Res-embedding for Deep Learning Based Click-Through Rate Prediction Modeling

Guorui Zhou, Kailun Wu, Weijie Bian, Zhao Yang, Xiaoqiang Zhu, Kun Gai
Alibaba Group
{guorui.xgr,kailun.wukailun,weijie.bwj,haoyi.yz,xiaoqiang.zxq,jingshi.gk}@alibaba-inc.com

ABSTRACT

Recently, click-through rate (CTR) prediction models have evolved from shallow methods to deep neural networks. Most deep CTR models follow an Embedding&MLP paradigm, that is, first mapping discrete id features, e.g. user visited items, into low dimensional vectors with an embedding module, then learn a multi-layer perception (MLP) to fit the target. In this way, embedding module performs as the representative learning and plays a key role in the model performance. However, in many real-world applications, deep CTR model often suffers from poor generalization performance, which is mostly due to the learning of embedding parameters. In this paper, we model user behavior using an interest delay model, study carefully the embedding mechanism, and obtain two important results: (i) We theoretically prove that small aggregation radius of embedding vectors of items which belongs to a same user interest domain will result in good generalization performance of deep CTR model. (ii) Following our theoretical analysis, we design a new embedding structure named res-embedding. In res-embedding module, embedding vector of each item is the sum of two components: (i) a central embedding vector calculated from an item-based interest graph (ii) a residual embedding vector with its scale to be relatively small. Empirical evaluation on several public datasets demonstrates the effectiveness of the proposed res-embedding structure, which brings significant improvement on the model performance.

ACM Reference format:
Guorui Zhou, Kailun Wu, Weijie Bian, Zhao Yang, Xiaoqiang Zhu, Kun Gai. 2016. Res-embedding for Deep Learning Based Click-Through Rate Prediction Modeling. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 9 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In recommender systems, CTR (click-through rate) prediction is a crucial task, which has attracted a lot of attention and become the cornerstone. The aim of CTR model is to predict the probability of one user click a given candidate item, which will decide the final ranking of items presented to users.

Thanks to the rapid development of deep learning technology[14], deep neural network based models for CTR prediction task have gained significant progress and become state-of-the-art methods. Most deep CTR models follow a Embedding&MLP paradigm, as illustrated in Fig.1: (i) **Embedding** module which maps discrete id features, e.g., user historical clicked items, into low dimensional vectors and then transformed into a fixed-length vector by pooling. (ii) **MLP** module which aims to learn the nonlinear relationship among features and fit the target by a fully connected network, a.k.a. multi-layer perception.

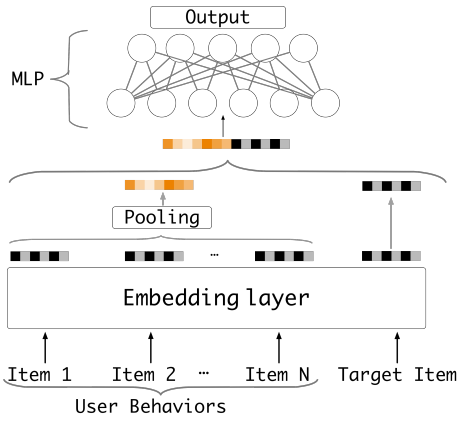


Figure 1: The illustration of the Embedding&MLP structure.

Many approaches have been proposed to improve the performance of deep CTR model, such as PNN[16], DeepFM[9], DIN[26] and DIEN[25]. However, most work focus on designing new network architectures to substitute MLP module, aiming to better capture the nonlinear relationship among features, while little effort has been made to improve the basic yet important embedding module. In this paper, we study carefully the embedding mechanism theoretically and try to fill the gap in practical.

Generally speaking, traditional embedding module works in a look-up table manner, that is, each discrete feature corresponds to a low dimensional vector, with parameters learned from training data of CTR task. Note that parameters of both embedding module and MLP module are learned end-to-end. In this way, embedding module actually performs as a representative mapping and determines the input distribution of the subsequent MLP module. Referring to the data-dependent generalization theory[12], input distribution will influence the generalization performance of model. Therefore, embedding module is vital for the generalization performance of deep CTR model. As [26] reported, in practice, overfitting phenomenon commonly exists during training of deep CTR model, especially in industrial applications with large scale discrete features. We argue that it is the embedding module that might cause the poor generalization performance. The reason lies in two-folds: (i) In many real systems, number of features can scale up to billions, causing the number of embedding parameters to be huge. This would promote the memory ability but decrease the generalization ability. (ii) With the supervision of click labels only, it might be hard for the traditional embedding module to learn a representative mapping with high generalization ability. For example, distance of embedding vectors of two similar items might change greatly with different initializations in the end-to-end training manner.

Conference'17, Washington, DC, USA

DOI: 10.1145/nnnnnnn.nnnnnnn

Motivated by the above observation, in this paper, we propose to (i) quantitatively analyze which variables are involved in the generalization error bound of deep CTR models, and (ii) design corresponding solutions to enhance generalization ability according to this quantitative relationship.

We take the recommender system in e-commerce industry as an example. In e-commerce scenarios, we shall first model users' behavior with an fiinterest-delayfi model. According to experience, we suppose that user's interests will last for a while and users' click behaviors are generally affected by their interests. Different interests will drive users to click different types of items. Hence, assume that each item has its own interest domain. E.g., item "iphone 6" might belong to the interest domain of "smartphone". Intuitively, items belonging to the same interest domain should be similar and distances between their embedding vectors should be small. In fact, we do have proved this mathematically. Specifically, **we prove that the generalization error of deep CTR model is bounded by the envelope radius of items with the same interest domain in the embedding space**. Moreover, following this theoretical analysis, we design a new residual embedding structure which is theoretically helpful for improving the generalization ability named as res-embedding. In this structure, **embedding vector of each item is the sum of a central embedding vector and an independent low-scale residual embedding vector**. Items in the same interest domain have similar central embedding vectors. To achieve this goal, we build an item-based interest graph based on the co-occurrence frequency of items in user historical behavior sequences. Central embedding vector of each item is calculated as the linear combination of the central embedding basis vectors of its neighboring items in the interest graph, with the linear combination coefficients calculated by three practical implementations including average, GCN (Graph Convolutional Network)[13] and attention. Besides, the residual embedding vector is forced to be small-scale, by penalizing the l_2-norm of parameters of residual embedding vectors on the final objective function of deep CTR model.

