Deep Attentional Factorization Machines Learning Approach for Driving Safety Risk Prediction

Jun Zhang\textsuperscript{1,2,3,*}, Zhongcheng Wu\textsuperscript{1,2}, Fang Li\textsuperscript{1}, Wenjing Li\textsuperscript{1,2}, Tingting Ren\textsuperscript{1}, Wei Li\textsuperscript{1}, Jie Chen\textsuperscript{1,2}

\textsuperscript{1}High Magnetic Field Laboratory, and Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China
\textsuperscript{2}University of Science and Technology of China, Hefei 230026, China
\textsuperscript{3}Institute of Intelligent Machines, and Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei 230031, China

*Corresponding author e-mail: zhang_jun@hmfl.ac.cn

Abstract. The data of Internet of Vehicles (IoV) can be used to evaluate the driving safety risk of auto insurance policyholder and provide technical means for Usage Based Insurance (UBI). There are many types of IoV data, such as continuous or ordinal, categorical, binary etc., which contain highly sparse and dimensional features after One-Hot processing, thus they learning the interaction between critical features and training predictive model difficult. Furthermore, some of the available data have been desensitized, so it is impossible to perform feature engineering based on experience. We propose an end-to-end deep learning framework named Deep Attentional Factorization Machine (DeepAFM), which combines the power of attentional factorization machine with deep learning for feature learning in a new neural network architecture. Compared with existing deep learning models, our approach can learn the weighted interactions between various features effectively by introducing the structure of feature fields without feature engineering. Experimental results showed that our model yields excellent results in real-world data.

1. Introduction

Driving safety risk prediction and analysis provide an effective technical means to assess the driving safety risk, and thus to formulate a reasonable auto insurance rate. In Kaggle competition [1], competitors make full use of the correlation between risk driving, accident rate and auto insurance claims, and define the driving risk value of driver as the probability filing an insurance claim next year. Therefore, the auto insurance policyholder’s claim can be selected as the target variable to an equivalent model for the prediction of driving risk. Driving risk prediction is essentially considered to be a regression problem to model the probability of insurance policyholder’s claims, based on driver’s personal and vehicle’s information, and driving behavior. Learning the implicit feature interaction behind driver, vehicle and driving behavior information, and discovering the feature combination behind the data are of great significance to the prediction. Usually, there are many difficulties such as "feature explosion" in manually combining features. The second challenge is how to effectively train the model, since IoV data sometimes have huge categorical features. Usually, we have to transform the categorical feature to a binary feature through One-Hot coding, but the increase of feature space will lead to an increase in the number of parameters, which also increase the complexity of model training.
Inspired by the enormous success of deep learning in various applications, recently several deep models have been developed to model sophisticated sparse feature interactions. FM modeled pairwise feature interactions as an inner product of latent vectors between features showing the promising results [2]. Factorization-machine supported Neural(FNN) [3] is an FM-initialized feed-forward neural network, which can capture only high-order feature interactions. Product-based Neural Network (PNN) imposed a product layer between the embedding layer and the first hidden layer to capture high-order feature interactions [4]. Moreover, Neural Factorization Machines (NFM) was designed for Sparse Predictive Analytics [5]. In [6, 7], Factorization-Machine based neural network (DeepFM) was proposed for the prediction of click-through rate (CTR). Whereas, DeepFM can learn both high-order and low-order feature interactions. In [8], A novel model named Attentional Factorization Machine (AFM) was proposed, which can learns the importance of each feature interaction from data via a neural attention network. In [9, 10], They use Porto Seguro Data set [11] to train DeepFM for driving safety risk prediction.

This paper proposes an in-depth learning model denote as DeepAFM. Compared to the structure of DeepFM proposed in [6, 7], we introduce attention mechanism in [8], and redesigned the embedding layer referring to [9, 10], which condense the sparse binary features of different fields to low-dimensional vector space. For the prediction problem, the main contributions are summarized as follows:

1) DeepAFM is the integration of DeepFM and AFM, which use factorization and embedding layer to decompose the parameter matrix with complexity $O(n^2)$ into two low-dimensional (for example, $nk \times nk$) parameter matrices, where the quadratic term of FM has $nk$ implicit vectors, and the complexity is reduced to $O(nk)$. The deep component of DeepAFM can learn both low- and high-order feature interactions.

2) DeepAFM can learn the weight of feature interactions via attention mechanism, which extract such features that are important to the prediction.

