Expert Systems With Applications 220 (2023) 119704

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Explainable enterprise credit rating using deep feature crossing

Weiyu Guo [a,*], Zhijiang Yang [c], Shu Wu [b], Xiuli Wang [a], Fu Chen [a]

[a] School of Information, Central University of Finance and Economics, BeiJing, PR China
[b] Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, PR China
[c] School of Computing, National University of Singapore, Singapore, Singapore

ARTICLE INFO

Keywords:
Feature crossing
Explainable credit rating
Attention network

ABSTRACT

Deep Neural Networks (DNNs) have powerful learning abilities on high-rank and non-linear features, and thus have been applied to various fields, exhibiting higher discrimination performance than conventional methods. However, their applications in enterprise credit rating tasks are rare, as most DNNs employ the "end-to-end" learning paradigm, producing high-rank representations of objects or predictive results without any explanations. This "black box" approach makes it difficult for users in the financial industry to understand how these predictive results are generated, or what correlations exist with the raw inputs, leading to a lack of trust to the predictions. To address this issue, this paper proposes a novel network to explicitly model the enterprise credit rating problem using DNNs and attention mechanisms, allowing for explainable enterprise credit ratings. Experiments conducted on real-world enterprise datasets show that the proposed approach achieves higher performance than conventional methods, while also providing insights into individual rating results and the reliability of model training. The code is provided on https://github.com/Weyne168/creditRatting.

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, such as loan, credit guarantee, and venture investment. This problem is challenging due to the multi-source and heterogeneous information of a company, resulting in sparse, multi-type, and high-dimensional features. Accurate predictions depend on effective high-rank features, which can add non-linearity to the data and improve the performance of learning methods. For example, the second-rank feature "profit⊗revenue", which is a compound indicator that multiplies or divides the profit by the revenue, often indicates the repaying ability of a company and is meaningful for the Enterprise Credit Rating Task. However, obtaining hand-crafted high-rank features is time-consuming and enumerating various combinations in polynomial fitting time is impossible. Thus, it is desirable to find a comprehensive solution for automatically and efficiently generating effective high-rank features, which is also applicable to other real-world applications, such as medical treatment and fraud detection.

With the rapid development of deep learning, Deep Neural Networks (DNNs) have become increasingly popular in many fields, such as image processing (He, Zhang, Ren, & Sun, 2016; Krizhevsky, Sutskever, & Hinton, 2012) and natural language processing (Devlin, Chang, Lee, & Toutanova, 2019; Vaswani, Shazeer, Parmar, et al., 2017), due to their ability to automatically extract valuable high-rank features from raw data without manual feature engineering and to represent all features with a low dimensional but more effective feature representation. Previous studies (Golbayani, Wang, & Florescu, 2020; Guo, Cao, & Li, 2020; Hosaka, 2019) have applied DNNs to the enterprise credit rating task for automatically learning the low dimensional representations of company credit and predicting the company credit rating. However, DNN-based forecasting models lack interpretability due to their complex and automatic feature selection and expression. This information asymmetry and opacity shakes the norm of fair lending, which is a fundamental principle of the financial industry. As a result, users in the finance field do not trust the predictions provided by a "black box" model. Therefore, one significant challenge of using DNN models to predict enterprise credit ratings is to provide "reason codes" to users. For example, users should be given an easy-to-understand explanation of why they were denied credit, especially when the decision is based on an opaque machine learning algorithm.

In this paper, we propose a novel attention-based deep neural network, DeepCross, to model the credit ratings of enterprises with output readable explanations. To cope with sparse, multi-type, and high-dimensional features, the categorical and numerical features are

* Correspondence to: Central University of Finance and Economics, 39 South College Road, Haidian District, BeiJing, PR China.
E-mail addresses: weiyu.guo@cufe.edu.cn (W. Guo), yang.zhijiang@u.nus.edu (Z. Yang), shu.wu@nlpr.ia.ac.cn (S. Wu), wangxiuli@cufe.edu.cn (X. Wang), fu.chen@cufe.edu.cn (F. Chen).

https://doi.org/10.1016/j.eswa.2023.119704
Received 26 August 2021; Received in revised form 8 February 2023; Accepted 10 February 2023
Available online 14 February 2023

first embedded into an identical low-dimensional space, reducing the dimension of the sparse features and allowing different types of features to interact with each other. Inspired by self-attention (Vaswani et al., 2017), we construct a series of novel feature crossing modules to explicitly learn effective high-rank feature combinations for the credit ranking task. These modules take the raw feature as a field query, the last high-rank representation as a key and the next rank features as a value to learn the high-rank representations with self-attention paradigm. According to the outputs of proposed feature crossing modules in training stage, we can analyze the high-rank feature combinations bias in the dataset, and generate static explanations to investigate the reliability of the model. Additionally, leveraging dual attentions, i.e., attributive and temporal attention, the proposed model adaptively indicates informative features and important time points for samples, providing personalized explanations for a given pair of sample and rating. 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 task and meanwhile construct end-to-end Deep Neural Network (DNN) based model, called DeepCross, which has good explainability for the target problem.
- A proposed feature crossing approach based on attention-based neural network is presented. This approach can automatically learn high-rank feature interactions from both categorical and numerical input features, and provide static explanations to investigate the model's training process.
- A dual attention module is proposed to recognize the informative features and important time points about the given samples, thus enabling the furthering of personalized reasons for an enterprise's credit rating to gain insight.
- A series of experiments have been conducted to validate the proposed model, with the results demonstrating that our approach can achieve higher precision enterprise credit ratings than conventional methods. Furthermore, our approach provides multiple pipelines to provide users with insights into the predictive process and results.

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, statistic-based methods, and model-based methods.

Typically, factor analysis-based methods (Mccrae & John, 1992) 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 of quantitative analysis for enterprise credit. 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 (Altman, 2013) 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 human experts' experience.

With the development of machine learning technologies, model-based approaches, e.g., logistic regression (Bolton et al., 2010) and decision tree (Xia, Liu, Li, & Liu, 2017), have been used for the enterprise credit rating problem. For example, logistic regression (Bolton et al., 2010) is often used (rather than the Z-Score Altman, 2013) to handle the large-scale credit rating task. In addition, the decision tree (Xia et al., 2017) 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 (Hosaka, 2019; Matin, Hansen, Hansen, & Mlgaard, 2019). 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 multi-layer 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 can neither understand the meaning of the feature representations generated by DNNs nor 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 (Song, Cheng, Zhou, et al., 2020; Song, Shi, Xiao, et al., 2019; Wang, Fu, Fu, & Wang, 2017) and financial analysis (Hosaka, 2019; Matin et al., 2019). The results of feature crossing can indicate the co-occurrence of features and add non-linearity to data, which can improve the performance of learning methods significantly.

Factorization Machines (FM) (Rendle, 2010) and its extensions (Guo, Wu, Wang, & Tan, 2016; Rendle, Gantner, Freudenthaler, & Schmidt-Thieme, 2011) 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 (Song et al., 2020, 2019) have been proposed to model high-rank feature interactions using DNNs to achieve more effective feature representations. They typically follow the paradigm of embedding and stacked DNNs, which first represents both categorical and numerical original features by low-dimensional vectors, and then utilizes fully-connected neural networks to learn the representation of high-rank feature interactions from their feature embedding. However, these approaches may result in two issues. First, fully-connected neural networks are inefficient in terms of learning multiplicative feature interactions (Beutel, Covington, Jain, et al., 2018), thus they cannot obtain quality feature representation for prediction. Second, these models learn the feature interactions in an implicit manner, thus they lack effective explanation to answer which feature combinations are meaningful.

