on two datasets differ somewhat. When the range of feature rank over 4, the performance of the proposed model on the CH-Storcks dataset began to decrease. The reason for this reduced performance was likely by the fact that the number of first-rank features in CH-Storcks dataset is small, and the fourth-rank and above features contain too many invalid feature combinations. As a result, the number of training

TABLE 2: Performance of enterprise credit rating prediction of different models. In this experiment, our model was trained to be a third-rank model both on CH-Stocks and US-Stocks, which the output dimension of the embedding layer is 64. The output dimensions of PCA layers in our stacked feature crossing modules are [128, 64, 32], which indicate the numbers of feature combinations that are retained in different rank feature crossing. DeepCross and AutoInt were trained and tested by using the data of 5 consecutive time points on US-Stocks and 15 consecutive time points on CH-Stocks.

Model class	Models	CH-Stocks				US-Stocks			
		Acc	AUC	Err ₁	Err ₂	Acc	AUC	Err ₁	Err ₂
First-rank	Z-Score [16]	0.6359	–	0.3033	0.7523	0.6987	–	0.2915	0.3660
	LR [46]	0.8464	0.9252	0.1678	0.0883	0.7256	0.7813	0.2682	0.2901
	SVM [47]	0.7926	0.8458	0.2223	0.0928	0.7269	0.7791	0.2879	0.2187
High-rank	FM [48]	0.8244	–	0.1837	0.1373	0.7249	–	0.2632	0.3029
	GBDT [49]	0.9526	0.9713	0.0397	0.0521	0.7503	0.8279	0.2546	0.228
	DT [50]	0.9314	0.9623	0.0551	0.0846	0.7342	0.8035	0.2856	0.1889
Deep-rank	AutoInt [11]	0.9612	0.9736	0.02758	0.0416	0.7497	0.7869	0.2457	0.2653
	IFR-CNN [8]	0.6716	0.7072	0.3275	0.3292	0.7127	0.7012	0.3027	0.3662
	DeepCross	0.9801	0.9955	0.0166	0.0275	0.7723	0.8341	0.2303	0.2537

samples may be relatively small compared to the feature dimension, which caused the proposed model to exhibit over-fitting.

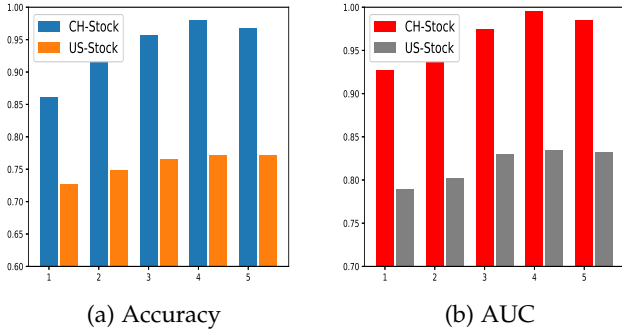


Fig. 4: Performance w.r.t. the rank of the model.

5.3.3 Influence of Different Time Spans

We predict the credit ratings of companies by sequential modeling. Therefore, we investigate the feature of our model in training and testing phase w.r.t. the parameter t , which is the time spans of data used to training and testing. As shown in Fig. 5, accuracy and AUC firstly increase as we increasing the time span on testing phase since credit rating of a company is temporal evolution. However, when the time span exceeds a certain range, the performance of our model begins to decrease on CH-Stocks. The reason of this phenomenon is probable that the increase of time span leads to significant reduction of training samples in CH-Stocks, which causes the trained model to suffer from over-fitting.

In addition, different from the testing phase, with any time spans settings, the proposed model achieved convergence on the training datasets, and the convergence speed of training was accelerated as the time span was increased. Note that as the time span increases, the number of training samples in datasets decreases. This phenomenon further indicates that the number of training samples can affect the fitting performance of the model, and increasing the number of samples may be an effective way to improve the proposed model's performance.