Contributions of this paper are summarized as follows:

- We theoretically proved that increasing the aggregation degree of embedding vectors of items in the same interest domain helps decrease the generalization error bound. This may be a direction worth studying for the future to improve the generalization performance of embeddings.
- Following the theoretical analysis, we propose a new res-embedding structure as well as three practical implementations including average, GCN and attention. In addition, We also mathematically prove the consistency of theory and method.
- We conduct careful experiments on several public datasets. Adding with res-embedding structure, state-of-the-art deep CTR models all gains significant improvement on the AUC metric. This clearly verifies the proposed theory as well as res-embedding method.

The rest of the paper is organized as follows. The related work is summarized in section 2. Section 3 theoretically analyzes the influence of the distribution of embedding vectors on the generalization performance of deep CTR prediction model.Then we propose the detail of the res-embedding and explain its consistency with theoretical analysis in section 4. Our experimental results are shown in section 5 and conclusion is shown in section 6.

2 RELATED WORK

Deep CTR model: Recently, the deep learning has been widely used in the CTR prediction task. In the very first, NNLM [3] learns the representation of each word, which has inspired many natural language models and CTR prediction models that need to handle large-scale sparse input features. Piece-wise Linear Models (LS-PLM) [7] and factorization machine (FM) [17] adopt the embedding layer for the sparse input feature and capture the relationship amongs the different features through the specific form functions, which can be regard as a single-layer neural network. Based on these model, many deeper models like Deep Crossing [19], Wide&Deep Learning [4] and YouTube Recommendation CTR model [5] employ more complex MLP. PNN[16] tries to capture high-order feature interactions by involving a product layer after embedding layer. Therefore, there are also some works[26] that extract advanced information with more abstract features. Though improvements based on deep model continuously enhance the performance of CTR prediction tasks, their inputs are mostly from the original embedding layer. We design embedding layer to promote the generalization performance based on these networks.

Embedding learning: Traditional methods calculate embedding vector by the relationship between high dimensional data. Some embedding algorithm like Laplacian Eigenmaps [2] and LLE (Locally Linear Embedding) Graph Factorization [18], LINE [20] keep stronger related nodes closer to each other in the vector space. Some deep methods like GCN [13] define a convolution operator on graph to learn the embedding based on the graph. The model iteratively aggregates the embeddings of neighbors for a node and uses them to obtain the new embedding. Individual literature [15] uses an additional network and context information of the data to pre-train embedding vectors to predict the CTR model.

Generalization: [23] proposes a theoretical framework to define the robustness of learning algorithm and proves that generalization performance of the learning algorithm is determined by the robustness of a learning algorithm. It provides the robustness of some common learning algorithms like SVM, DNN and so on. This theoretical framework and example based on DNN help us to theoretically analyze the influence of distribution of embedding vectors on the generalization of deep CTR model. [24] utilizes this framework to study the relationship of metric structure of the features and robustness of the algorithm and proposes structured metric learning method. Other similar work [21][22] discuss the structure of the embedding should be tide. Compared with these work, our paper study more complex structure of embedding vectors. With modeling based on the interest state, we prove that embeddings under a kind of group aggregated structure could promote the generalization of the deep model.

3 THEORETICAL ANALYSIS

In this section, we shall first model the user's click behavior above the user's interest called interest delay model. For ease of expression, we divide interest into several categories, called interest domains. Then, the mathematical discussion of generalization of CTR model is based on the interest delay model. According to some generalization theory, we finally come to the conclusion that the generalized error bound can be effectively reduced by reducing the envelope radius of the items belonging to the same interest domain in the embedding space without greatly changing the total envelope radius of all items in the embedding space. Based on this conclusion, we propose a basic prototype of res-embedding and its final improved version.

3.1 Notation

Expression	Meaning
d	The dimension of embeding space
$\{x^1, \cdots, x^M\}_h$	The user's click behavior sequence with length
x_t	The target item
y^*	Ground-truth label
$f(\cdot, \cdot)$	Deep CTR model
z	Interest hidden state
N_z	Number of the Interest hidden state
$P(x\|z)$	Conditional distribution of item x under the Interest hidden state x
$P(x), P(z)$	The marginal distribution of item x and prior distribution of interest hidden state z
T, p	Number of the preriods and the time step
N	The number of the training samples
$l(f, s)$	Loss function of the sample s based on the model f
$E_s(l(f, s))$	Expected loss
$\sum_{i=1}^{N} l(f, s_i))/N$	Empirical loss
$\phi(V)$	Envelope radius of the set V

Table 1: The notations of some main Expressions and their meanings.

The expressions and variables in Table **??** are some important mathematical symbols in this section. The specific definition and detailed introduction are shown as follows.

Embedding Space: In this section, all of the items have their own representation vectors in the d-dimensional embedding space $\mathbb{R}^d$; We mainly analyze the influence of the distribution of items in this embedding space on CTR model. For simplicity, item with embedding vector is referred to as item itself in this section (e.g. "item x" means item with embedding vector x.)

Data and model: In CTR prediction task, one sample s is compose of the input features and label. That is $s = \{\{\{x^1, \cdots, x^M\}_h, x_t\}, y^*\}$. The input features of one sample are composed of user's behavior sequence, i.e. the clicked item sequence $\{x^1, \cdots, x^M\}_h$, ($x^i \in \mathbb{R}^d$ means item clicked in i-th time step, M is the total number of time steps) and the target item $x_t \in \mathbb{R}^d$. The label of the sample $y^* \in \{0, 1\}$ indicates whether the user clicked on the target item. The CTR prediction model $f(\cdot, \cdot) : (\mathbb{R}^{M \times d}, \mathbb{R}^d) \rightarrow [0, 1]$ is learned by fitting training samples. In this paper, we shall discuss the paper around the D layers MLP with ReLU nonlinear function which is defined in Definition 1

Definition 1. *D layers MLP with ReLU nonlinear function and $n \times d$ input dimensions is defined as follows. The input is concatenation of $\{x_1, \cdots, x_n\}$ and $x_i \in \mathbb{R}^d$ for $\forall i$, and the label is $y^* \in \mathcal{Y}$($\mathcal{Y} = \{0, 1\}$). The trainable parameters are $\{W_t, b_t\}$ for $\forall t = 1, \cdots, D$. Its forward process is*

$$\begin{aligned} h^0 &= [x_1, \cdots, x_n], \\ h^t &= ReLU(W_t h^{t-1} + b_t), \\ f(x_1, x_2, \cdots, x_n) &= sigmoid(W_D h^{D-1} + b_D). \end{aligned} \tag{1}$$

$ReLU(x) = max(x, 0)$ and the $sigmoid(x)$ is the sigmoid function.