2. Material and Methods

2.1. Problem Formulation

As mentioned above, the auto insurance claims data provided by the insurance company is formed as a data set and further used to train the proposed model. Supposed that the training data contains $n$ instances $(x, y)$, where $x=(x_1, \ldots, x_m)$ is an $m$-dimensional data record and $y$ is the associated label indicating whether the driver has filed a claim. The raw feature vector $x$ may include categorical features (e.g., CarModels), binary features (e.g., Gender) or continuous/ordinal features (e.g., Age). Therefore, the multi-field categorical form is widely used in auto insurance claims data. Each raw feature is regarded as a field. After One-Hot encoding, each categorical/binary feature is represented as a one-hot vector, while each continuous feature is represented as the value itself. For example, one input instance $[\text{Occupation}=\text{Professor}, \text{CarModels}=\text{MVP}, \text{Gender}=\text{male}, \text{Age}=28, \ldots]$ is normally transformed into a high-dimensional sparse feature via field-aware One-Hot encoding:

$$
\begin{bmatrix}
0 & \ldots & 1 & 0 & \ldots & 1 & 0 & 28
\end{bmatrix}
$$

Thus we can get the field vector space $X_f = (x_{f_1}, \ldots, x_{f_m})$, where $x_{f_i}$ is the vector representation of the $i^{th}$ field of $x$.

$$
x_{f_i} = \begin{cases} 
[\chi_i] & \text{if } \chi_i \text{ is a continuous or ordinal feature in } x_{f_i} \\
[\chi_{i0}, \chi_{i1}] & \text{if } \chi_i \text{ is a binary feature with 2 categories in } x_{f_i} \\
[\chi_{i1}, \ldots, \chi_{id}] & \text{if } \chi_i \text{ is a categorical feature with d categories in } x_{f_i} 
\end{cases}
$$

The target variable of the data set $Y = (y_1, \ldots, y_n)$ indicates whether the driver has filed a claim,
for the $i^{th}$ driver in the training set, where

$$y_i = \begin{cases} 1 & \text{if the } i^{th} \text{ driver has filed a claim} \\ 0 & \text{if the } i^{th} \text{ driver hasn’t filed a claim} \end{cases}$$

(3)

Since the risk of driving is defined as the probability of the driver filing an insurance claim next year, the target variable of driving safety risk prediction is also $Y$.

2.2. Prediction Models

We add the Attention unit to the DeepFM framework proposed in [6, 7]. As depicted in Fig. 1, DeepAFM consists of two components including AFM [8] and DNN, which share the same input. All parameters are trained jointly for the combined prediction model. The predicted risk score can be represented as:

$$\hat{y}(x) = \text{sigmoid}(y_{AFM}(x) + y_{DNN}(x))$$

(4)

where $\hat{y}(x) \in (0,1)$, $y_{AFM}(x)$ is the output of AFM component, and $y_{DNN}(x)$ is the output of DNN component. As presented in Equation 4, the output of AFM component $y_{AFM}(x)$ is part of the final driving safety risk prediction. As shown in Fig. 1, the output of AFM is the summation of an Addition unit and Attention units.

![Figure 1. The architecture of DeepAFM](image)

1) AFM Component of DeepAFM: The AFM component is used to learn weighted feature interactions, while FM is proposed in [2]. FM estimates the target by modelling all interactions between each pair of features. For our task, the FM model expression with raw feature inputs $\chi = (\chi_1, \cdots, \chi_n)$ is as follow:

$$y_{FM}(\chi) = \theta_0 + \sum_{i=1}^{n} \theta_i \chi_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \theta_{ij} \chi_i \chi_j$$

(5)

After One-Hot processing, the raw feature space $\chi$ is transformed to $X = (x_1, \cdots, x_n)$.
according to the field space $X_f$:

\[
\begin{align*}
[x_i & \quad \text{if } x_i = [X_i] \\
[x_i & \quad x_{i,1} = [X_{i0} \quad X_{i1}] \quad \text{if } x_{i,1} = [X_{i0} \quad X_{i1}] \\
[x_i & \quad \ldots \quad x_{i,d} = [X_{i1} \quad \ldots \quad X_{id}] \quad \text{if } x_{i,d} = [X_{i1} \quad \ldots \quad X_{id}]
\end{align*}
\]  
\tag{6}

So the following results can be obtained from Equation 5 and Equation 6:

\[
y_{F\alpha}(x) = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} a_{ij} x_i x_j
\]  
\tag{7}

where $n$ is the number of sample features that have been encoded by One-Hot, and $x_i$ indicates the $i^{th}$ feature. For the field which have categorical, binary raw feature, it just has one non-zero feature for a certain instance after One-Hot processing and makes feature space tend to be highly sparse, high-dimensional, which causes the network overwhelming to train. Therefore, factorization technique is performed on the second-order parameter matrix and a hidden variable space $V$ is introduced to solve the matrix.