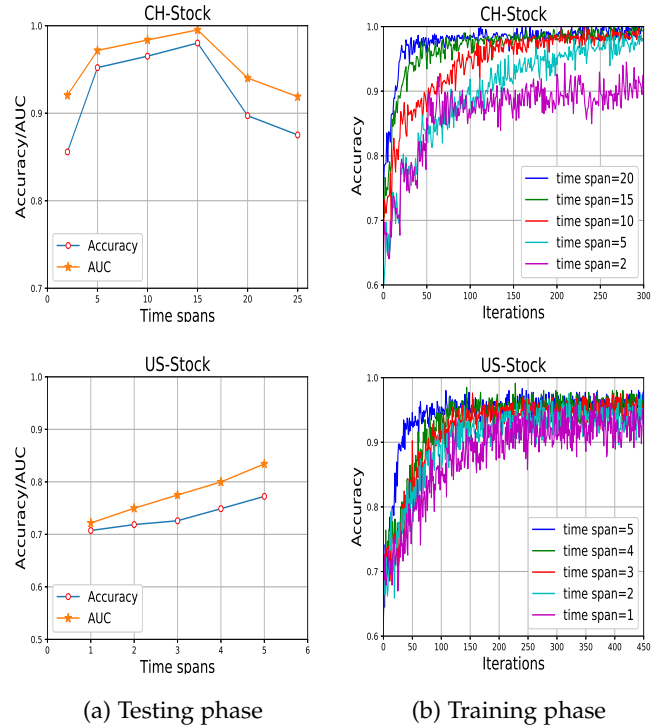


Fig. 5: Influence of different time spans in training and testing phases. In training phase, we adopted the method of stochastic gradient descent with 0.001 learning rate to train our models.

5.4 Explainable Enterprise Credit Rating

A good enterprise credit rating system can provide accuracy evaluations and good explainability of the model and the outputs. Here, we describe how the proposed model is able to explain the output results and the modeling process. Benefiting from the dual attention modules and the feature crossing modules, the proposed model not only can provide credit rating prediction for a given company, but also can generate static and personalized explanations for the modeling process and the output prediction results, respectively.

Due to the PCA layers and the lasso regularizations

TABLE 3: Most important feature combinations on CH-Stocks dataset. The f_i in table indicates a first-rank feature, which special semantics can be queried in Appendix A. l_i is the rank of features.

Top-K	l_1		l_2		l_3		l_4	
	feature	weight	feature	weight	feature	weight	feature	weight
1	f_{18}	0.3852	f_{11}, f_{18}	0.1995	f_6, f_{18}, f_{20}	0.0177	f_4, f_4, f_7, f_5	0.0500
2	f_{15}	0.0878	f_{18}, f_{20}	0.1599	f_0, f_3, f_{22}	0.0177	$f_0, f_{12}, f_{18}, f_{21}$	0.0499
3	f_{13}	0.0809	f_6, f_{18}	0.1594	f_{18}, f_{18}, f_{21}	0.0177	f_3, f_3, f_9, f_{15}	0.0250
4	f_{12}	0.0667	f_3, f_{15}	0.1198	f_9, f_{12}, f_{13}	0.0177	$f_{11}, f_{15}, f_{17}, f_{19}$	0.0250
5	f_{10}	0.0593	f_{12}, f_{18}	0.1198	f_8, f_0, f_{22}	0.0177	$f_6, f_{10}, f_{12}, f_{24}$	0.0250

