Deeper Attention-Based Network for Structured Data

Xiaohua Wu, Youping Fan, Wanwan Peng, Hong Pang, and Yu Luo(✉)

School of Information and Software Engineering,
University of Electronic Science and Technology of China, Chengdu 610054, China
luoyu@uestc.edu.cn

Abstract. Deep learning methods are applied into structured data and in typical methods, low-order features are discarded after combining with high-order featuresfor prediction tasks. However, in structured data, ignorance of low-order features may cause the low prediction rate. To address this issue, in this paper, deeper attention-based network (DAN) is proposed. With DAN method, to keep both low- and high-order features, attention average pooling layer was utilized to aggregate features of each order. Furthermore, by shortcut connections from each layer to attention average pooling layer, DAN can be built extremely deep to obtain enough capacity. Experimental results show DAN has good performance and works effectively.

Keywords: Structured data · DeepLearning · Feature aggregation

1 Introduction

Deep learning [12] has been receiving widespread attention since Krizhevsky et al. proposed AlexNet [11], which won the championship on imagenet dataset [3] in 2012. Numerous experiments have established that deep learning has led to many breakthroughs across various areas including natural language processing [4], computer vision [6], speech and audio processing and many other areas [19]. However, less research has focused on processing structured data using deep learning methods.

Data in many areas is structured, such as recommender systems [18]. In traditional methods, linear models are applied to structured inputs, such as LR, FTRL [13]. However, these methods lack the ability to combined features. In contrast, due to the ability of automatically combine features, in recent years, researchers have gradually started using deep learning methods to process structured data. Typically efforts fall into two groups. One is multilayer perceptron (MLP) [16], which is used as backbone to learn interactions among features, such as Wide&Deep [2], Deep&Cross [17], and DIN [20]. And for the other, common methods combine Factorization Machine [15] and MLP. Factorization Machine is used to model the pairwise interactions between features, and then high-order feature interactions are learned by MLP such as PNN [14], DeepFM [5]

Supported by Sichuan Science and Technology Program 2018GZDZX0042 and 2018HH0061.

and NFM [8]. These methods aim at combining low-order features to obtain higher-order features. Nevertheless, only higher-order features are used for predictions. Unlike meaningless single pixel which needs to be combined into higher-level abstract features in an image, features of all orders in structured data are meaningful for predictions.

Recently, the number of samples and feature sizes in structured data are getting larger and larger. It is necessary to build a network with large capacity. In other areas of deep learning, some methods have made many breakthroughs. Typically works to build large capacity networks include ResNet [7], DenseNet [10] and so on. The key point of these methods to make networks deeper is to address the gradient vanishing problem by shortcut connections.

In order to address the above problems, we propose Deeper Attention-Based Network (DAN), which is based on MLP. Considering the need for feature information of all orders, DAN use attention average pooling layer to convert the outputs of all MLP hidden layers into a fixed-length vector, which contains information of all order features. Furthermore, in DAN, only one dense layer is stacked after attention average pooling layer. Hence, all parameters of DAN are very closed to the output. DAN will not encounter gradient vanishing. In this paper, our main contributions are summarized as follows:

- We propose a novel network architecture DAN, which can aggregate the features of each order through attention average pooling layer. By automatically adjusting attention weight parameters in pooling layer, DAN can focus on higher-order or lower-order features to adapt different datasets.
- Also, we design shortcut connections through attention average pooling layer and only one stacked dense layer after it. Despite the depth is significantly increased, DAN does not suffer from gradient vanishing.

2 Preliminary

Before introducing our proposed method, we give a brief review of the feature representation and multilayer perceptron (MLP).

2.1 Feature Representation

Structured data includes numerical and categorical values. Numerical values refer to continuous and discrete values such as [height = 1.75, age = 23]. In contrast, categorical values are in a limited set, for example, [gender = Female]. The normalized numerical values can be used directly as the input of deep neural networks. In general, categorical value is represented by a one-hot vector such as [gender = Female] and [1, 0], [gender = male] and [0, 1].

However, due to sparsity and high dimensionality, one-hot vectors are not suitable as deep neural network inputs. In order to better extract categorical feature to improve performance, high dimensional sparse one-hot vectors are embedded into low dimensional dense vector spaces [1]. For i-th categorical feature t_i, which is a one-hot vector, let $E^i = [e_i^1, e_i^2, \cdots, e_i^k] \in \mathbb{R}^{d \times k}$ represents the i-th embedding matrix, where d is the dimension of low dimensional dense vector, k is the size of i-th feature value set.

Then, all embedding vectors of categorical features and normalized numerical features are concatenated into the input vector $x = [e_1, e_2, \cdots, e_m, v_1, \cdots, v_n]$, where e is the embeddings of categorical features with the number m. And v is the embeddings or values of numerical features with the number n.