Generalization error bound: Generalization error bound of model f is the upper bound of the absolute value of the difference between expected loss $E_s(l(f, s))$ and empirical loss $\sum_{i=1}^{N} l(f, s_i)/N$. $l(f, s)$ is the loss function of the ground-truth label of s and the output of f when the input is the feature of s. Empirical loss is the expectation of loss based on distribution of sample s,i.e $E_s(l(f, s)) = \int_s l(f, s)p(s)$, and experience loss is the average loss of all training samples s_i, $i \in \{1, \cdots, N\}$. N is the number of training samples.

Envelope radius: For a set $V \subset \mathbb{R}^d$, if $\exists w \in \mathbb{R}^d$ s.t. $\forall s \in V$, $\|w - s\|_2 \leq R_0$, then envelope radius of set V is no more than R_0, which is defined as $\phi(V) \leq R_0$.

3.2 User Behavior Modeling: Interest Delay Model

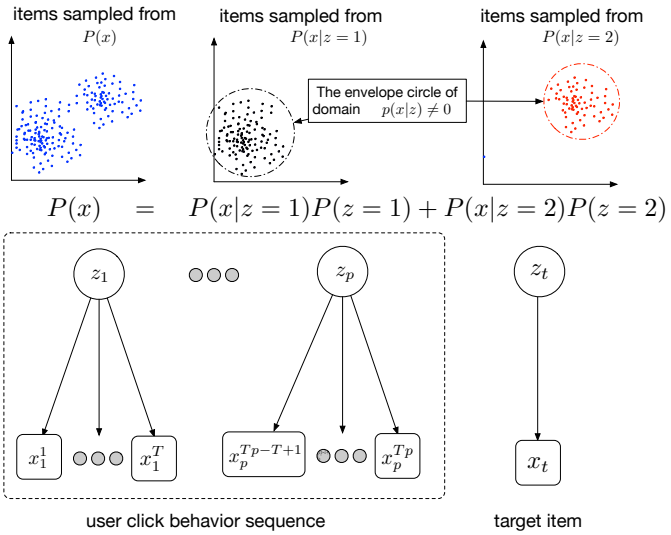


Figure 2: The upper part of this figure is the case of $N_z = 2$. The distribution of item embedding $P(x)$ is composed of conditional distribution $P(x|z = 1)$ and $P(x|z = 2)$. The two dashed circle is the envelope circle of the domain$P(x|z) \neq 0$. The lower part of this figure is the sampling process of behavior sequence and target item of one sample. The behavior sequence is generated from the hidden state of p time periods from z_1 to z_p. One hidden state controls T time steps.

Intuitively, when browsing products in a E-commerce website, users will click different products due to different interest. Based on experience and intuitive common sense, we assume that the user's interest will last for a period of time when browsing the internet. Each click is defined as one fitime stepfi and the sequence of the time steps with the same interest is one "period". This model is so called "interest delay" model.

Focusing on a single time step, the user's single click behavior is determined by the user's current interests. We mathematically illustrate this behavior as the sampling process as follows. Assume that there are N_z interest domains, and interest hidden state $z \in \{1, 2, \cdots, N_z\}$ is defined as a quantitative discrete variable of interest domain. The probability distribution function $P(x)$ of one user click item x is related to his interest hidden state z as $P(x|z)$. The distribution of items in embedding space consists of conditional distribution $P(x) = \sum_z P(x|z)P(z)$, where $P(z)$ is the prior probability distribution of z. As shown in upper part of Figure

2, the total distribution of the clicked item $P(x)$ can be regarded as a combination of two conditional distributions $P(x|z=1)$ and $P(x|z=2)$. By the way, we define the domain of $P(x|z)$ is the area that $P(x_0|z=i) \neq 0$.

A click behavior sequence of user in interest delay model is generated from two phases: Firstly, a user determine an interest hidden state z_i in each period(Totally p periods) with T time steps, and then sample (randomly click) a item $x_{p_i}^t$ with the influence of interest hidden state z_i. Thus, user's clicked items sequence could be written as $\{\{x_{p_1}^1, x_{p_1}^2, \cdots, x_{p_1}^T\}, \{x_{p_2}^{T+1}, \cdots, x_{p_2}^{2T}\}, \cdots, \{\cdots, x_{p_p}^{T\times p}\}\}_h$ (p_i means the i-th period, $M = T \times p$). The sampling process of behavior sequence is shown in lower part of Figure 2. Assumption 1 summarizes the quantitative sampling process of user behavior in the interest delay model.

Assumption 1. *Training sample* $s = (\{\{x_{p_1}^1, \cdots, x_{p_1}^T\}, \{x_{p_2}^{T+1}, \cdots, x_{p_2}^{T\times 2}\}, \cdots, \{x_{p_p}^{T\times(p-1)+1}, \cdots, x_{p_p}^{T\times p}\}\}_h, x_t, y^*)$ *consists of clicked items* $\{x^1, \cdots, x^{T\times p}\}_h$ *in p periods with T time steps, target item* x_t *and label* y^*. *The interest hidden state of target item* z_t *is sampled from* $P_t(z)$, *and target item* x_t *are sampled from the conditional distribution* $P(x|z)$. *For user's click behavior, interest hidden state sequence of p periods* $\tilde{z} = \{z_1, \cdots, z_p\}$ *is sampled from a set* $\tilde{S}_z \subset \bigcup_1^p \{1, \cdots, N_z\}$. *Each element of* $\tilde{z}$ *controls the user's click behavior in T time steps. That is, each item of historical click behaviors subsequence* $\{x^{T\times(i-1)+1}, \cdots, x^{T\times i}\}_h$ *is sampled from the conditional distribution* $P(x|z_i)$. z_i *is interest hidden state in the i-th period. The number of elements in set* $\tilde{S}_z$ *is* N_S. *The label* y^* *is sampled from* $\{0, 1\}$.

The sampling process of behavior sequence is shown in Figure 2. It assumes that one interest hidden state can control a T time steps peroid, which is consistent with the actual situation for the user's interest always lasts for a while in the e-commerce scenario. Interest hidden of click behavior is a sequence $\tilde{z}$ of p periods, which is sample from a set $\tilde{S}_z$. Therefore, the final length of click behavior sequence length should be $T \times p$.

3.3 Analysis of Generalization Error Bound on Interest Delay Model

Section 3.2 models user behavior quantitatively through the interest delay model. In this subsection we will discuss generalization error of the DNNs on the data generated by the interest delay model. Which variable is related to generalization error? Theorem 1 will give a generalization error bound of the MLP with ReLU in definition 1 on the data sampled as the setting of Assumption 1.