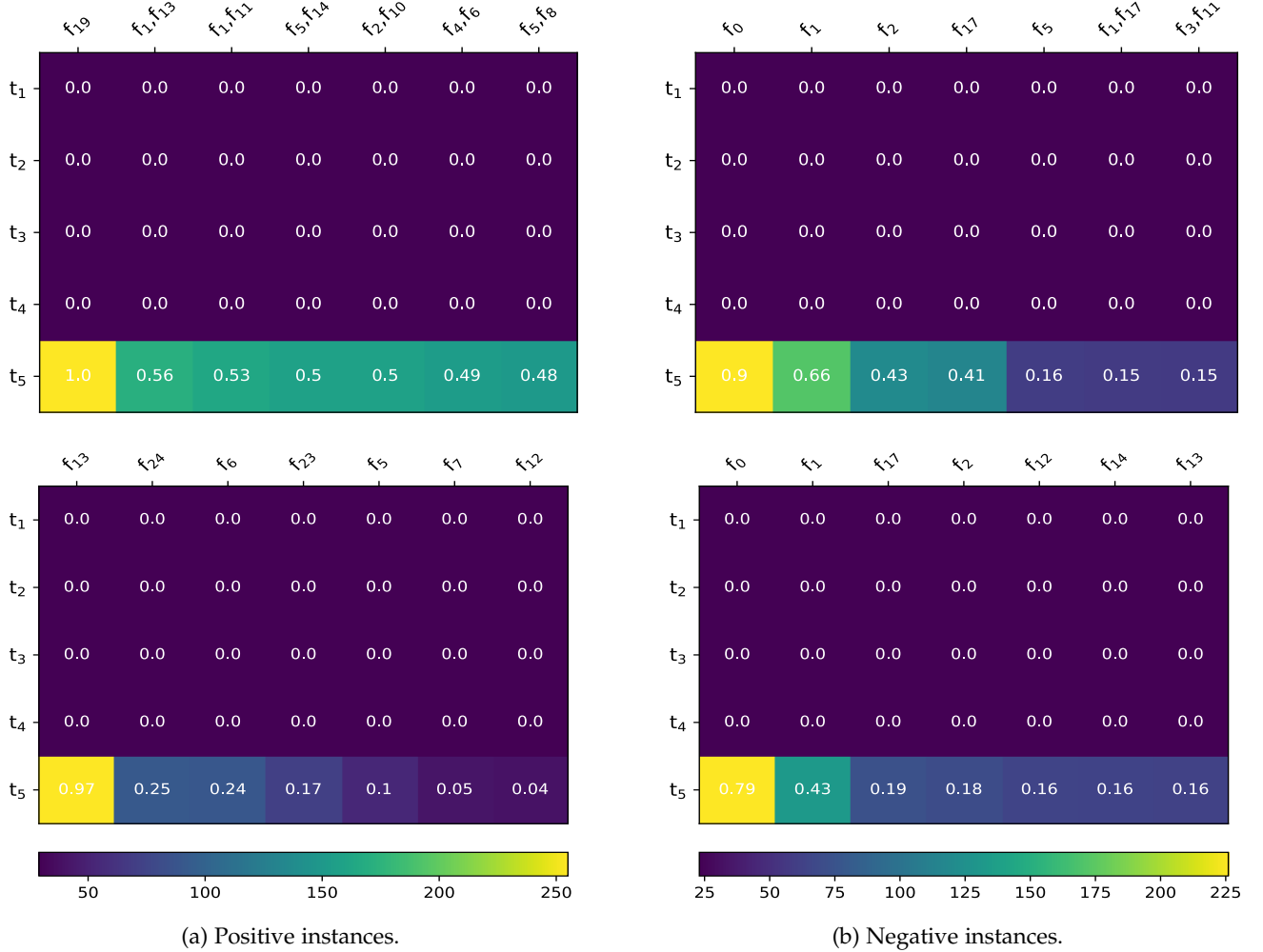


Fig. 6: Heatmap examples of input feature weights w.r.t time points. The weights were first normalized to $[0, 1]$, and then mapped to the color space $[0, 255]$. The intensity of the color blocks indicates the importance of the corresponding features. We show the seven most important features (the sum of weights on different time points is greater than others). Here, f_i indicates a feature, which special semantics can be queried in the Appendix A. t_i is a time point and t_5 is the predictive time point.

in the feature crossing modules, we can summarize the most important feature combinations in different ranks of feature sets for a given dataset. As shown in Table 3, we parsed the useful combination patterns and their weights from each PCA layer in a backtracking way (Section 4). By analyzing these static explanations, human experts in the field of enterprise credit rating can further investigate the trained model to determine whether there exists bias caused by the training datasets. For example, in the second-rank feature combinations, the combination of operation cycle

(i.e., f_{11}) and capital return (i.e., f_{18}) may be an indication of a company’s debt paying ability, and this pattern is consistent with common accounting principles. In addition, as shown in Table 3, as the rank increases, the weights of the feature combinations tend to be trivial. This phenomenon proves that the high-rank features typically include a lot of redundant and noise, thus, the feature selection in the proposed feature crossing module is necessary.

In addition, in the field of enterprise credit rating task, users are interested in the correlations between the features

of a given sample and the specific credit rating results. Therefore, we further provide a credit rating to a given company with visualized the correlations between the most important feature combinations and time points. We observe the following phenomenon from the visualized results shown in Fig. 6: (1) The features of the time points closer to the rating time point are more important to the result than the features farther away from the rating time point, e.g., the features on t_5 are most effective to both positive samples and negative samples. (2) The results of different samples are affected by different features. (3) Some features are important to both positive samples and negative samples, e.g., the net profit cut growth rate (i.e., f_{13}) can discriminate both positive and negative samples.

In summary, the proposed model demonstrates good explainability and provide both global and personalized explanations, which can assist financial experts to assess the reliability of the trained models and users to understand the rating results.