In contrast, several studies have investigated learning feature interactions in explicit manners. For example, Lian, Zhou, Zhang, et al. (2018) and Wang et al. (2017) performed to learn 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. Pursuing this leads to excessive computational complexity, and may generate irrelevant or redundant feature interactions. In addition, the tree-based models (Luo, Wang, Zhou, et al., 2019; Wang, He, Feng, et al., 2018; Zhu, Shan, Mao, et al.,

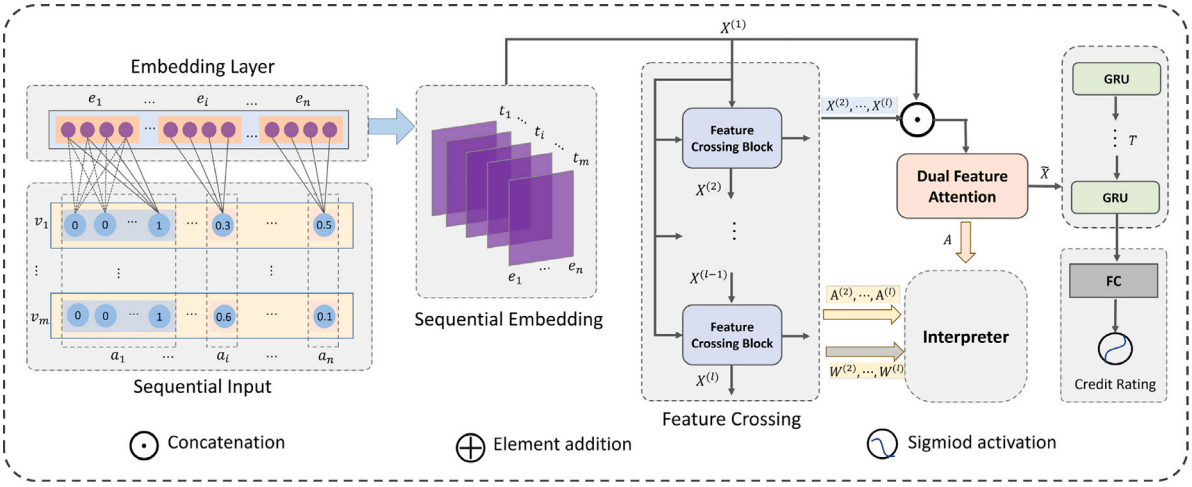

Fig. 1. Overview of proposed model. The embedding layer projects both numerical and categorical features into the same low-dimensional feature space. The details of feature crossing block are illustrated in Fig. 2, which automatically and explicitly learn the meaningful feature combinations for the task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

2017) have been used to conduct meaningful feature interactions. In such methods, the feature interaction modeling procedure is usually divided into the feature construction and feature selection, which may utilize different optimization criterion and result in a locally optimal solution. Moreover, the tree-based solutions rely on some crucial hyper-parameters, e.g., the maximum depth of tree, which setups require expert experience about given task and datasets.

Finally, previous studies (Song et al., 2019; Tao, Wang, He, et al., 2020) combined the power of embedding-based models and attention mechanisms (Seo, Huang, Yang, & Liu, 2017; Vaswani et al., 2017; Wu, Wu, Ge, et al., 2019) 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 (Zhao & Yu, 2006). As a result, we can learn compact and explainable feature combination patterns automatically.

2.3. Attention networks

Attention mechanism was first proposed in the context of neural machine translation (Bahdanau, Cho, & Bengio, 2015) and has been proved effective in a variety of tasks, e.g., question answering (Sukhbaatar, Szlam, Weston, & Fergus, 2015), text summarization (Rush, Chopra, & Weston, 2015), and recommender systems (Seo et al., 2017). Recently, the self-attention mechanisms (Vaswani et al., 2017) have been used frequently to construct transformer models, from natural language processing (Devlin et al., 2019) to computer vision (Dosovitskiy, Beyer, Kolesnikov, et al., 2020), which leverage multi-heads self attention (Vaswani et al., 2017) 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 (Seo et al., 2017; Vaswani et al., 2017) 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 present an overview of DeepCross, a proposed deep feature crossing network. This network can automatically learn high-rank feature crossing for the enterprise credit rating task, and identify which combinations of features have a significant effect on the credit rating of a given company at a given time. We then provide 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 proposed approach aims to map a sparse, high-dimensional feature vector into a low-dimensional space and explicitly model high-rank feature interactions. As depicted in Fig. 1, the model takes sparse sequential data as input, and an embedding layer projects both numerical and categorical input features into the same low-dimensional feature space. This is then fed into a series of stacked feature crossing blocks (Fig. 2), implemented using a self-attentive neural network and a principal feature selection module (PFS). Each feature crossing block combines high-rank features based on a self-attentive mechanism, and the PFS module selects useful feature combinations to avoid irrelevant and redundant interactions and reduce computational complexity. By stacking multiple feature crossing blocks, different ranks of feature combinations can be generated, forming the low-dimensional representations.

The outputs of the feature crossing blocks are used to estimate a company's credit rating. To ensure accurate and understandable estimations, a dual attention module is added after the feature crossing stage. This module models the correlations between the credit rating and the feature combinations, as well as the temporal dependence of the credit rating.

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 $v_t = [a_1; a_2; \ldots; a_n]$, which is the concatenation of all feature fields including both numerical and categorical attributes. Here, n is the total number of feature fields, and a_i is the attribute representation of the i-th feature field. If the i-th feature field is categorical (e.g., a_1 in Fig. 1), then a_i is an one-hot vector; otherwise, a_i is a scalar, as the i-th feature field is numerical (e.g., a_n in Fig. 1).

Common methods to handle sparse and high-dimensional categorical features involve projecting them into a low-dimensional space.

Inspired by word embedding (Mikolov, Chen, Corrado, & Dean, 2013), we treat each field of the categorical attributes as a lexicon, and each possible value in this field as a word. We then learn the low-dimensional vector representation for each categorical attribute with the embedding layer. Specifically, we represent a given categorical feature a_j with a low-dimensional vector.

$$e_j = a_j \cdot L_j \tag{1}$$

For a given feature field j, let $L_j \in \mathbb{R}^{c_j \times d}$ be an embedding matrix, a_j an one-hot vector, c_j the number of categories in this feature field, and d the dimensions of the embedding vector. Additionally, to account for multi-valued categorical features, such as the business scopes of a company, we can transform a_j into a probability distribution vector, where each dimension denotes the importance of the related category. For example, we can represent the revenue share of different business lines of the company as a_j.

To realize the feature crossing between categorical and numerical features, we follow the method proposed by AutoInt (Song et al., 2019) to represent the numerical features into a feature space, which dimension 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, where the i-th row represents the basis of the i-th numerical feature in a d-dimensional feature space. The number of numerical features used in this task is denoted as k. A given numerical feature a_i can be expressed as follows:

$$e_i = a_i \cdot b_i \tag{2}$$

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

After passing through the embedding layer, each raw input feature a_i is represented by a d-dimensional embedding vector e_i. These vectors are then collected into a tensor $X^{(1)} \in \mathbb{R}^{T \times n_1 \times d}$ as the sequential embedding of the first rank features, where T is the length of time points, n_1 is the total number of raw feature fields, and d is the embedding dimension.

3.3. Feature crossing module

When projecting both numerical and categorical features into the same low-dimensional space, we can further model high-rank feature combinations in the given representation space. Here, the challenge lies in determining which features should be combined to form meaningful higher-rank features. Traditionally, this has been partially addressed by domain experts, who create meaningful combinations based on their experience. However, human experts are limited to creating only low-rank combinations, such as cost-benefit ratio, since it is impossible for them to imagine and enumerate all high-rank feature combinations. To address this issue, we use a neural network module, referred to as a PFS module.