Theorem 1. *The data sample represented as* $s = \{\{x^1, \cdots, x^{T\times p}\}_h, x_t, y^*\}$ *is generated from the way of Assumption 1. For the D layers MLP with ReLU and* $(Tp+1) \times d$ *input dimensions defined in Definition 1,* $f(\{x^1, \cdots, x^{T\times p}\}_h, x_t)$, *the loss function of MLP is* $f(l, s) = |f(\{x^1, \cdots, x^{T\times p}\}_h, , x_t) - y^*|$. *If there are N samples sampled from the way of Assumption 1 to train the MLP, with the probability* $1-\delta$, *the generalization error bound satisfies*

$$|E_s(l(f,s)) - \frac{1}{N}\sum_{i=1}^{N} l(f, s_i)| \leq \inf_r \{\sqrt{Tp+1}\|W\|_2^D r + l_M\sqrt{\frac{4N_z N_S(2R_{max}\sqrt{d}/r)^{d(Tp+1)} ln2 + 2ln(1/\delta)}{N}}\} \tag{2}$$

Among them, r is a parameter and will disappear in the infimum operation. $\|W\|_2$ *is the average of 2-norm of all parameter matrices in MLP. That is* $\|W\|_2 = \sum_{i=1}^{D} \|W_i\|_2 / D$. l_M *is the maximum value of* $l(f, s)$, *and* $l_M = 1$ *in Definition 1.* N_z *is the number of the interest domains and* N_S *is the number of the interest sequence set.* $R_{max} = \max_i \phi(domainP_i)$ *for all* $i = 1, \cdots, N_z$. d *is dimension of embedding vector.*

As long as the structrue of model and the training data are fixed, D, d, p, T, N, N_z and N_S are fixed too. According to Theorem 1, **generalization error will only be affected by the** $\|W\|_2$ **and** R_{max}. We will not discuss $\|W\|_2$ here for $\|W\|_2$ is affected by too many factors that it is difficult to analyze the relationship between $\|W\|_2$ and embedding layer. Reducing the whole scale of the embedding vectors seems to corroborate the effectiveness of applying a regularization term of embedding layer in some cases. However, it will also reduce the capacity and representation ability of embeddings. By individually reducing the radius of envelope circles of each interest domain, one could **make the items with the same interest domain closer in embedding space but maintain the distances among the items of different interest domains**, which can control the generalization error bound and maintain capacity and representation ability of CTR prediction model at the same time.

The proof of the Theorem 1 and the details of specific derivation process is in the supplementary material.

3.4 Prototype and discussion

Based on the theoretical analysis, we firstly propose and concentrate on a basic prototype. In this prototype, items in the same interest domain shared the same **central** embedding vectors and each item has its unique **residual** embedding vector with smaller scale. The final embedding vector of one item is the sum of its **central** and **residual** embedding vectors. By reducing the scale of the **residual** embedding, we can effectively reduce the distance of the items of the same interest domain in embedding space, which will reduce R_{max} in (2).

Assume there are I interest domains and H items totally. We define central embedding matrix as $\boldsymbol{C} \in \mathbb{R}^{I\times d}$. Each row of $\boldsymbol{C}$ is the central embedding vector of each interest domain. $\boldsymbol{P} \in \mathbb{R}^{H\times I}$ is the relationship matrix between items and interest domains. Each row of $\boldsymbol{P}$ is one-hot vector. If i-th item belongs to j-th interest domain, the element $P(i, j) = 1$ otherwise 0. $\boldsymbol{R} \in \mathbb{R}^{H\times d}$ is residual embedding matrix. Each row of $\boldsymbol{R}$ is the residual embedding vector of each item. The final embedding matrix $\boldsymbol{E} \in \mathbb{R}^{H\times d}$, and each row of $\boldsymbol{E}$ is the final embedding vector of each item. $\mathbb{E}$ is calculated as

$$\boldsymbol{E} = \boldsymbol{P}\boldsymbol{C} + \boldsymbol{R}. \tag{3}$$

For instance, in Figure 3, 8 embedding vectors are replaced into 2 central embedding vectors and 8 residual embedding vectors. With this structure, the envelope radius of embedding vectors in the same interest domain is bounded by the scale of residual embedding vectors.

However, the relationship matrix $\boldsymbol{P}$ is unknown and hard to obtain. In another word, it is unknown that which interest domain each item belongs to. Wrong $\boldsymbol{P}$ may bring worse generalization to deep CTR prediction model. It is necessary to ascertain the reasonable relationship between items and interest domain. Moreover, It is very likely that a item does not belong to only one interest domain. That is, the constraint relationship of central embedding could be soft, which means the central embeddings of items in the

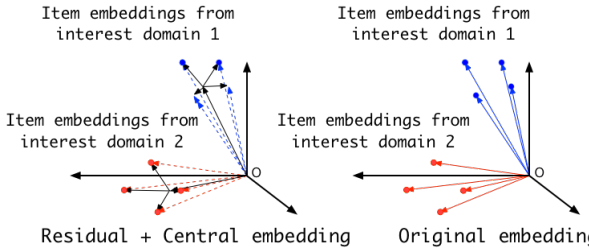


Figure 3: Sketch Map of the basic res-embedding prototype with 8 items and 2 interest domains ($H = 8$ and $I = 2$) in the 3-d embedding space.

same interest domain may not be exactly the same, but just more similar.

In order to solve this problem, we reexamine Assumption 1 and notice that one hidden interest state z will keep T time steps to affect user click behavior. Therefore, a conclusion that two items are more likely to be in the same interest domain if they appear more frequently in a short-term of user click behaviors can be deduce. From this conclusion, we define an item interest graph Z constructed by co-occurrence frequency of each item pair to describe similarity relation of interest domain among items.

4 METHOD

In the Section 3, there is a conclusion that the reducing the distance of items with the same interest domain in the embedding space is helpful to promote the generalization of CTR prediction model. However, **as our discussion in sub-section 3.4**, in the real scenario, the original prototype proposed in Section 3 has the problem that it is impossible to determine which interest domain each item belongs to for the detail information of interest domain of the is unknown. In this section, based on the theoretical analysis and the prototype, we construct an interest graph describing the similarity of interest domains between items, and propose a improved res-embedding base on the interest graph. As the evolutionary version of the proposed prototype, res-embedding contain central embedding part and residual embedding part. In central embedding part, Central embedding vector of each item is calculated through its adjacent items in the interest graph. In residual embedding part, each item owns an independent embedding vector.