6 CONCLUSION

In this paper, we have proposed a novel deep feature crossing based model to predict enterprise credit ratings with high accuracy and explainability. The proposed model maps the original sparse and high-dimensional features into low-dimensional spaces and explicitly models the interactions of the high-rank feature. First, we construct and stack multiple feature crossing modules to generate useful high-rank feature combinations in an explicit manner. Then, by leveraging the proposed feature crossing modules, we learn static patterns of the high-rank feature combinations from the training data, which helps human experts in the field of credit rating identify bias in the trained model. Next, to obtain an accurate estimation with individual explanations, we further construct feature attention and temporal attention modules following the feature crossing stage. These attention modules are used to model the correlations of feature combinations and their temporal dependence, respectively. By mining and visualizing the Cartesian product of their attentions, the proposed method can provide personalized explanations of multiple feature combinations for a given sample and credit rating pair. In addition, we have presented methods to train the proposed model and generate corresponding explanations. The experimental results confirm that the proposed model demonstrates higher prediction accuracy than traditional enterprise credit rating models and provide explanations of both the prediction results and the training process.

However, the DNNs are a data-driven approach that can easily suffer over-fitting on small training sets. Therefore, in future, we plan to extend our experimental datasets to include non-listed companies and extend the proposed model to support non-financial information, which can effectively mitigate the problem of insufficient numbers of training samples. In addition, we would like to obtain the more accurate company representation by considering the news about companies and their propagation on social media, e.g., Weibo and Twitter to improve the performance of the proposed model.

APPENDIX A

The raw features of the listed Chinese companies used in our experiments are listed in Table A1.

TABLE A1: Attribute names of CH-Stocks dataset. f_i indicates the feature number in our explanation system.

NO.	Annotation	NO.	Annotation
f_0	Industry category	f_1	Net profit
f_2	Net profit cut	f_3	Gross revenue
f_4	Earnings per share	f_5	Net assets value per share
f_6	Capital surplus fund per share	f_7	Undivided profit per share
f_8	Operation cash flow per share	f_9	Days sales of inventory
f_{10}	Accounts receivable turnover days	f_{11}	Operation cycle
f_{12}	Net profit growth rate	f_{13}	Net profit cut growth rate
f_{14}	Operation revenue growth rate	f_{15}	Net profit ratio
f_{17}	Gross income ratio	f_{18}	Capital return
f_{19}	Return on equity	f_{20}	Inventory turning rate
f_{21}	Current ratio	f_{22}	Quick ratio
f_{23}	Super quick ratio	f_{24}	Debt equity ratio
f_{25}	Debt assets ratio	—	—

REFERENCES

- [1] N. Yoshino, F. Taghizadeh-Hesary, P. Charoensivakorn, and B. Niraola, "Small and medium-sized enterprise (sme) credit risk analysis using bank lending data: An analysis of thai smes," *Journal of Comparative Asian Development*, pp. 1–24, 2016.
- [2] B. Shi, B. Meng, H. Yang, J. Wang, and W. Shi, "A novel approach for reducing attributes and its application to small enterprise financing ability evaluation," *Complexity*, vol. 2018, pp. 1 032 643:1–1 032 643:17, 2018.
- [3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
- [4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
- [5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019.
- [6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.
- [7] P. M. Addo, D. Guegan, and B. Hassani, "Credit risk analysis using machine and deep learning models," *Risks*, vol. 6, no. 2, p. 38, 2018.
- [8] T. Hosaka, "Bankruptcy prediction using imaged financial ratios and convolutional neural networks," *Expert systems with applications*, vol. 117, pp. 287–299, 2019.
- [9] W. Guo, B. Cao, and Z. Li, "Explainable enterprise rating using attention based convolutional neural network," in *WISA*, 2020.
- [10] P. Golbayani, D. Wang, and I. Florescu, "Application of deep neural networks to assess corporate credit rating," *arXiv preprint arXiv:2003.02334*, 2020.
- [11] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, "Autoint: Automatic feature interaction learning via self-attentive neural networks," in *CIKM*, 2019.
- [12] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep and cross network for ad click predictions," in *ADKDD*, 2017.
- [13] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "xdeepfm: Combining explicit and implicit feature interactions for recommender systems," in *SIGKDD*, 2018.
- [14] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and E. H. Chi, "Latent cross: Making use of context in recurrent recommender systems," in *WSDM*, 2018.
- [15] R. R. McCrae and O. P. John, "An introduction to the five-factor model and its applications," *Journal of Personality*, vol. 60, no. 2, pp. 175–215, 1992.
- [16] Altman and E. I., "Predicting financial distress of companies: Revisiting the z-score and zeta models," *Handbook of Research Methods and Applications in Empirical Finance*, 2010.
- [17] C. Bolton et al., "Logistic regression and its application in credit scoring," Ph.D. dissertation, University of Pretoria, 2010.