2.2 MLP

Multilayer perceptron provides a universal approximation framework by stacking hidden layers [9]. Hence, it is fully capable of learning higher-order interactions among features in structured data. Firstly, categorical features are transformed into low-dimensional dense vectors through embedding matrix. Then low-dimensional dense vectors and continuous features are as the input of MLP. MLP extracts higher-order features by stacking more hidden layers. Formally, the definition of l-th hidden layers is as the follow:

$$A_l = \sigma(W_l A_{l-1} + b_l), \tag{1}$$

where W_l, b_l denotes weight matrix and bias, σ is non-linear activation function, A_l, A_{l-1} denotes the output of this and last layer, specially, A_0 is the input matrix X consisting of all input vectors x.

3 Deeper Attention-Based Network

Deeper Attention-Based Network (DAN) is based on MLP. There are two motivations behind it. Firstly, without discarding any information, DAN keeps the features of each order for predictions. In this paper, we use attention average pooling layer to aggregate the features of all orders. To build large capacity networks is our second motivation. Thus, there are shortcut connections in DAN. Making all learned parameters closed to the output, after aggregation, DAN only stack one dense layer. At last, we also theoretically prove why DAN can be built very deep. The architecture of DAN is shown in Fig. 1.

Notation: Let A_i denote the output of i-th layer, W_i, b_i denote the weight matrix and bias, A_0 is the input matrix X consisting of all input vectors x.

3.1 Feature Aggregation

In order to obtain the abstract features of each layer, we must extract the output of all hidden layers of the MLP. Next, we aggregate all outputs as the input of a shallow feedforward neural network with a fix-length input. Nevertheless, the number of the outputs changes as the hidden layers are stacked. It is a common practice to transform the list of vectors to get a fixed-length vector via a pooling layer:

$$C = pooling(A_0, A_1, \cdots, A_L), \tag{2}$$

where A_i is the output of i-th layer, L is the number of MLP hidden layers, C is a fix-length vector. The most common pooling operations are max pooling and average

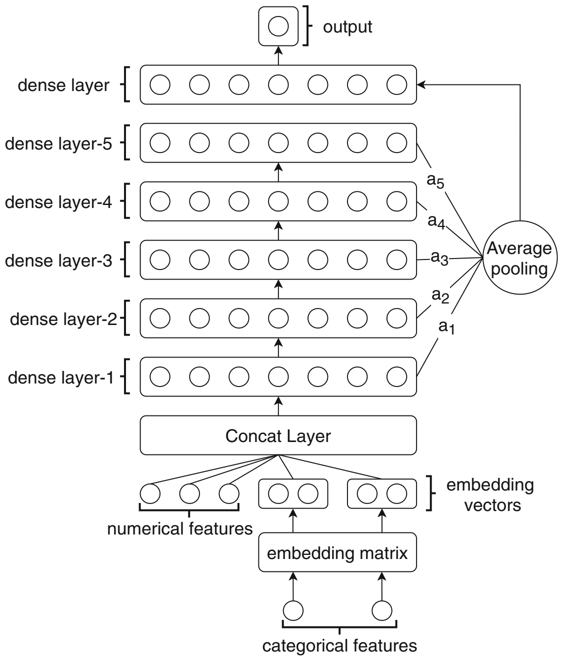

Fig. 1. The architecture of DAN based on MLP

pooling. Max pooling selects the maximum value and average pooling calculates the average value of each position among all vectors. In this paper, our purpose is to take into account the abstract features of each level, we use average pooling without dropping any values.

In addition, on various datasets and tasks, the output decision focuses on different levels of the abstract features. Attention average pooling with learnable weight parameters is used to address it in DAN:

$$pooling(A_0, A_1, \cdots, A_L) = \sum_{i=1}^{L} \alpha_i A_i, \tag{3}$$

where α_i denotes learnable weight parameter. Through backpropagation algorithm, DAN can automatically find one group of adaptive weight parameters for the specific datasets and tasks. For a DAN with l hidden layers, if $[\alpha_1, \alpha_2, \cdots, \alpha_{l-1}, \alpha_l] = [0, 0, \cdots, 0, 1]$, DAN is the same as a MLP. Hence, our proposed DAN contains MLP and has larger capacity than MLP. The optimal solution of DAN must be at least equal to the MLP. Intuitively, DAN has a better performance than MLP.

3.2 Deeper Network

Gradient vanishing is the problem that must be considered when building deeper networks. To tackle these problems, in this paper, we make the trainable parameters closer to the output by shortcut connections. In DAN, all layers containing trainable parameters have a path directly connected to the input of the pooling layer. After the pooling layer, we just stack a single dense layer to avoid the trainable parameters being too far from the output:

$$\hat{y} = \sigma(WC + b), \tag{4}$$

where C is the output of pooling layer, W, b is weight matrix and bias of last layer, $\hat{y}$ is the output of DAN. In last layer, two most commonly used non-linear activation function σ are $sigmoid$ and $softmax$.