Recently, the self-attentive network (Vaswani et al., 2017) has demonstrated remarkable performance in self-driven feature correlation modeling, showing superior performance when modeling arbitrary word dependencies in machine translation Raganato, Scherrer, and Tiedemann (2020), Shaw, Uszkoreit, and Vaswani (2018) and long-range dependencies of pixels in image analysis Cao, Xu, Lin, et al. (2019). We further 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 to dynamically determine which feature combinations are meaningful. As shown in Fig. 2, taking the generation of the i-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 $i-1$-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. Here, n_{i-1} is the number of $i-1$-th rank features, and $n_1 = n$ indicates the number of the first rank features (both original categorical and numerical features). We then perform a vector-level crossing product operation between $X^{(1)}$ and $X^{(i-1)}$.

$$X^{(i)} = X^{(1)} \otimes X^{(i-1)} \tag{3}$$

By adopting the key–value attention mechanism, we extract the query, key and value information from $X^{(1)}$, $X^{(i-1)}$ and $X^{(i)}$, respectively. The correlations of features between the query and key can then be expressed as follows:

$$\begin{aligned} a^{(i)}_{m,k} &= \frac{exp(\Psi(x^{(i-1)}_m, x^{(1)}_k))}{\sum_{j=1}^{n_{i-1}} exp(\Psi(x^{(i-1)}_j, x^{(1)}_k))} \\ \Psi(x^{(i-1)}_m, x^{(1)}_k) &= \left\langle f(W_{query}, x^{(i-1)}_m), f(W_{key}, x^{(1)}_k) \right\rangle \end{aligned} \tag{4}$$

where $a^{(i)}_{m,k}$ is an element of $A^{(i)} \in \mathbb{R}^{n_1 \times n_{i-1}}$ indicating the correlation coefficient between the m-th feature $x^{(i-1)}_m \in X^{(i-1)}$ and the k-th feature $x^{(1)}_k \in X^{(1)}$. We use an attention function $\Psi(*,*)$ to calculate the correlation between $x^{(i-1)}_m$ and $x^{(1)}_k$, where the calculation function is the inner product of the input vectors. A lightweight convolution net $f(*,*)$ is employed, which first aggregates the inputs $x^{(q)}_p \in \mathbb{R}^{T \times 1 \times d}$ on temporal dimension by a $T \times 1 \times 1$ conv-layer, and then projects the aggregated feature $\hat{x} \in \mathbb{R}^d$ into a low-rank space $\mathbb{R}^z$, where z is the dimension of the low-rank space. Additionally, W_{query} and $W_{key} \in \mathbb{R}^{d \times z}$ are the learnable parameters for extracting query and key information from the temporal aggregations, respectively. To learn the effectiveness of the i-th rank features $X^{(i)}$, we update the representation of the crossing feature $x^{(i)}_{m,k} \in X^{(i)}$ in d-dimensional feature space with residual connections guided by attention coefficients:

$$\begin{aligned} v^{(i)}_{m,k} &= f(W_{value}, x^{(i)}_{m,k}) \\ \widetilde{x}^{(i)}_{m,k} &= g(a^{(i)}_{m,k} \cdot v^{(i)}_{m,k}) + x^{(i)}_{m,k} \end{aligned} \tag{5}$$

Here, we adopt the Leaky Rectified Linear Units (LReLU) (Zhang, Pan, Sun, & Tang, 2018) as the non-linear activation function $g(*)$, with a negative slope of 0.1. Additionally, $f(*,*)$ represents a convolution layer with a filter kernel size of 1×1, having d input channels and d output channels. The learnable parameters of the convolution layers are denoted as $W_{value} \in \mathbb{R}^{d \times z}$.

As the rank of the features increases, the number of corresponding combinations grows exponentially. Enumerating all possible high-rank features would lead to an exponential increase in memory and computation requirements. In fact, only a few high-rank features are effective for the target task. To address this, we utilize a convolutional neural network to construct a Principal Feature Selection (PFS) module, as shown in Fig. 2. This module employs a point-wise convolutional neural network, with lasso regularization (Zhao & Yu, 2006) applied to the filters during the training stage, to explicitly extract meaningful features and reduce computational costs. The implementation of the PFS module is expressed as follows:

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

where $L(*,*)$ is the loss function for the target label y. A convolution neural network $f(*,*)$ is used in PFS module for explicitly extracting meaningful features, where the number of input channels is $c_i = n_1 \times n_{i-1}$, and the number of output channels is a hyper-parameter $c_o = n_i$. The learnable parameters of convolution kernels $W_{:,k}$ are imposed with lasso, which makes most of the elements in $W_{:,k}$ approach to zeros after model training. This allows us to identify which feature combinations in $X^{(i)}$ are useful for the target task by locating the non-zero values of W. Representations of meaningful feature combinations $\{X^{(2)}, ..., X^{(l)}\}$ and their conditioned on attention weights $\{A^{(2)}, ..., A^{(l)}\}$ are then generated using a battery of stacked $l-1$ feature crossing modules. Finally, $N = \sum_{i=1}^{l} n_i$ feature combinations with different ranks are collected by using the concatenation operator $\odot$.

$$\widetilde{X} = X^{(1)} \odot X^{(2)} \odot \cdots \odot X^{(l)} \tag{7}$$

As a result, we obtain a mixed feature representation $\widetilde{X} \in \mathbb{R}^{T \times N \times d}$ about the target task, wherein each feature has explicit combination semantics. This representation has l feature ranks.

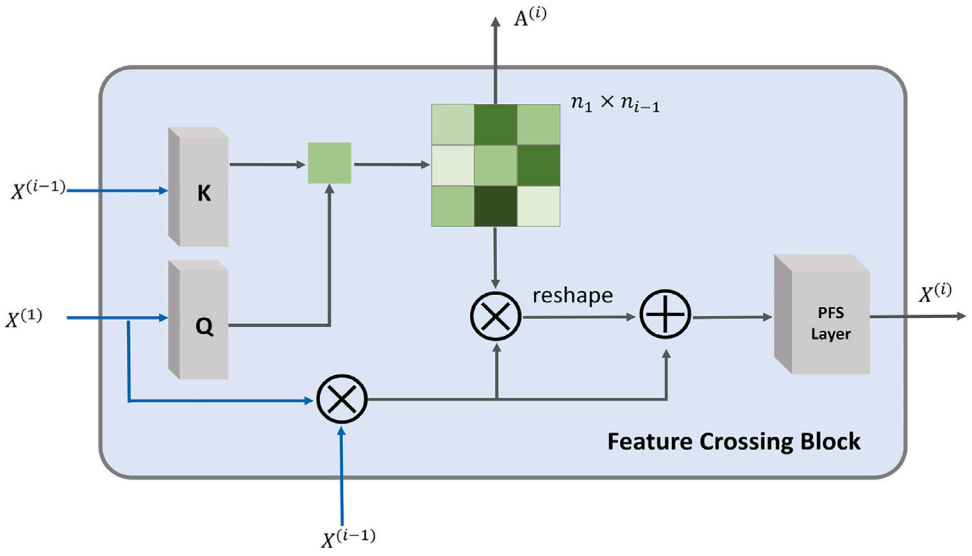

Fig. 2. Feature crossing block, where $\oplus$ indicates the element wise addition and $\otimes$ is a cartesian product. Grey cubes represent the convolutional networks.

3.4. Dual attention module

Once the combination features are generated in the same low-dimensional space, we further employ a feature attention module to learn the influence of different features on the final enterprise credit rating. To adaptively calculate the attention scores of each feature at different time points, which indicate the correlations between the result of enterprise credit rating and features, we assign each feature $\widetilde{X}_{t,i,:} \in \widetilde{\mathbf{X}}$ an attention score $s_{t,i}$ through a convolution layer-based projector.

$$\begin{aligned} &A = f(W, \widetilde{X}) \\ &a_{t,i} = \frac{exp(a_{t,i})}{\sum_j^T \sum_k^N exp(a_{j,k})}, \quad a_{j,k}, a_{t,i} \in S \end{aligned} \tag{8}$$

Let $A \in \mathbb{R}^{T \times N}$ be the matrix of attention scores, where $f(*,*)$ is a convolution neural network with a filter kernel size of $1 \times d$, input and output channels equal to the sequence length T, and weights $W \in \mathbb{R}^{T \times T \times 1 \times d}$. $f(*,*)$ calculates the attention scores through convolution and reshape operations.

To preserve the information of previously learned combination features, we add a residual connection to the end of this module. Formally, the output of this module can be expressed as follows:

$$\widetilde{X}_{t,i,:} = g(s_{t,i} \cdot \widetilde{X}_{t,i,:}) + \widetilde{X}_{t,i,:} \tag{9}$$

where $g(*)$ represents the rectified linear units (ReLU), which can add the non-linearity into the proposed model and select the meaningful features.

3.5. Credit rating and model training

Enterprise credit is a reflection of the operational status of the enterprise, and is influenced by both long-term and short-term operations. To balance efficiency and performance, we use Gated Recurrent Units (GRUs) (Chung, Gülçehre, Cho, & Bengio, 2014) to model the long-term and short-term dependencies of enterprise credit on the input time series. GRUs are advantageous over Recurrent Neural Networks (RNNs) as they overcome the vanishing gradients problem, and are faster than Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997).