\[
V = \begin{pmatrix} v_{11} & \ldots & v_{1k} \\ \vdots & \ddots & \vdots \\ v_{n1} & \ldots & v_{nk} \end{pmatrix}, V_\alpha = (v_{11} & \ldots & v_{n1} \ldots & v_{nk} \ldots & v_{nk})
\]  
\tag{8}

So we get the following results:

\[
W_1 = (a_1 & \ldots & a_n)
\]  
\tag{9}

\[
W_2 = \begin{pmatrix} a_{11} & \ldots & a_{1d} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nd} \end{pmatrix} \begin{pmatrix} V_{11} \\ \vdots \\ V_{n1} \\ \vdots \\ V_{nk} \end{pmatrix} (V_1^T & \ldots & V_\alpha^T)
\]  
\tag{10}

We use the Pairwise Interaction Layer proposed in [8] to expand $n$ vectors to $n(n-1)/2$ interacted vectors, where each interacted vector is the element-wise product of two distinct vectors to encode their interaction. So we get the following result from Equation 7 and Equation 10:

\[
y_{F\alpha}(X) = a_0 + \langle W_1, X \rangle + \sum_{i=1}^{n} \sum_{j=i+1}^{n} (V_i \odot V_j)(x_i x_j)
\]  
\tag{11}

where $V_i \in \mathbb{R}^k$, $\odot$ denotes the element-wise product of two vectors, $W_1$ is used to weigh its order-1 importance, and the latent vector $V_i$ is used to measure the impact of interactions between features $x_i$ and $x_j$. For all fields $X_f$, supposing that the non-zero feature is $x_{i}'$ in the field $x_{i'}$, we can get the following results:

\[
\begin{align*}
[x_i & \quad \text{if } x_{i} = [X_i] \\
[x_i & \quad x_{i,1} = [X_{i0} \quad X_{i1}] \quad \text{if } x_{i,1} = [X_{i0} \quad X_{i1}] \\
[x_i & \quad \ldots \quad x_{i,d} = [X_{i1} \quad \ldots \quad X_{i1}] \quad \text{if } x_{i,d} = [X_{i1} \quad \ldots \quad X_{i1}]
\end{align*}
\]  
\tag{12}

In the field vector space $X_f$, we have $m$ fields in all. For each field $x_{f_m}$, we just have single non-zero feature. The FM model expression with non-zero feature inputs can be rewrited as follow:
\[
y_{FM}(x) = \omega_0 + \langle W'_i, X' \rangle + \sum_{i=1}^{m} \sum_{j=1}^{n} (V'_i \odot V'_j)(x'_i, x'_j)
\]

where \( V'_i \subset V, X' \subset X, W'_i \subset W \). Moreover, an embedding layer is developed to compress the input vector to a low dimensional and dense real-value vector, before further feeding into the network to train the latent vector. We follow the structure of embedding layer as in [6], denote the output of the embedding layer as:

\[
E = (e_i \cdots e_m), e_m = V'_m \cdot X_m
\]

where \( e_m \in \mathbb{R}^k \), \( E \) is a vector space of low dimensional and dense real-values, which can be easily trained in the network. The embedding layer is then fed into AFM component to model order-2 feature interactions, and fed into DNN component to model high-order feature interactions.

Not all feature interactions contribute equally to the classification. Thus we extend the attention mechanism (see Fig. 1) introduced by [12] to capture salient structures of data, which extract salient feature interactions that are important to the classification. The attention model can be also viewed as weighted averaging of output, but the weights are learnt by neural networks [13].

As depicted in Fig. 1, The attention unit takes the input vector \( \{(V'_i \odot V'_j), x'_i, x'_j\} \), which is the output of FM, returns an attention-based vector, which is a weighted arithmetic mean of the input vector, and the weights are chosen according to the importance of each element of the vector. \( \alpha_{xy} \) is the representation of the feature interaction filtered such that only the important parts of the feature interaction remain. We use the following function referring to [8] to calculate the weights:

\[
\begin{align*}
\alpha_{xy} &= \frac{\exp(s_{xy})}{\sum_{i=1}^{m} \sum_{j=1}^{n} \exp(s_{ij})}, \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_{ij} = 1 \\
\end{align*}
\]

where \( \sigma \) is an activation function, \( W_s, W_{xy} \) and \( b_s \) are the parameters to be learned respectively. Given the current feature interaction, it returns the unnormalized importance score \( s_{xy} \). Once the score for all the feature interactions are computed, we can obtain \( \alpha_{xy} \), which is the attention weights at describing the importance of the feature interaction of the \( x'_i \) and \( x'_j \). Next, we add \( \alpha_{xy} \) into the FM model to rewrite the Equation 13 as:

\[
y_{AFM}(x) = \omega_0 + \langle W'_i, X' \rangle + \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_{xy}(V'_i \odot V'_j)(x'_i, x'_j)
\]