3.3 Theoretical Proof

We study the back-propagated gradient of the $Loss$ function on the inputs weights at each layer in DAN. There are $L - l + 1$ path to the output from l-th layer in DAN. For weight matrix W_l of l-th layer, the gradient of the $Loss$ function is as follows:

$$\frac{\partial Loss}{\partial W_l} = \sum_{i=l}^{L} \frac{\partial A_l}{\partial W_l} \cdot \frac{\partial A_i}{\partial A_l} \cdot \frac{\partial C}{\partial A_i} \cdot \frac{\partial Loss}{\partial C}, \tag{5}$$

$$\frac{\partial Loss}{\partial W_l} = \sum_{i=l}^{L} \alpha_i \frac{\partial A_l}{\partial W_l} \prod_{j=l+1}^{i} \frac{\partial A_j}{\partial A_{j-1}} \cdot \frac{\partial Loss}{\partial C}, \tag{6}$$

where α_i denotes learnable weight parameter of i-th layer, A_j denotes the output of i-th layer. In the formula (6), there is always one term $\alpha_l \cdot \frac{\partial A_l}{\partial W_l} \cdot \frac{\partial Loss}{\partial C}$ where gradients are only propagated three times. Thus, no matter how deep the network layer is, DAN will not encounter the problem of gradient vanishing.

Since the gradient of all parameters will not be too small, during the training process, the convergence speed of DAN will be relatively fast. Fast convergence speed is very important for large data sets.

4 Experiments

4.1 Experimental Settings

Dataset. We evaluate the effectiveness of DAN on two public datasets. **Criteo** dataset was used for the Display Advertising Challenge hosted by Kaggle and includes 45 million users'click records. **Porto Seguro** dataset was used to predict the probability that an auto insurance policy holder files a claim.

Evaluation Metrics. To evaluate the performance, we adopt AUC (Area Under ROC) and Logloss. AUC is a widely used metric in CTR prediction field. It sets different thresholds to evaluate model performance. And Logloss is the value of $Loss$ function where a lower score indicates a better performance.

Models for Comparisons. We compare DAN with two models: basic model (MLP) and DeepFM which includes MLP and Factorization Machine (FM).

Hyper-parameter Settings. To be fair, all models use the same setting of hyper-parameters shown in Table 1.

Table 1. Hyper-parameters setting

	Criteo	Porto Seguro
Embedding size	32	8
Hidden layer width	256	64
Hidden layer depth	5, 10, 20, 50, 100	5, 10, 15, 20, 25
Batch size	256	256
Learning rate	3e−5 with Adam	3e−5 with Adam
Dropout rate	0.5	0.5
Training epochs	1 to 10	1 to 20

Table 2. The best result on two datasets

	Criteo		Porto Seguro	
	AUC	Logloss	AUC	Logloss
MLP	0.8028	0.4656	0.6290	0.1529
DeepFM	0.8015	0.4546	0.6260	0.1531
DAN	**0.8037**	**0.4478**	**0.6330**	**0.1524**

4.2 Performance Evaluation

In this section, we evaluate model performance on **Criteo** and **Porto Seguro** datasets based on the hyper-parameters listed in last section. While keeping other hyper-parameters constant, we gradually make the network deeper. Then the best results are chosen for different models on both dataset and shown as Table 2.

Overall, DAN beats other competitors in AUC and Logloss. In fact, compared with other models, our proposed model improves performance as the number of layers increases. In next section, we show the details through experiments of Comparison.

4.3 Layer Depth Study

In this section, first we list the details of different layers. Next, we observe the trend of the evaluation score with epochs. Furthermore, we analyze the reasons for the experimental results.

Details. In our experiments, we mainly explore the effect of the layer depth which is closely related to model capacity. Let layer depth be [5, 10, 20, 50, 100] or **Criteo** while keeping other hyper-parameters constant.

Table 3. AUC of different layers on Criteo

	5	10	20	50	100
MLP	0.8028	0.8017	0.7931	0.500	0.500
DeepFM	0.8015	0.8015	0.7987	0.7862	0.7866
DAN	**0.8032**	**0.8033**	**0.8035**	**0.8035**	**0.8037**

Table 4. Logloss of different layers on Criteo

	5	10	20	50	100
MLP	0.4656	0.4672	0.4673	0.5692	0.5692
DeepFM	0.4546	0.4633	0.4730	0.4638	0.4634
DAN	**0.4482**	**0.4481**	**0.4479**	**0.4479**	**0.4478**

Table 3 to Table 4 show AUC and Logloss with different layers. We have the following observations:

- Our proposed model DAN outperforms MLP and DeepFM with any layer depth. This verifies the usefulness of combining features of each order.
- The performance of MLP and DeepFM drops sharply when the network is extremely deep. However, due to no gradient vanishing, the deeper network has higher performance for DAN.
- DeepFM has better performance than MLP with network very deep. The reason is consistent with the intuition behind DAN. There is a short path from input to output through FM. The trainable parameters in FM component are closed to the output of DeepFM.