4.1 Res-embedding

We still assume that there are H items totally. Res-embedding structure is parameterized by two trainable parameter matrices, central embedding basis matrix $\boldsymbol{C_b} \in \mathbb{R}^{H\times d}$ and the residual embedding matrix $\boldsymbol{R} \in \mathbb{R}^{H\times d}$. Each row vector of $\boldsymbol{C_b}$ is central embedding basis vector of corresponding item and Each row vector of $\boldsymbol{R}$ is residual embedding vector of corresponding item. The final embedding vector of one item is the sum of its central embedding vector and its residual embedding vector. The residual embedding vector of one item can directly read from $\boldsymbol{R}$ and central embedding vector of one item is derived from the linear combination of other items' central embedding basis vectors. We define the residual embedding matrix $\boldsymbol{W} \in \mathbb{R}^{H\times H}$ and the i-th row vector in $\boldsymbol{W}$ represents the linear combination coefficients of i-item about all items central embedding bases. The calculation process of final embedding matrix $\boldsymbol{E}$ is shown as follows

$$\boldsymbol{E} = \boldsymbol{W}\boldsymbol{C_b} + \boldsymbol{R}. \tag{4}$$

Algorithm 1 Graph construction

Input: user click behavior sequences set $B = \{b_1, \cdots, b_n\}$, window radius Δ
Output: Adjacent matrix Z

Initialize the all element of adjacent matrix Z as 0.
for i in $1 \cdots N$ **do**
  $b_i = \{g^i_1, \ldots, g^i_{m_i}\}$
  for g^i_c in $\{g^i_1, \ldots, g^i_{m_i}\}$ **do**
    $N_l = max(1, c - \Delta)$
    $N_r = min(c + \Delta, m_i)$
    for g_w in $\{g^i_{N_l}, g^i_{N_l+1}, \ldots, g^i_{N_r}\} \backslash g_c$ **do**
      $Z(g_c, g_w) \leftarrow Z(g_c, g_w) + 1$
Keeping the largest K values in each row of Z, and set the rest to 0
return Z

Each row vector of $\boldsymbol{E}$ is the embedding vector of corresponding item finally entered into the MLP.

According to the theoretical analysis, a reasonable linear combination matrix $\boldsymbol{W}$ should make the central embedding of the items with the same interest domain closer. In order to calculate a reasonable linear combination matrix $\boldsymbol{W}$, we utilize the information of item interest graph mentioned above. The central embedding of item should be a linear combination of the center embedding basis of items connected with it in the item interest graph. We construct a item interest graph with its adjacent matrix $Z \in \mathbb{R}^{H\times H}$ based on the co-occurrence frequency of items on historical click behavior sequence of all users according to the inference of Assumption 1. The construction process of the graph is illustrated in Algorithm 1. In Algorithm 1, the $B = \{b_1, \cdots, b_N\}$ is the set of user click behavior sequences of all N training samples, and $b_i = \{g^i_1, \ldots, g^i_{m_i}\}$ is the user click behavior sequence of i-th sample with m_i clicked items arranged in time sequence. g^i_j means the ID of j-th clicked item in b_i. Item ID is the index of the adjacent matrix, $\{1, 2, \cdots, H\}$.

The linear combination matrix $\boldsymbol{W}$ is decuded from adjacent matrix Z of item interest graph.

$$\boldsymbol{W} = g(Z) \tag{5}$$

function $g() : \mathbb{R}^{N\times N} \rightarrow \mathbb{R}^{N\times N}$ converts the adjcent matrix to a linear combination matrix with the same size.

The final central embedding of one item is composed of linear combination of the central embedding basis of its adjacent items in the interest graph Z. The linear combination coefficients w are determined by the connection weight of the graph.

For instance, to get the embedding vector of item i, the adjacent items list $\{i_1, i_2, i_3, i_4\}$is retrieved from the graph in Figure 4. We select the central embedding basis of items $\{i_1, i_2, i_3, i_4\}$ as the basis of the final central embedding of item i. The final central embedding vector of item i is weighted average by the weight w_1, w_2, w_3, w_4. The weights w_1, w_2, w_3, w_4 are obtained based on the connection weight f_1, f_2, f_3, and f_4 in the graph. Finally, the final embedding vector of item i is derived as the sum of its final central embedding vector and residual embedding vector.

We adopt three different forms of function g to fetch the linear combination coefficients matrix $\boldsymbol{W}$. The simplest one is the average function which means that all of the linear combination coefficients is the same:

$$g_{AVG}(Z) = \text{avg}_{\text{nz}}(I(Z > 0)). \tag{6}$$

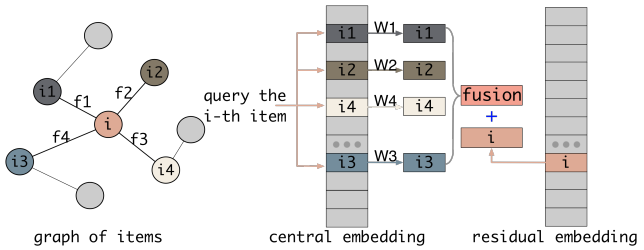


Figure 4: The left part of this figure is the items graph (Weighted undirected graph) constructed based on users' click behaviors. The weight between the each item in the graph are the co-occurrence frequency in a short-term period of users' click behaviors. When querying the embedding vector of item i, we queries the adjacent items i_1, i_2, i_3 and i_4 and the connection weights f_1, f_2, f_3 and f_4. We combine the central embedding basis vectors of these adjacent in weight w_1, w_2, w_3 and w_4 as the final central embedding vector linearly. Weights varies based on different function $g()$. Finally, the central embedding vector is added by residual embedding vector of i-th item.

The indicator function $I(Z > 0)$ means that each element of the F is changed to 1 if it is greater than 0, otherwise changed to 0. The avg_{nz} operation averages all the non-zero elements of every row vector in the matrix, and the 0 elements remain unchanged.

The second form of function borrows the idea from GCNs. Row normalization and column normalization are performed on adjacent matrix, and the linear combination weight is the connection weight. That is

$$g_{GCN}(\boldsymbol{Z}) = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{Z}\boldsymbol{D}^{-\frac{1}{2}}. \tag{7}$$

The degree matrix $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is calculated by row sum of adjacency matrix $\boldsymbol{Z}$.

Last form of function is the attention method. We need to introduce $\boldsymbol{C_b}$ to calculate the attention score as the linear combination weights, so the $g_{ATT}()$ is rewritten as $g_{ATT}(\boldsymbol{Z}, \boldsymbol{C_b})$

The $g_{ATT}(\boldsymbol{Z}, \boldsymbol{C_b})(i, j)$ means the i-th row and j-th column element of output of $g_{ATT}(\boldsymbol{Z}, \boldsymbol{C_b})$. $\cdot$ represents the inner product of two vectors. $\boldsymbol{C_b}(i, :)$ repesents the i-th row vector of matrix $\boldsymbol{C_b}$.

$$g_{ATT}(\boldsymbol{Z}, \boldsymbol{C_b})(i, j) = \frac{\exp\left(\boldsymbol{C_b}(i, :) \cdot \boldsymbol{C_b}(j, :)\right)}{\sum_{k\in\{k|Z(i,k)>0\}} \exp\left(\boldsymbol{C_b}(i, :) \cdot \boldsymbol{C_b}(k, :)\right)}, \tag{8}$$
$$(i, j) \in \{(i, j)|Z(i, j) > 0\}$$

Beyond averaging, GCNs and attention methods pay more attention on the importance of each adjacent item. Therefore, their performance is supposed to be better, which are reflected in the experiments.