Specifically, for the given company i, we concatenate the feature vectors $\{\widetilde{X}^{(i)}\}_{i=1}^{l}$ outputted by the dual attention module to the same time point inputs of GRU for the final credit rating prediction.

$$e_t = \widetilde{X}_{t,1,:} \odot \cdots \odot \widetilde{X}_{t,N,:} \tag{10}$$

Let e_t be the t-th input vector of GRU, the formulations of GRU can be expressed as follows:

$$\begin{aligned} u_t &= \sigma(W_u \cdot [h_{t-1}, e_t]) \\ z_t &= \sigma(W_z \cdot [h_{t-1}, e_t]) \\ \widetilde{h}_t &= tanh(W_{\widetilde{h}} \cdot [u_t * h_{t-1}, e_t]) \\ h_t &= z_t * \widetilde{h}_t + (1 - z_t) * h_{t-1} \end{aligned} \tag{11}$$

Here, the GRU are realized with the sigmoid activation function $\sigma(*)$, the tanh activation function $tanh(*)$, the learnable feature projected matrices W_u, W_z and $W_{\widetilde{h}}$, where the $*$ in $\sigma(*)$ and $tanh(*)$ represents the element-wise product, h_t is the t-th hidden states of GRU. We utilize the last hidden state h_T of GRU as the low-dimensional feature representation of the company and predict the company's credit rating as follows:

$$\mathbf{r} = softmax(f_{FC}(W_{FC}, h_T)) \tag{12}$$

A fully connected layer $f_{FC}(*,*)$ is used to project the final credit rating representation of a company, h_T, into a credit ratings distribution vector $\mathbf{r}$. This projection is conducted by the matrix $W_{FC} \in \mathbb{R}^{c \times n}$, where n is the dimension of h_T and c is the number of ratings. The final predicted rating of the company $\widetilde{y} = index(max(\mathbf{r}))$ is the index of the maximal element in the credit ratings distribution vector $\mathbf{r}$.

To effectively train the proposed model, we leverage a loss function L_q (Zhang & Sabuncu, 2018) which is situated between a regression loss (e.g., MAE loss) and a classification loss (e.g., cross entropy loss) to supervise the learning process. This is done for two reasons: 1) credit rating tasks can be realized through either classification or regression; and 2) MAE loss usually has good generalization but lacks fitting ability, whereas cross entropy loss is the opposite. Our training process is formally expressed as follows:

$$\begin{aligned} &\arg\min_{\mathbb{W}} L_q(y_j, \widetilde{y}_j) + \sum_{i=2}^{l} \sum_{k=1}^{c_o} \|W_{:,k}^{(i)}\|_{l1} \\ &L_q(y_j, \widetilde{y}_j) = \frac{1 - (y_j \cdot log\ \widetilde{y}_j)^q}{q} \end{aligned} \tag{13}$$

where $\mathbb{W}$ denote the learnable parameters of the proposed model, which are updated by minimizing the total loss using gradient descent. For $W_{:,k}^{(i)} \in \mathbb{W}$, this represents the learnable parameters of the PFS

module in the i-th feature crossing module (Section 3.3). $\widetilde{y}_j$ is the j-th element in the predictive vector $\widetilde{y}$, and $q \in (0, 1]$ is a hyper-parameter that allows for tuning of the supervised learning between classification and regression. The loss function is equivalent to a cross entropy loss when $q \to 0$, and to a MAE loss when $q = 0$. In this work, we set q to 0.5 as a hyper-parameter.

4. Explainable enterprise rating

Through the interpreter shown in Fig. 1, we attempt to generate two types of explanations: global explanations for the rating model and individual explanations for each rating result. These explanations can be rationalized from various perspectives, such as the training data, model reliability, and individual samples.

Owing to the imposition of lasso regularization on the parameters of PFS modules, most of the weights about feature combinations tend to zero after model training. Thus, given a dataset and a trained model, we can identify the meaningful feature combinations for the target task by indicating the non-zero weights. Here, we assume the useful combination patterns $\Omega = \{S^{(l)} | l \in [1, 2, \dots, L]\}$, where $S^{(l)} = \{s_j^{(l)} | 1 \le j \le n_l, s_j^{(l)} = index(max(W_{:,j}^{(l)}))\}$ is the meaningful feature combinations of the lth rank, and $index(max(W_{:,j}))$ returns the jth meanful feature combination from the set indicated by the non-zero weight. By examining the global explanations, i.e., Ω, financial experts can verify 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.

Moreover, personalized explanations for each prediction can be obtained by mining the attention cues of the feature crossing and dual feature attention modules. Specifically, given an attention score matrix $S \in \mathbb{R}^{t \times n}$ generated from the dual feature attention module for a company, personalized explanations for the prediction can be generated by followed recursive tracking algorithm.

$$\begin{aligned} &\{e_1, e_2, \dots, e_k\} = \text{topk}(E) \\ &E = max(\mathbf{r}) \cdot S \\ &\delta(e_i) = E_{K(e_i)} \cdot \prod_l^L \prod_{s \in \pi(S^{(l)}, e_i)} A_{I(s)}^{(l)} \cdot max(W_{:,J(s)}^{(l)}) \end{aligned} \tag{14}$$

As the Eq. (14) shown, given the set of significant features of rating prediction $\{e_1, e_2, .., e_k\}$, their significance weights and indexes can be obtained from the matrix $E \in \mathbb{R}^{t \times n}$ by the $E_{K(e_i)}$ and the $topk(E)$, respectively, where $topk(E)$ returns the indexes of the top K maximal elements in matrix E. The matrix E can be treated as the weights of the feature combinations at different time points. The $\mathbf{r}$ is a distribution vector of ratings of the company, which can be treated as the probabilities on different rating classes, the confidence coefficient of rating prediction then can be determined by $max(\mathbf{r})$. To further parse these significant features' components, we can recursively track their process of generation by mining the global explanations Ω. The significance of feature e_i for a given prediction can be scored with $\delta(e_i)$, where $\pi(S^{(l)}, e_i)$ is a subset of the l-th rank meaningful feature combinations associated with e_i. The index functions $I(s)$, $J(s)$ and $K(e_i)$ are used to return the location of elements in the matrix $A^{(l)}$, the associated column of $W^{(l)}$ and E, respectively. Through the above calculation, the interpreter allows to provide different explanations for different input and prediction pairs.

5. Experiments

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

- What is the performance of the proposed model on the enterprise credit rating problem? Is it effective for large-scale, sparse, high-dimensional, and multi-type data sets?
- Is the proposed model and its outputs explainable? How can we generate and understand the explanations using our proposed methods?
- What impact do different model configurations have on predictive performance?

5.1. Experimental setup

5.1.1. Datasets

We evaluate the proposed approaches using three real-world datasets: the CH-Stocks, CH-Rating, and US-Stocks[1] datasets.

CH-Stocks collected the historical data of 7968 Chinese listed companies from multiple data sources, which includes 23 first-rank financial features(shown in Appendix) from their IPO to the third fiscal quarter of 2019. Investment analysts (Bellovwy & Don, 2005) have suggested that a company's revenue situation may be indicative of its credit rating. For example, the revenue situation of Tech startups may be an important indicator for investors to evaluate the company's value and credit. Therefore, in this experiment, we used the revenue situation as an indirect proxy of company credit rating. We randomly split the dataset into a training set (70%) and a testing set (30%), and classified whether the revenue increased in the next fiscal quarter to indirectly predict the credit rating of the company. The testing data contained 2451 Chinese listed companies, with 1493 being positive samples.

CH-Rating contains historical credit rating data of 1602 Chinese companies, collected from "Wind Terminal". For each company, we can access its financial, solvency, and operational features, as well as its credit ratings. To predict the company's credit rating for the following fiscal quarter, we randomly split the dataset into a training set (70%) and a testing set (30%) from the company dimension. Since the dataset is composed of large, publicly traded companies, most of them have favorable ratings. To address the uneven sample distribution problem of this dataset, we rearranged the original 19 ratings into three: good, normal, and bad. Specifically, the top 5 ratings are treated as good, the following 4 ratings as normal, and the rest as bad.