Trends. Firstly, we observe the trends of the AUC and Logloss curves under the condition that the layer depth is 5 without gradient vanishing. Then, to highlight the key advantage of DAN, we adjust the layer depth to 100 for **Criteo**.

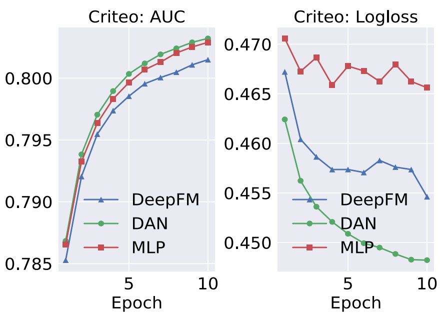

Fig. 2. Trends of layer depth 5 on Criteo

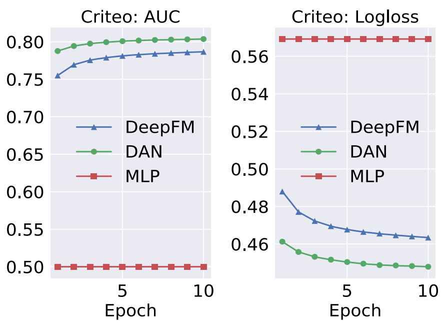

Fig. 3. Trends of layer depth 100 on Criteo

Figure 2 compares the AUC and Logloss of DeepFM, MLP and DAN of each epoch with the layer depth 5. We observe that DAN converges faster and obtains a better

Logloss and AUC score than MLP and DeepFM, which indicates that DAN can fit the data better because of greater capability. Faster convergence speed of DAN helps us get better performance in less time that is important for actual productions.

Figure 3 shows the key advantage of DAN, which can be built extremely deep and obtain better performance. The deeper networks mean that the greater capability. However, with the network depth significantly increasing, MLP completely degrades with AUC 0.5 which means random prediction. Although there is a short path to the output in Factorization Machine, DeepFM's performance still drops significantly. Only DAN gets benefit from the very deep architecture.

5 The Conclusions

In this paper, we propose DAN based on MLP. DAN gains performance improvement with any layer depth by extracting the features of each order and being built more deep. The results show that DAN has faster convergence speed and better performance when architecture is very deep.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)
2. Cheng, H., et al.: Wide & deep learning for recommender systems. In: DLRS@RecSys, pp. 7–10. ACM (2016)
3. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: Imagenet: a large-scale hierarchical image database. In: CVPR, pp. 248–255. IEEE Computer Society (2009)
4. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1), pp. 4171–4186. Association for Computational Linguistics (2019)
5. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: DeepFM: a factorization-machine based neural network for CTR prediction. In: IJCAI, pp. 1725–1731. ijcai.org (2017)
6. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: ICCV. pp. 2980–2988. IEEE Computer Society (2017)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778. IEEE Computer Society (2016)
8. He, X., Chua, T.: Neural factorization machines for sparse predictive analytics. In: SIGIR, pp. 355–364. ACM (2017)
9. Hornik, K., Stinchcombe, M.B., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
10. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR, pp. 2261–2269. IEEE Computer Society (2017)
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1106–1114 (2012)
12. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
13. McMahan, H.B., et al.: Ad click prediction: a view from the trenches. In: KDD, pp. 1222–1230. ACM (2013)
14. Qu, Y., et al.: Product-based neural networks for user response prediction over multi-field categorical data. ACM Trans. Inf. Syst. **37**(1), 5:1–5:35 (2019)

15. Rendle, S.: Factorization machines. In: ICDM, pp. 995–1000. IEEE Computer Society (2010)
16. Rosenblatt, F.: Principles of neurodynamics. Perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc., Buffalo, NY (1961)
17. Wang, R., Fu, B., Fu, G., Wang, M.: Deep & cross network for ad click predictions. In: ADKDD@KDD, pp. 12:1–12:7. ACM (2017)
18. Xi, W., Huang, L., Wang, C., Zheng, Y., Lai, J.: BPAM: recommendation based on BP neural network with attention mechanism. In: IJCAI, pp. 3905–3911. ijcai.org (2019)
19. Xinyi, Z., Chen, L.: Capsule graph neural network. In: ICLR (Poster). OpenReview.net (2019)
20. Zhou, G., et al.: Deep interest network for click-through rate prediction. In: KDD, pp. 1059–1068. ACM (2018)