Our optimization objective function is

$$\min_{\Theta, \boldsymbol{C_b}, \boldsymbol{R}} E_0(\Theta, \boldsymbol{C_b}, \boldsymbol{R}) + \lambda L(\boldsymbol{R}). \tag{9}$$

Θ is the parameter set of the deep CTR prediction model except for the embedding layer. $E_0(\Theta, \boldsymbol{C_b}, \boldsymbol{R})$ is the cross-entropy loss of CTR, and the l_2 regularization term $L(\boldsymbol{R})$ with the coefficient λ is added to bound the scale of residual embedding vectors. Proposition 1 in the appendix and related analysis will illustrate that the proposed res-embedding can reduce the distance among the goods belonging to the same interest domain in the embedding space and the goods, while maintaining the distance between the goods in different interest domains.

5 EXPERIMENTS

In this section, we perform a series of experiments around res-embedding. The generalization experiments verify the theoretical conclusion through training set decrement experiment and visualization experiments. Experiments on Amazon and Movielens datasets illustrate that the proposed embedding layers could enhance the performance of various deep CTR prediction models.

5.1 Datasets and Experimental Setup

Datasets	Users	Items	Categories	Clicks
Amazon-Electronics	192,403	63,001	801	1,689,188
Amazon-Books	603,669	367,984	1,602	8,898,041
MovieLens	138,493	27,278	21	20,000,263

Table 2: Statistical information of datasets in this paper. "Clicks" means click number of the dataset, and for Movielens dataset, it means the number of the ratings.

Amazon Datasets[11]: We utilize the product reviews and metadata from Amazon to validate performance of res-embedding. Electronics and Books subsets are used by our experiments. The statistical information of Electronics and Books is shown in Table 2. The complete user behaviors are $(g_1, \ldots, g_N)$. Our model is to predict the probability of one item reviewed in the $n+1$-th step under the first n reviewed item. For each user, there is a pair sharing the same historical behavior $(b_1, \ldots, b_n)$ with one positive and one negative sample. The target item of the positive sample is b_{n+1} and that of the negative sample is randomly sampled from all of the sample. Training and Testing datasets are divided by users.

MovieLens Datasets[10]: The statistical information of Electronics and Books is shown in Table 2. We choose the movie that users rate at a certain time as the target movie m_t and $n = 50$ movies $(m_1, ..., m_n)$ with a score of no less than 3 before this time as the user behavior sequence. Target movie rated more than 3 are labeled as positive, otherwise negative.$\{(m_1, \cdots, m_n), m_t, y\}$ is a CTR prediction sample.

For all models and datasets, we use Adam as the optimizer with exponential decay, in which learning rate is set as 0.1 and batch-size is set as 128. The regularization coefficient λ of the residual embedding vectors in res-embedding structure is set as 0.006. We select the windows redius $\Delta = 2$ and the $K = 8$ mentioned in Algorithm 1.

AUC[6] is adopted as criteria which measures the goodness of order by ranking all the samples with predicted CTR. We utilize res-embedding on several different deep models to validate the performance of res-embedding. These deep models are listed as follows:

Basic MLP: The basic MLP is the original MLP. It receive the embedding vector of target item and the sum of embedding vectors of the items clicked by users historically as the input. In this experiments, layers of MLP are set as 36×400×120×2.

PNN[16]: PNN is an variant of the basic MLP, and the difference between them is that the PNN receives the product the sum embedding vectors and target product as the an extra input vector. The layers of PNN are set as 54×400×120×2.

DIN[4]: DIN introduces the attention mechanism into the CTR model. It extracts several interest vectors of the users from their historical behaviors. The embedding vector of the target item is

extracted into an interest vector and adopt the attention operation with the interest vector of the behavior sequence, then input the result to the basic MLP.

5.2 Validation of Theorem 1

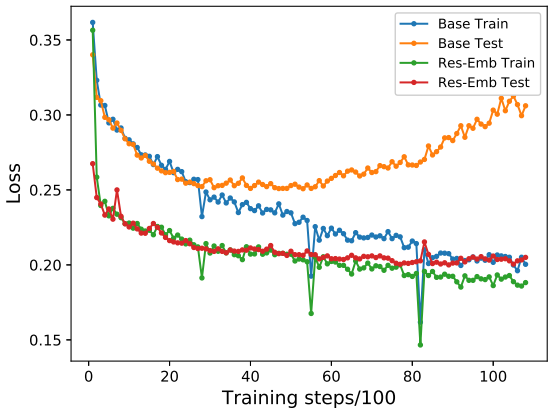


Figure 5: Compare the original embedding and the res-embedding with attention in DIN model for electronics dataset, The horizontal axis is the number of training steps/100, and the longitudinal axis is the loss function of the CTR model. The lines of different colors represent the loss function curves of different models under different data. For example, "Base Train" represents the line of the original embedding structure under the training dataset.

We compare the train-test line of the original embedding layer and that of the res-embedding with attention in DIN model for electronics dataset. As shown in Figure 5, under the original embedding structure, the CTR task has a very strong over fitting phenomenon. That is, as the number of training steps increases, the gap between training and test loss group quite quickly. When the res-embedding is adopt as the input of model, this error has been greatly reduced, which indicates that the generalization error bound of CTR prediction model is effectively controled by res-embedding structure.

Aiming to verify the generalization performance promotion of res-embedding further, we compared original embedding structure with res-embedding in DIN model under the decay datasets. We reduce scale of the training set of Books dataset and utilize them to train the deep CTR model and the proportion of training dataset to whole training dataset decays from 90% to 10%. In Figure 6, we find that only adapting 20% or 30% dataset to train with res-embedding could obtain the similar performance to using the whole training set. This phenomenon demonstrates that res-embedding structure can partly relieve the problem of generalization performance loss due to insufficient training data.