US-Stocks contains over 200 financial indicators for all stocks of in the US stock market from 2014 to 2018, which aimed to understand whether it is possible to predict the future performance of a company by looking at the financial information released in financial reports. Empirical observations (Dichev & Piotroski, 2001) suggest that there is a correlation between a company's credit rating and its stock price movement, i.e., when the company's credit rating changes, its stock price often also fluctuates. Thus, the stock price movement can be treated as a proxy to reflect the credit situation of the company in reverse. In this experiment, a company whose average stock price increases in the next year was classified as having a positive rating, and one whose stock price decreases was classified as having a negative rating. The dataset was randomly split into a training set (70%) and a testing set (30%), and the credit rating of the company was predicted by classifying whether its stock price increased in the subsequent year.

The list of listed companies changes over time, so the data time horizon varies for each company. As Fig. 3 shows, most companies in the experimental datasets have a continuous record of around five years. Therefore, we mainly trained and tested the proposed model with five consecutive years on both the CH-Stocks and US-Stocks datasets.

5.1.2. Competing methods

We compared the proposed model to both deep learning-based models (AutoInt (Song et al., 2019) and IFR-CNN (Hosaka, 2019)) and conventional shallow models (Decision Tree (DT) (Xia et al., 2017), Support Vector Machines (SVM) Chang and Lin (2011), Factorization Machine (FM) (Rendle, 2012), Gradient Boosted Decision Trees

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

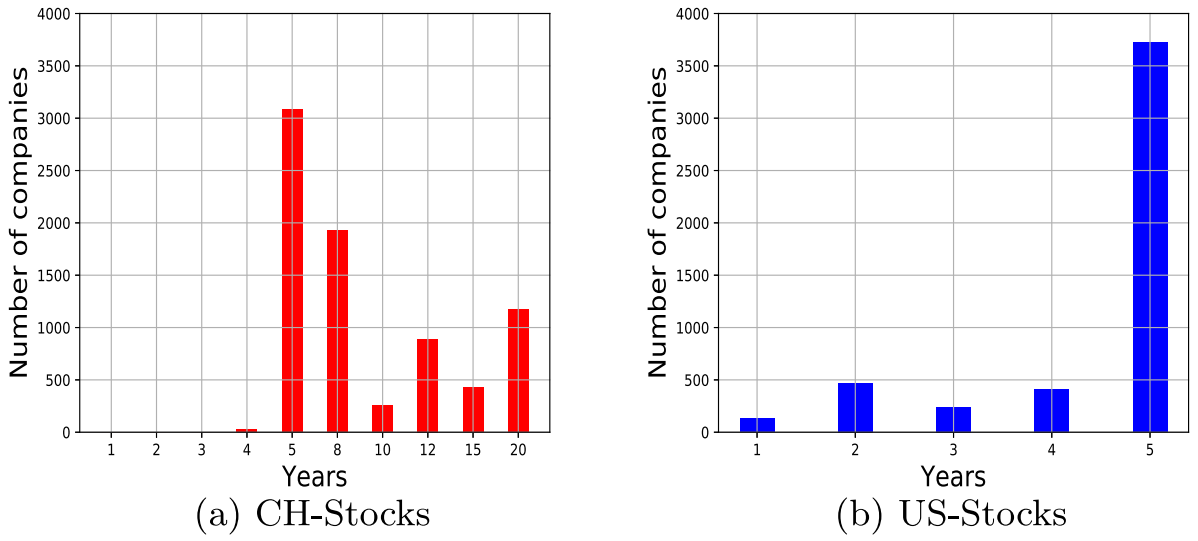

Fig. 3. Statistical distribution of the number of companies w.r.t. the time span.

(GBDT) (Ke, Meng, Finley, et al., 2017), Z-Score (Altman, 2013), and Logistics Regression (LR) (Sohn, Kim, & Yoon, 2016)). Except for the Z-Score, all raw features of a company over all time periods were concatenated as the input for the shallow models. While AutoInt and IFR-CNN were originally designed for click-through rate prediction and bankruptcy prediction, respectively, we transform them to perform enterprise credit rating prediction.

Z-Score is a classic enterprise credit rating model which considers a linear weighted sum of five financial indicators of public companies as the company's credit score, i.e., working capital/total assets (x_1), retained earnings/total assets (x_2), earnings before interest and taxes/total assets (x_3), market value of equity/book value of total liabilities (x_4), and sales/total assets (x_5). The score $Z = 1.2x_1 + 1.4x_2 + 3.3x_3 + 0.6x_4 + 1.0x_5$ is then used to rate the company by setting experience threshold scores: if $Z < 1.81$, the company is deemed a failure with a bad credit, and if $Z > 2.67$, the company has a good credit.

LR is a popular choice for credit assessment, similar to the Z-Score model as it is a linear weighted model. However, the weights in LR are learned from the training data. We used scikit-learn[2] with an L_1 penalty and the "liblinear" solver to realize the training algorithm. The tolerance for the stopping criteria was set to 0.0001, while other hyper-parameters were kept at the default settings.

SVM[3] is a classic and effective kernel-based model, especially in high-dimensional spaces, even when the number of dimensions is greater than the number of samples. This makes it a popular choice in the credit assessment field. In this experiment, the training algorithm was realized using "libsvm", with "nu-SVC" and "rbf" kernel type selected, and all other settings kept to their default configurations.

DT was a widely used algorithm for classification in finance. In this experiment, the maximum tree depth set to 5, the training algorithm was implemented using scikit-learn, and the criterion for building the tree was evaluated using Gini impurity.

FM is a general framework which uses factorization techniques to model second-rank feature interactions and can provide high prediction accuracy. In this experiment, we used libFM[4] and the adaptive SGD learning method with a learning rate 0.1, 500 iterations , and default configurations for other settings to model 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, which cab produce a prediction model in the form of an ensemble of weak decision trees. In this experiment, we realized the training algorithm using scikit-learn, setting the number of estimators to 100 and the number of random states to 10. All other hyper-parameters were left at their default values.

AutoInt[5] was employed to automatically learn high-rank feature interactions using multi-head self-attention, which maps both the numerical and categorical features into the same low-dimensional space. In this experiment, we set the number of heads and blocks to be 2 and 3, respectively, with a block shape of $[64, 64, 64]$. To ensure a fair comparison, we employed AutoInt to obtain the feature representation of a sample, and then used our dual feature attention module to generate prediction.

IFR-CNN transforms the bankruptcy prediction to a 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 (Szegedy, Liu, Jia, et al., 2015) realized by torchvision[6] is trained and tested by using those 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 data to make predictions.

5.2. Evaluation criteria

In binary classification problems, four types of predictions can be obtained: true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). To evaluate the predictive performance of the compared models, we defined a confusion matrix as shown in Table 1. We then used several indicators, such as accuracy, type I and II errors, and area under the ROC curve (AUC), to obtain a comprehensive evaluation. Based on the confusion matrix defined in Table 1, the accuracy, type I and II errors can be defined as follows:

$$\begin{aligned} Acc &= \frac{TP+TN}{TN+FP+FN+TP} \\ Err_1 &= \frac{FP}{TN+FP} \\ Err_2 &= \frac{FN}{FN+TP} \end{aligned} \tag{15}$$

[2] scikit-learn.org/stable.
[3] www.csie.ntu.edu.tw/cjlin/libsvm.
[4] www.libfm.org.
[5] github.com/shichence/AutoInt.
[6] github.com/pytorch/vision.

Table 1
Confusion matrix of classifiers in this experiment. $y = 1$ indicates that the model gives the company a positive outlook, $y = 0$ represents a 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)

Table 2
Performance of enterprise credit rating prediction of different models. In this experiment, our model,i.e., DeepCross, was trained to be a third-rank model on CH-Stocks, US-Stocks and CH-Rating which the output dimension of the embedding layer is 64. The output dimensions of PFS modules 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 with consecutive time, i.e., US-Stocks is 5 years data, CH-Stocks is 15 fiscal quarters data and CH-Rating is 3 fiscal quarters data.