2) DNN Component of DeepAFM: DNN is used to learn high-order feature interactions, which is a feed-forward neural network. We follow the structure of DNN as in [6]. The embedding layer is fed into the DNN network. We have

\[
z^{(0)} = (e_i \cdots e_m)
\]

\[
z^{(l+1)} = \sigma \left( W^{(l)}z^{(l)} + b^{(l)} \right)
\]

where \( l \) is the layer depth and \( \sigma \) is an activation function, \( z^{(l)}, W^{(l)}, b^{(l)} \) are the output, model weight, and bias of the \( l^{th} \) layer, respectively. The output of DNN is generated as:

\[
y_{DNN}(x) = W^{H}z^{(H)} + b^{H}
\]

where \( H \) is the number of hidden layers.
2.3. Models Training

2.3.1 Learning. Here we perform the supervised training to minimize prediction error based on DeepAFM model. Since our prediction model is a binary logit model, the most commonly used objective function is Log-loss, which is equivalent to the K-L divergence between two distributions:

\[
J(W) = -\sum_{i=1}^{m} \left( y^{(i)} \log(\hat{y}^{(i)}(x^{(i)})) + (1-y^{(i)}) \log(1-\hat{y}^{(i)}(x^{(i)})) \right) + \min \frac{\lambda}{2} \|W\|^2
\]  

(20)

where \( (x^{(i)}, y^{(i)}) \) is the \( i^{th} \) data instance, \( x^{(i)} \) is the feature vector and \( y^{(i)} \) is the label, \( \hat{y}^{(i)}(x^{(i)}) \) is the prediction of the feature vector \( x^{(i)} \), and \( J(w) \) is the Log-loss of the model on the entire training set. The second term of the Equation 20 represents the weight penalty term \( \lambda \) that can control the strength of the penalty term \( W \).

2.3.2 Overfitting. An overfitting model has poor performance since it overreacts to the given training data. Therefore, in DeepAFM framework, L2-norm and dropout were adopted to prevent the network from overfitting. When the elements of \( W \) become large, the elements of \( W \) are limited with the L2 norm as a penalty term.

2.4. Model Evaluation

We use three evaluation metrics in our experiments: Gini (Normalized Gini Coefficient) [14], AUC (Area Under ROC Curve) [15] and Log-loss (Logistic loss) [16]. Our model is a binary logit model that predicts if a driver will file an insurance claim next year, therefore, the prediction is a probability value that one instance belongs to the claim class. Both AUC and Log-loss are estimated in probabilities, in consistent with the target values of our prediction model.

2.5. Data Preparation and Exploration

This work evaluated the effectiveness of DeepAFM models on Porto Seguro Data set [11]. The dataset is provided by Porto Seguro, one of Brazil’s largest auto and homeowner insurance companies. In the training and test dataset, features are grouped into four categories, ind (individual-related), reg (region-related), car (car-related) and calc (calculation-obtained). The target columns signify whether a claim was filed for that policyholder.

Since instances with “missing value” features affect the prediction performance of model, these features have to be reconstructed. In our work, The missing value of the multi-category feature is interpolated to -1, that is, it turns to construct a new category and thus participate in modeling.

3. Results

We explored the impact of different hyper-parameters of DeepAFM models to find the best hyper-parameters setting, and compared our proposed method with other models, including DeepAFM, PNN, NFM and DNN. We split the Ocslab Driving Dataset into a train set and a test set according to 8:2 for testing the model performance. We take the Gini, AUC and Log-loss of the test set as the final results.

To achieve the best performance for each model on Porto Seguro dataset, parameters were carefully investigated. The hyper-parameters of compared deep models on dataset are stated in Table 1 where the activation function, dropout, structure of hidden layers, optimizer, embedding dimensions of models are given. We keep the deep components of DeepAFM models with the same setting to validate the superiority of the four models.
Table 1. Hyper-parameters of models on Porto Seguro data set

| Model | Activation Function | Dropout | L2-norm penalty \((\lambda)\) | Embedding Dimensions \((k)\) | Layers \((l)\) | optimizer |
|-------|---------------------|---------|-----------------------------|-----------------------------|-------------|-----------|
| DeepAFM | ReLU | 0.5 | 0.04 | 20 | 128-128 | Adam |
| PNN | ReLU | 0.5 | 0.04 | 20 | 128-128 | Adam |
| NFM | ReLU | 0.5 | 0.04 | 20 | 128-128 | Adam |
| DNN | ReLU | 0.5 | 0.04 | 20 | 128-128 | Adam |

The performance of the compared models on Porto Seguro dataset is presented in Table 2. We can observe that our model, learning interactions between features, can improve the prediction performance.