Theorem 1 tells us that the group aggregating embedding vectors are helpful to improve generalization performance. In order to prove the consistency between res-embedding and theoretical analysis, we visualize embedding vectors of the top 1000 highest frequency items in the deep CTR model with t-sne visualization methods. Figure 7(a) shows visualized embedding vectors of original embedding structure. It can be seen that embedding vectors are randomly

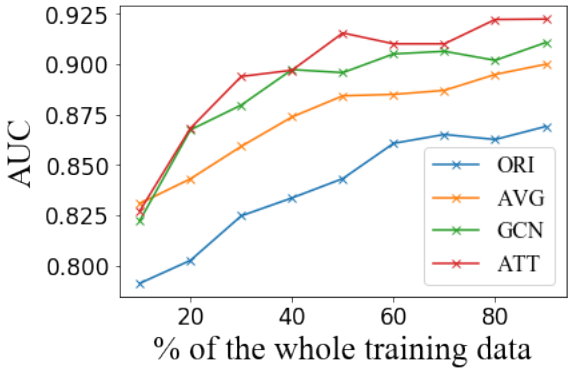
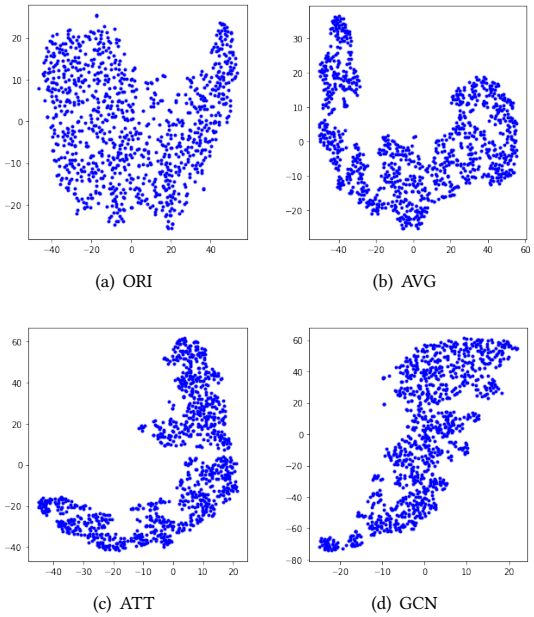


Figure 6: The proportion of to whole training set varies from 10% to 90%. ORI, AVG, GCN, ATT represent the original embedding, res-embedding with average, GCN, and attention fusion mechanism.



(a) ORI (b) AVG

(c) ATT (d) GCN

Figure 7: The blue dots denote embedding vectors of the top 1000 high frequency items. ORI represents the original embedding structure, AVG, ATT and GCN are res-embedding with three different fusion mechanisms.

located in the whole region. While in Figure 7(b), 7(d), and 7(c), the density of embedding distribution is not uniform, which is different from the original embedding structure. Embedding vectors in some parts are dense and in other parts are sparse. In another word, embedding vectors are more aggregated locally, which is consistent with our theory.

5.3 Total Promotion of Res-embedding

Table 3 shows the experimental results on two Amazon Datasets and MovieLens Datasets. All the results contain the mean AUC and its variance under experiments repeated for 5 times. Comparing with the original embedding structure with L2-regularization term and without it, we find that the performance has been improved after adding regularization terms, but the promotion is not so large. As mentioned in the theoretical analysis, it is because reducing

Methods	AUC(Electronics)	AUC(Books)	AUC(Movielens)
MLP	0.8558±0.0004	0.8833±0.0013	0.7242±0.0002
MLP+R	0.8649±0.0008	0.8802±0.0021	0.7337±0.0003
MLP+Skip	0.8710±0.0006	0.8964±0.0013	0.7275±0.0003
MLP+n2v	0.8756±0.0005	0.8973±0.0010	0.7292±0.0002
MLP+sms	0.8695±0.0008	0.9172±0.0002	0.7270±0.0003
MLP+AVG	0.8857±0.0008	0.8977±0.0018	0.7326±0.0001
MLP+GCN	**0.8917±0.0006**	0.9240±0.0028	**0.7389±0.0003**
MLP+ATT	0.8907±0.0011	**0.9295±0.0000**	0.7385±0.0004
PNN	0.8601±0.0005	0.8930±0.0017	0.7311±0.0004
PNN+R	0.8671±0.0002	0.8932±0.0009	0.7363±0.0002
PNN+Skip	0.8771±0.0006	0.9006±0.0012	0.7313±0.0004
PNN+n2v	0.8817±0.0004	0.9041±0.0013	0.7327±0.0004
PNN+sms	0.8758±0.0002	0.9182±0.0015	0.7330±0.0005
PNN+AVG	0.8930±0.0003	0.9052±0.0009	0.7449±0.0004
PNN+GCN	0.9042±0.0006	0.9325±0.0023	**0.7458±0.0003**
PNN+ATT	**0.9057±0.0003**	**0.9357±0.0011**	0.7449±0.0006
DIN	0.8635±0.0000	0.8971±0.0003	0.7288±0.0001
DIN+R	0.8752±0.0000	0.8902±0.0002	0.7360±0.0001
DIN+Skip	0.8786±0.0001	0.9048±0.0001	0.7303±0.0000
DIN+n2v	0.8832±0.0001	0.9078±0.0001	0.7319±0.0000
DIN+sms	0.8802±0.0002	0.9285±0.0001	0.7275±0.0001
DIN+AVG	0.8981±0.0003	0.9052±0.0003	0.7383±0.0002
DIN+GCN	0.9090±0.0003	0.9372±0.0001	**0.7429±0.0001**
DIN+ATT	**0.9106±0.0003**	**0.9404±0.0003**	0.7412±0.0000

Table 3: The AUC results on amazon datasets (Electronics and Books) and movielens dataset. model without any addition represents the deep model with original embedding layer. R represents the deep model with original embedding layer with regularization constraints. The AVG, GCN, ATT represent the fusion mechanism under res-embedding structure: average, GCN, attention. The MLP+Skip[1],n2v[8] and sms[15] are 3 pretrain methods, means Skip-gram, node2vec, and siamese auxiliary network.

the scale of embedding as a whole limits the expressive power of embedding while reducing R_{max}. Res-embedding structure promotes the AUC of the deep model under the CTR task. Compared with some other embedding methods like Skip-gram, node2vec and siamese auxiliary network, res-embedding also acheive better performance. The average function promotes less than the attention and GCN function. The main reason is that the average function only aggregates the nearest neighbors indiscriminately while the other two methods take account of the different characteristics of different neighboring items.