Model class	Models	CH-Stocks			
		Acc	AUC	Err_1	Err_2
First-rank	Z-Score	0.739	–	0.192	0.691
	LR	0.946	0.955	0.033	0.038
	SVM	0.793	0.846	0.222	0.093
High-rank	FM	0.824	–	0.184	0.137
	GBDT	0.953	0.971	0.040	0.052
	DT	0.931	0.962	0.055	0.085
Deep-rank	AutoInt	0.961	0.974	0.026	0.042
	IFR-CNN	0.889	0.943	0.079	0.196
	DeepCross	**0.980**	**0.996**	**0.017**	**0.028**
Model class	Models	US-Stocks			
		Acc	AUC	Err_1	Err_2
First-rank	Z-Score	0.559	–	0.279	0.587
	LR	0.726	0.781	0.268	0.290
	SVM	0.727	0.779	0.288	0.219
High-rank	FM	0.724	–	0.263	0.303
	GBDT	0.750	0.828	0.255	0.228
	DT	0.734	0.804	0.286	**0.189**
Deep-rank	AutoInt	0.750	0.787	0.246	0.265
	IFR-CNN	0.713	0.701	0.303	0.366
	DeepCross	**0.772**	**0.834**	**0.230**	0.253
Model class	Models	CH-Rating			
		Acc	AUC	Err_1	Err_2
First-rank	Z-Score	0.570	–	–	–
	LR	0.978	0.799	–	–
	SVM	0.962	0.858	–	–
High-rank	FM	0.952	–	–	–
	GBDT	0.976	0.818	–	–
	DT	0.980	0.659	–	–
Deep-rank	AutoInt	0.971	0.873	–	–
	IFR-CNN	0.975	0.819	–	–
	DeepCross	**0.980**	**0.913**	–	–

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. These metrics provide insight into how well labels are predicted by a given binary classification model.

To further evaluate the quality of models in this experiment, the AUC (Area Under the Curve) of the ROC (Receiver Operating Characteristic) is utilized to avoid the evaluation error caused by sample imbalance. The ROC is a comprehensive indicator that reflects the continuous variables of True Positive Rate and False Positive Rate. The AUC value is between 0.5 and 1, with a higher value being indicative of better performance.

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. From the experimental results, we observe that (1) machine learning based linear models, such as LR and SVM, significantly outperform the statistics analysis-based Z-Score model as they can adaptively fit the given dataset, which may be more suitable for the company credit rating task on large-scale data. (2) Surprisingly, the performance of Z-Score model on CH-Rating dataset was worse than on CH-stocks. This may be due to the fact that the modern enterprise rating system considers more information besides the operation and financial situation of the enterprise, while the situation of the enterprise revenue is a more direct reflection of a company's operating and financial situation. Thus, the situation of the enterprise revenue probably can be treated as an indirect proxy variable to study the financial credit situation of the enterprise. (3) GBDT and DT, which explore high-rank feature engineering, consistently outperform the first-rank approaches by a large margin on all datasets, indicating that using only first-rank features may be insufficient in company credit rating prediction. (4) Deep learning models such as DeepCross, AutoInt, and IFR-CNN, which benefit from the feature engineering capabilities and the attention mechanism, typically achieve better performance than other models. (5) The proposed model, DeepCross, obtains the best performance, suggesting that using feature crossing modules to explore deeper-rank feature interactions is essential. Note that the proposed model shares the same structures on the embedding layer and credit rating layer as AutoInt. (6) The deep learning-based model IFR-CNN does 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, as the company credit changes with time and it may be a sequential process.

In conclusion, the proposed model outperformed all compared models. Specifically, compared to the most competitive baseline, AutoInt, the proposed model was able to explore deeper-rank feature interactions with similar resource consumption and was more efficient during online inference. This was achieved through the stacked feature crossing modules, which first perform explicit feature crossing via vectorized Cartesian product, and then perform PFS 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 the rank of the proposed model was increased, because higher rank feature crossing means that more feature combinations are used for prediction. However, the results obtained on two datasets differed somewhat. When the range of feature rank was over 4, the performance of the proposed model on the CH-Storcks dataset began to decrease. The reason for this reduced performance is probably that the number of first-rank features in CH-Storcks dataset is small, and there may contain too many invalid feature combinations over the fourth-rank. As a result, the number of training samples may be relatively small compared to the feature dimension, leading to the proposed model exhibiting over-fitting.

5.3.3. Influence of different time spans

We predict the credit ratings of companies by sequential modeling. To investigate the feature of our model, we changed setting of the parameter t, which is the time spans of data used to training and testing. As shown in Fig. 5, the accuracy and AUC firstly increased as we increased the time span on testing phase, since credit rating of a company was changing over time. However, when the time span

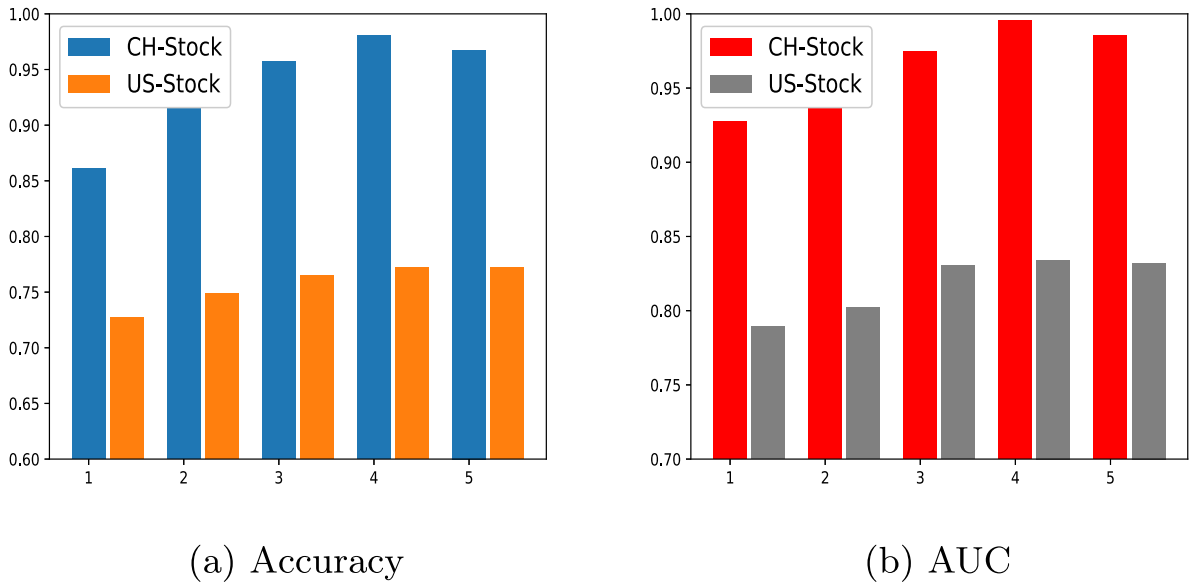

Fig. 4. Performance w.r.t. the rank of the model.

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. 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_{16}	0.3852	f_{11}, f_{16}	0.1995	f_6, f_{16}, f_{18}	0.0177	f_4, f_4, f_7, f_5	0.0500
2	f_{14}	0.0878	f_{16}, f_{18}	0.1599	f_0, f_3, f_{20}	0.0177	f_0, f_{12}, f_{16}, f_{19}	0.0499
3	f_{13}	0.0809	f_6, f_{16}	0.1594	f_{16}, f_{16}, f_{19}	0.0177	f_3, f_3, f_9, f_{14}	0.0250
4	f_{12}	0.0667	f_3, f_{14}	0.1198	f_9, f_{12}, f_{13}	0.0177	f_{11}, f_{14}, f_{15}, f_{17}	0.0250
5	f_{10}	0.0593	f_{12}, f_{16}	0.1198	f_8, f_0, f_{20}	0.0177	f_6, f_{10}, f_{12}, f_{22}	0.0250

exceeded a certain range, the performance of our model began to decrease on CH-Stocks. In contrast, with any time spans settings, the proposed model achieved convergence on the training datasets, and the convergence speed was accelerated as the time span was increased. This phenomenon is likely due to the significant reduction of training samples in CH-Stocks, which causes the trained model to suffer from over-fitting, and the indirect proxy task we adopted on CH-stocks relies heavily on the short term data. It 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 if we want to obtain more accurate rating prediction by using longer period's data.

5.4. Explainable enterprise credit rating

A good enterprise credit rating system should provide accurate evaluations with good explainability of the model. Here, we describe the proposed model, which is able to provide accurate evaluations with good explainability and explainable modeling process. Benefiting from the dual attention modules and the feature crossing modules, the model not only can predict the credit rating of a given company, but also can generate static explanations for the modeling process.