- [18] Y. Xia, C. Liu, Y. Li, and N. Liu, "A boosted decision tree approach using bayesian hyper-parameter optimization for credit scoring," *Expert Systems With Applications*, vol. 78, pp. 225–241, 2017.
- [19] R. Matin, C. Hansen, C. Hansen, and P. Mlgard, "Predicting distresses using deep learning of text segments in annual reports," *Expert Systems with Applications*, vol. 132, pp. 199–208, 2019.
- [20] Q. Song, D. Cheng, H. Zhou, J. Yang, Y. Tian, and X. Hu, "Towards automated neural interaction discovery for click-through rate prediction," in *SIGKDD*, 2020.
- [21] S. Rendle, "Factorization machines," in *ICDM*, 2010.
- [22] W. Guo, S. Wu, L. Wang, and T. Tan, "Personalized ranking with pairwise factorization machines," *Neurocomputing*, vol. 214, pp. 191–200, 2016.
- [23] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, "Fast context-aware recommendations with factorization machines," in *SIGIR*, W. Ma, J. Nie, R. Baeza-Yates, T. Chua, and W. B. Croft, Eds., 2011.
- [24] Y. Luo, M. Wang, H. Zhou, Q. Yao, W. Tu, Y. Chen, W. Dai, and Q. Yang, "Autocross: Automatic feature crossing for tabular data in real-world applications," in *SIGKDD*, 2019.
- [25] X. Wang, X. He, F. Feng, L. Nie, and T. Chua, "TEM: tree-enhanced embedding model for explainable recommendation," in *WWW*, 2018.
- [26] J. Zhu, Y. Shan, J. C. Mao, D. Yu, H. Rahmanian, and Y. Zhang, "Deep embedding forest: Forest-based serving with deep embedding features," in *SIGKDD*, 2017.
- [27] Z. Tao, X. Wang, X. He, X. Huang, and T. Chua, "Hoafm: A high-order attentive factorization machine for ctr prediction," *Information Processing and Management*, vol. 57, no. 6, p. 102076, 2020.
- [28] C. Wu, F. Wu, S. Ge, T. Qi, Y. Huang, and X. Xie, "Neural news recommendation with multi-head self-attention," in *EMNLP-IJCNLP*, 2019.
- [29] S. Seo, J. Huang, H. Yang, and Y. Liu, "Interpretable convolutional neural networks with dual local and global attention for review rating prediction," in *RecSys*, 2017.
- [30] P. Zhao and B. Yu, "On model selection consistency of lasso," *The Journal of Machine Learning Research*, vol. 7, pp. 2541–2563, 2006.
- [31] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.
- [32] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *NIPS*, 2015.
- [33] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *EMNLP*, 2015.
- [34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. [Online]. Available: <https://arxiv.org/abs/2010.11929>
- [35] G. Sharir, A. Noy, and L. Zelnik-Manor, "An image is worth 16x16 words, what is a video worth?" *CoRR*, vol. abs/2103.13915, 2021. [Online]. Available: <https://arxiv.org/abs/2103.13915>
- [36] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR*, 2013.
- [37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.
- [38] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in *NAACL-HLT*, 2018.
- [39] A. Raganato, Y. Scherrer, and J. Tiedemann, "Fixed encoder self-attention patterns in transformer-based machine translation," in *EMNLP*, 2020.
- [40] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *ICCV*, 2019.
- [41] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *CoRR*, vol. abs/1505.00853, 2015.
- [42] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: <http://arxiv.org/abs/1412.3555>
- [43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [44] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *NeurIPS*, 2018.
- [45] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.
- [46] S. Y. Sohn, D. H. Kim, and J. H. Yoon, "Technology credit scoring model with fuzzy logistic regression," *Applied Soft Computing*, vol. 43, pp. 150–158, 2016.
- [47] C. Chang and C. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011.
- [48] S. Rendle, "Factorization machines with libfm," *ACM Transactions on Intelligent Systems and Technology*, vol. 3, no. 3, pp. 57:1–57:22, May 2012.
- [49] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *NIPS*, 2017.
- [50] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.