Table 2. Performance of all the compared models

| Model | Gini | AUC | Log-loss |
|-------|------|-----|----------|
| DeepAFM | 0.27421 | 0.63711 | 0.13775 |
| PNN | 0.25983 | 0.62985 | 0.13847 |
| NFM | 0.24892 | 0.62446 | 0.13639 |
| DNN | 0.27161 | 0.63581 | 0.13834 |

4. Conclusion
At present, the development of AI technology in China has been developing by leaps and bounds, thanks to the breakthroughs in data acquisition technology, algorithms and integration technologies. As mentioned earlier, for extensive and multi-dimensional desensitized IoV data which has many types such as continuous or ordinal, categorical, binary, etc, it is tough to perform feature engineering and train predictive model because of the highly sparse and high-dimensional feature. In this paper, we provide an in-depth learning approach to solve the problem of driving risk prediction.

1) DeepAFM introduces a sharing strategy of embedding layer and feature field to solve sparse feature of extensive, multi-dimensional and heterogeneous IoV data. It can be a conventional method for processing data which has many types such as categorical, binary, etc.

2) Our framework can learn both high-order and low-order feature interactions without feature engineering. So it is suitable for training data which has been desensitized avoiding knowing what these features represent.

3) The experiments on real-world auto insurance policyholder’s claim data demonstrated that DeepAFM outperforms the other deep models regarding Normalized GINI, AUC and Log-loss on the dataset.

As a complicated neural networks model, DeepAFM is easy to be overfitting, which can lead to poor performance since it overreacts to the given training data. Experimental results showed that DeepAFM suffers from overfitting, especially for the AFM compoent, although Dropout and L2-norm are utilized to regularize the objective function. How to solve the problem of over-fitting is the next important task.

5. Acknowledgments
This research was financially supported by The Science and Technology Service Network (STS) Double Innovation Project of the Chinese Academy of Sciences, the construction and application of the comprehensive management service platform for urban intelligent business travel (Grant No. KFJ-STS-SCYD-017).

References
[1] kaggle, https://www.kaggle.com/c/porto-seguro-safe-driver-prediction, 2017
[2] S Rendle, "Factorization Machines", Proc. IEEE International Conference on Data Mining 2011.
[3] Weinan Zhang, Tianming Du, Jun Wang, "Deep Learning over Multi-field Categorical Data: A Case Study on User Response Prediction", arXiv:1601.02376, 2016
[4] Y Qu, H Cai, K Ren, W Zhang, Y Yu, "Product-Based Neural Networks for User Response Prediction". Proc. IEEE International Conference on Data Mining2017 pp. Pages

[5] He, Xiangnan, Chua, TatSeng, "Neural Factorization Machines for Sparse Predictive Analytics", 2017, pp. 355-364

[6] H Guo, R Tang, Y Ye, Z Li, X He, "Deepfm: a factorization-machine based neural network for ctr prediction", arXiv preprint arXiv:1703.04247, 2017

[7] H Guo, R Tang, Y Ye, Z Li, X He, "DeepFM: An End-to-End Wide & Deep Learning Framework for CTR Prediction", arXiv preprint arXiv:1804.04950, 2018

[8] J Xiao, H Ye, X He, "Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks", 2017, pp. 3119-3125

[9] Princwewen, Available online: https://github.com/princwewen/tensorflow_practice/tree/master/recommendation/Basic-DeepFM-model (accessed on 10 October 2018).

[10] ChenglongChen, Available online: https://github.com/ChenglongChen/tensorflow-DeepFM (accessed on 15 October 2018).

[11] Porto Seguro, Available online: https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/data (accessed on 12 November 2018).

[12] heuritech, Available online: https://blog.heuritech.com/2016/01/20/attention-mechanism/ (accessed on 19 November 2018).

[13] S Yao, S Hu, Y Zhao, "DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing", 2016, pp. 351-360

[14] Kilian Batzner, Available online: https://www.kaggle.com/batzner/gini-coefficient-an-intuitive-explanation/notebook (accessed on 29 November 2018).

[15] T Fawcett, "An introduction to ROC analysis", Pattern Recognition Letters, 2005, 27, (8), pp. 861-874

[16] A Painsky, GW Wornell, "On the Universality of the Logistic Loss Function", arXiv:1805.03804, 2018