By utilizing the PFS modules with lasso regularizations in the feature crossing modules, we can summarize the most important feature combinations in different ranks of feature sets for a given dataset. As Table 3 shows, we parsed the useful combination patterns and their weights from PFS modules in a backtracking way (Section 4). This static explanations allow human experts in the field of enterprise credit rating to further investigate the trained model to determine if there is any bias caused by the training datasets and model configurations. For example, the combination of operation cycle (i.e., f_{11}) and capital return (i.e., f_{16}) in the second-rank feature combinations, may be indicative of a company's debt paying ability, which is consistent with common accounting principles. Additionally, Table 3 also shows that as the rank increases, the weights of the feature combinations tend to be zero. This suggests that high-rank features typically include a lot of redundancy and noise, thus emphasizing the need for feature selection in the proposed feature crossing module.

In the field of enterprise credit rating tasks, users are interested in understanding the correlations between the features of a given sample and the specific credit rating results. To this end, we further provide a way to assess the credit rating of a given company by visualizing the correlations between the most important feature combinations and time points, as shown in Fig. 6. From the visualized results, we can observe the following: (1) The features of the time points closer to the rating time point are more influential to the result than the features farther away from the rating time point, e.g., the features on t_5 are the most effective for both positive and negative samples. This is likely due to the indirect proxy task, i.e., revenue situation prediction, relying heavily on short-term information, which is suitable for monitoring the short-term credit changes of a company. (2) The results of different samples are affected by different features, verifying that our model can output a personalized interpretation. (3) Some features are important to both positive and negative samples, e.g., the net profit cut growth rate (i.e., f_{13}) can discriminate both.

In summary, the proposed model is highly explainable, providing both global and personalized explanations. These explanations can help financial experts evaluate the reliability of the trained models, and enable users to comprehend the rating results.

6. Conclusion

In this paper, we propose a novel self-attention-based feature crossing network for predicting enterprise credit ratings with high accuracy

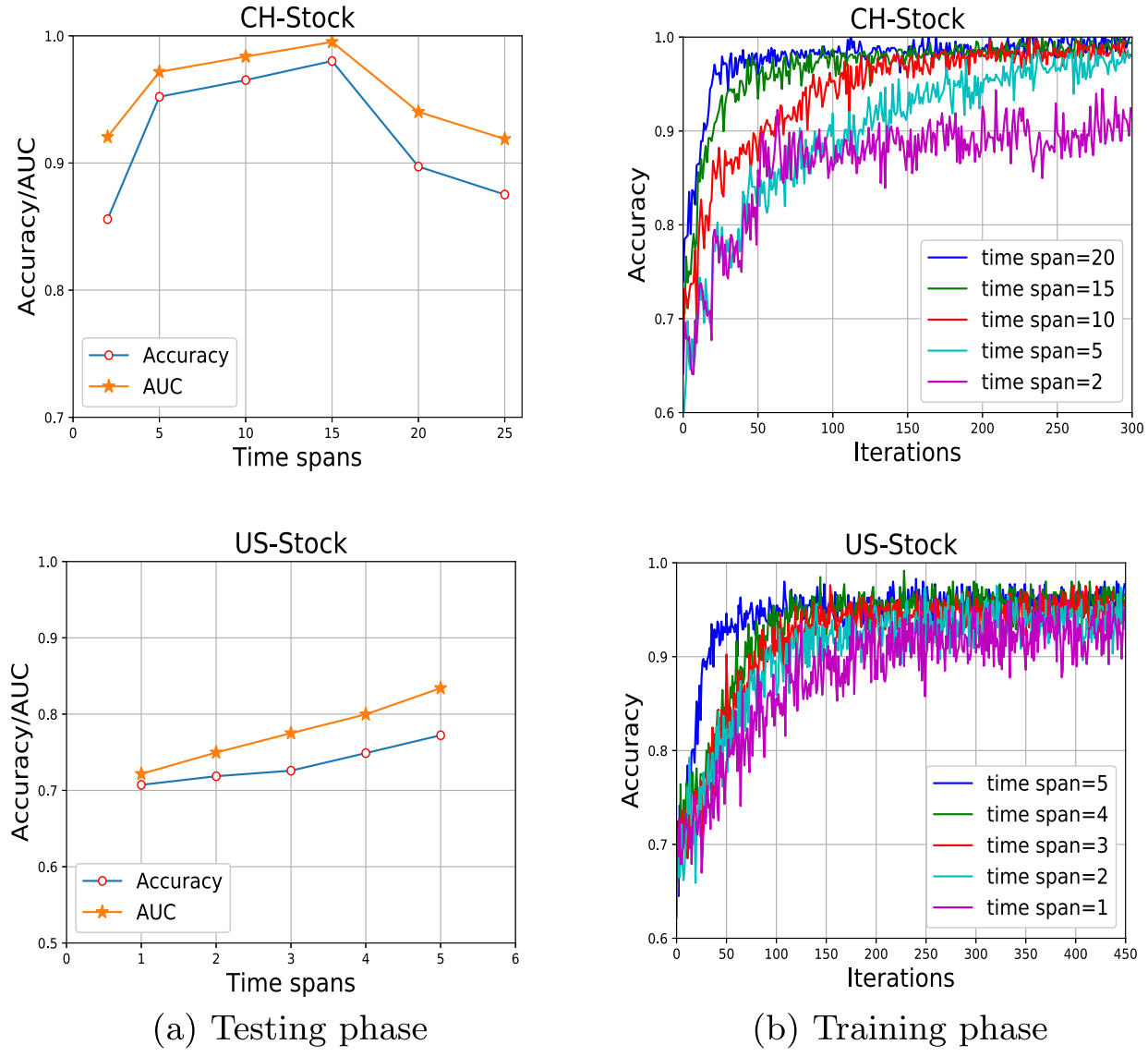

Fig. 5. Influence of different time spans in training and testing phases. In training phase, we adopted the method of stochastic gradient descent with 0.001 learning rate to train our models.

and good explainability. Our model maps the original sparse, high-dimensional features into low-dimensional spaces and explicitly models the interactions of the high-rank features. We mine and visualize the Cartesian product of attention maps of the proposed model to provide both static and personalized explanations for a given sample and credit rating pair. The experimental results confirm that our model has higher prediction accuracy than traditional enterprise credit rating models and can explain both the prediction results and the model's training process.

However, the proposed model is still data-driven, and may suffer from over-fitting when the number of training samples is small. To mitigate this issue, we plan to extend the model to support non-financial information, such as news about companies and their propagation on social media. This would allow us to extend expand the number of samples, and include more non-listed companies in our experimental datasets. Furthermore, we intend to obtain a more accurate company representation by considering graph information, such as news diffusion on social media, with graph structure attention. Finally, we plan to conduct experiments to explore the relation between enterprise credit rating and the indirect proxies used in this work, i.e., company revenue and stock price.

CRediT authorship contribution statement

Weiyu Guo: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Zhijiang Yang:** Data curation, Software, Visualization. **Shu Wu:** Supervision. **Xiuli Wang:** Resources. **Fu Chen:** Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

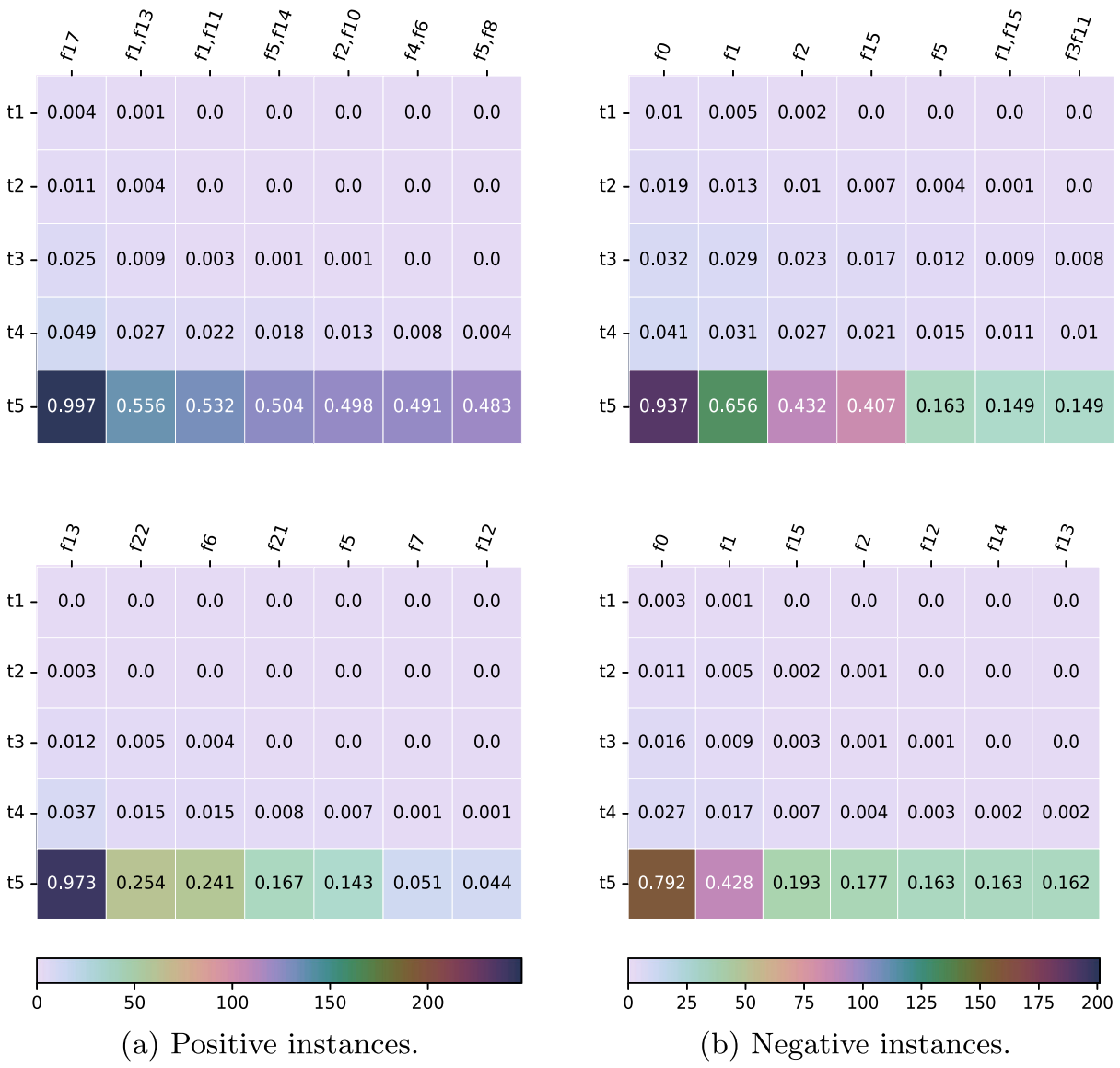

Fig. 6. Heatmap examples of input feature weights w.r.t time points. The weights were first normalized to $[0, 1]$, and then mapped to the color space $[0, 255]$. The intensity of the color blocks indicates the importance of the corresponding features. We show the seven most important features (the sum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

Data availability

Data will be made available on request.

Acknowledgments

This work is jointly supported by the National Natural Science Foundation of China (62106290) and the Program for Innovation Research in Central University of Finance and Economics PR China.

Appendix

The raw features of the listed Chinese companies used in our experiments are listed in Table A.4.

Table A.4
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	Debt assets ratio
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}	Net profit ratio	f_{15}	Gross income ratio
f_{16}	Capital return	f_{17}	Return on equity
f_{18}	Inventory turning rate	f_{19}	Current ratio
f_{20}	Quick ratio	f_{21}	Conservative quick ratio
f_{22}	Debt equity ratio	–	–

References

Altman, I. E. (2013). Predicting financial distress of companies: Revisiting the z-score and zeta models. In *Handbook of research methods and applications in empirical finance* (pp. 428–456).

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *ICLR*.

Bellovwy, J. L., & Don, E. (2005). Earnings quality: It's time to measure and report. *The CPA Journal*, *75*, 32–37.

Beutel, A., Covington, P., Jain, S., et al. (2018). Latent cross: Making use of context in recurrent recommender systems. In *WSDM*.

Bolton, C., et al. (2010). *Logistic regression and its application in credit scoring* (Ph.D. thesis), University of Pretoria.

Cao, Y., Xu, J., Lin, S., et al. (2019). Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *ICCV*.

Chang, C., & Lin, C. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*, 27:1–27:27.

Chung, J., Gülçehre, Ç., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Dichev, I. D., & Piotroski, J. D. (2001). The long-run stock returns following bond ratings changes. *The Journal of Finance*, *56*, 173–203.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Golbayani, P., Wang, D., & Florescu, I. (2020). Application of deep neural networks to assess corporate credit rating. arXiv preprint arXiv:2003.02334.

Guo, W., Cao, B., & Li, Z. (2020). Explainable enterprise rating using attention based convolutional neural network. In *WISA*.

Guo, W., Wu, S., Wang, L., & Tan, T. (2016). Personalized ranking with pairwise factorization machines. *Neurocomputing*, *214*, 191–200.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780.

Hosaka, T. (2019). Bankruptcy prediction using imaged financial ratios and convolutional neural networks. *Expert Systems with Applications*, *117*, 287–299.

Ke, G., Meng, Q., Finley, T., et al. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *NIPS*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

Lian, J., Zhou, X., Zhang, F., et al. (2018). Xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *SIGKDD*.

Luo, Y., Wang, M., Zhou, H., et al. (2019). Autocross: Automatic feature crossing for tabular data in real-world applications. In *SIGKDD*.

Matin, R., Hansen, C., Hansen, C., & Mlgaard, P. (2019). Predicting distresses using deep learning of text segments in annual reports. *Expert Systems with Applications*, *132*, 199–208.

Mccrae, R. R., & John, O. P. (1992). An introduction to the five-factor model and its applications. *Journal of Personality*, *60*, 175–215.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In *ICLR*.

Raganato, A., Scherrer, Y., & Tiedemann, J. (2020). Fixed encoder self-attention patterns in transformer-based machine translation. In *EMNLP*.

Rendle, S. (2010). Factorization machines. In *ICDM*.

Rendle, S. (2012). Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, *3*, 57:1–57:22.

Rendle, S., Gantner, Z., Freudenthaler, C., & Schmidt-Thieme, L. (2011). Fast context-aware recommendations with factorization machines. In *SIGIR*.

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *EMNLP*.

Seo, S., Huang, J., Yang, H., & Liu, Y. (2017). Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *RecSys*.

Shaw, P., Uszkoreit, J., & Vaswani, A. (2018). Self-attention with relative position representations. In *NAACL-HLT*.

Sohn, S. Y., Kim, D. H., & Yoon, J. H. (2016). Technology credit scoring model with fuzzy logistic regression. *Applied Soft Computing*, *43*, 150–158.

Song, Q., Cheng, D., Zhou, H., et al. (2020). Towards automated neural interaction discovery for click-through rate prediction. In *SIGKDD*.

Song, W., Shi, C., Xiao, Z., et al. (2019). Autoint: Automatic feature interaction learning via self-attentive neural networks. In *CIKM*.

Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015). End-to-end memory networks. In *NIPS*.

Szegedy, C., Liu, W., Jia, Y., et al. (2015). Going deeper with convolutions. In *CVPR*.

Tao, Z., Wang, X., He, X., et al. (2020). Hoafm: A high-order attentive factorization machine for ctr prediction. *Information Processing and Management*, *57*, Article 102076.

Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. In *NIPS*.

Wang, R., Fu, B., Fu, G., & Wang, M. (2017). Deep and cross network for ad click predictions. In *ADKDD*.

Wang, X., He, X., Feng, F., et al. (2018). TEM: tree-enhanced embedding model for explainable recommendation. In *WWW*.

Wu, C., Wu, F., Ge, S., et al. (2019). Neural news recommendation with multi-head self-attention. In *EMNLP-IJCNLP*.

Xia, Y., Liu, C., Li, Y., & Liu, N. (2017). A boosted decision tree approach using bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, *78*, 225–241.

Zhang, Y., Pan, C., Sun, J., & Tang, C. (2018). Multiple sclerosis identification by convolutional neural network with dropout and parametric relu. *Journal of Computer Science*, *28*, 1–10.

Zhang, Z., & Sabuncu, M. R. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*.

Zhao, P., & Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine Learning Research*, *7*, 2541–2563.

Zhu, J., Shan, Y., Mao, J. C., et al. (2017). Deep embedding forest: Forest-based serving with deep embedding features. In *SIGKDD*.