5.4 Comparison of parameter quantities

In Table 3, we ensure the embedding space of comparative method and res-embedding are the same. However, res-embedding uses twice parameters compared with the original embedding structure, so we are supposed to discuss the effect of changing of the parameters and embedding dimensions on performance. As shown in Table 4, with the increase of embedding dimension(compared with DIN models with 18 dim and DIN model with 36dim), the performance of DIN is improved, which indicates that increasing the dimension of embedding will at least not loss performance. Focusing on DIN models with 36dim, DIN+AVG model, DIN+GCN model and DIN+ATT model with 18dim, which means that res-embedding has the same size of parameters with original embedding structure and has smaller dimensions, our method is still greatly improved. In

Models	Dim of embedding	AUC(Electronics)
DIN	18	0.8635
DIN	36	0.8751
DIN+AVG	18	0.8981
DIN+AVG	36	0.9105
DIN+GCN	18	0.9090
DIN+GCN	36	0.9115
DIN+ATT	18	0.9106
DIN+ATT	36	0.9129

Table 4: For the electronic dataset, the AUC of the DIN model uses different embedding methods under different dimensions of embedding vector. Dim of embedding means the dimension of the embedding vectors.

summary, although res-embedding uses twice as many parameters as baseline, the improvement of res-embedding is due to the improvement of the method itself, not the increase of the parameters. Even if the parameters of baseline are increased to the same as res-embedding, and the embedding dimension of baseline is also higher than res-embedding, res-embedding still has a significant improvement.

5.5 Effectiveness of residual part

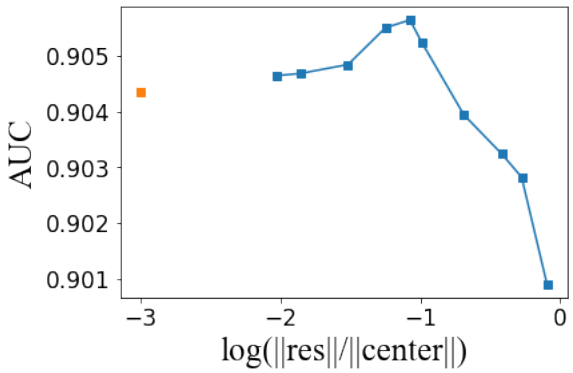


Figure 8: The relationship between the scale of the residual part and the AUC based on the electronics dataset. The orange dot means the situation that the residual embedding disappears. The deep model is DIN with res-embedding in attention.

In this subsection, we will verify the effectiveness of residual embedding vectors. The experimental results are shown in Figure 8, which is based on the electronics dataset, res-embedding in attention technology with DIN. In experiments, we increase the scale of residual embedding by controlling the regularization coefficient of residual embedding matrix. As the scale ratio between residual and central embedding vector varies from large to small, the performance is gradually improved. When the scale ratio between residual and central embedding is reduced apoach to 1:10, the AUC criteria reaches its peak value. The promotion of performance is consistent with our theory for the smaller scale of residual embedding means smaller envelope radius of items with the same interest domain in the embedding space, which will improve the generalization performance. However, while continuing to reduce scale

of the residual to disappear, the AUC criteria drops slightly, which may be attributed that over-small scale of the residual embedding vector reduces the representation ability of the embedding layer. In summary, the residual part can be considered as an adjuster to balance generalization and representation ability.

6 CONCLUSION

In this paper we propose a novel res-embedding structure to improve the generalization of deep CTR prediction task. We theoretically proved that the generalization error of the depth CTR model is related to the aggregation of embedding packets. We use the user historical behaviors to construct the graph of items and calculate the central embedding based on this graph. Several fusion methods are adopt to generate central embedding. Residual embedding is used to control the degree of embedding aggregation. In the experiment, we validate the effectiveness of the residual structure on the CTR task public datasets. We also observed a significant improvement in generalization performance, and the visualization experiments also verified the changes in embedding structure. The graph among items is pre-generated so far, which may be affected by changes of sample distribution. In the future, we will consider using an end-to-end approach to dynamically update the graph.

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*. IEEE, 1–6.

[2] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems* 14, 6 (2002), 585–591.

[3] Yoshua Bengio, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3, 6 (2003), 1137–1155.

[4] Heng Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, and Mustafa Ispir. 2016. Wide & Deep Learning for Recommender Systems. (2016), 7–10.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *ACM Conference on Recommender Systems*. 191–198.

[6] Tom Fawcett. 2005. An introduction to ROC analysis. *Pattern Recognition Letters* 27, 8 (2005), 861–874.

[7] Kun Gai, Xiaoqiang Zhu, Han Li, Kai Liu, and Zhe Wang. 2017. Learning Piecewise Linear Models from Large Scale Data for Ad Click Prediction. (2017).

[8] Aditya Grover and Jure Leskovec. 2016. node2vec:Scalable Feature Learning for Networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 855–864.

[9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. (2017), 1725–1731.

[10] F. Maxwell Harper and Joseph A. Konstan. 2015. *The MovieLens Datasets: History and Context.* ACM. 19 pages.

[11] Ruining He and Julian Mcauley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *International Conference on World Wide Web*. 507–517.

[12] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. 2018. Generalization in Deep Learning. (2018).

[13] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. (2016).

[14] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[15] Mehul Parsana, Krishna Poola, Yajun Wang, and Zhiguang Wang. 2018. Improving Native Ads CTR Prediction by Large Scale Event Embedding and Recurrent Networks. (2018).

[16] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2017. Product-Based Neural Networks for User Response Prediction. In *IEEE International Conference on Data Mining*. 1149–1154.

[17] Steffen Rendle. 2011. Factorization Machines. In *IEEE International Conference on Data Mining*. 995–1000.

[18] Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290, 5500 (2000), 2323–2326.

[19] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. 2016. Deep Crossing:Web-Scale Modeling without Manually Crafted Combinatorial Features. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 255–262.

[20] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE:Large-scale Information Network Embedding. 1067–1077.

[21] Mikhail Trofimov, Sumit Sidana, Oleh Horodnitskii, Charlotte Laclau, Yury Maximov, and Massih-Reza Amini. 2017. Representation Learning and Pairwise Ranking for Implicit and Explicit Feedback in Recommendation Systems. *CoRR* (2017).

[22] Nicolas Usunier, Massih-Reza Amini, and Patrick Gallinari. 2006. Generalization error bounds for classifiers trained with interdependent data. In *Advances in neural information processing systems*. 1369–1376.

[23] Huan Xu. 2012. Robustness and generalization. *Machine Learning* 86, 3 (2012), 391–423.

[24] Valentina Zantedeschi, Rémi Emonet, and Marc Sebban. 2016. Metric learning as convex combinations of local models with generalization guarantees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1478–1486.

[25] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2018. Deep Interest Evolution Network for Click-Through Rate Prediction. (2018).

[26] Guorui Zhou, Chengru Song, Xiaoqiang Zhu, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2017. Deep Interest Network for Click-Through Rate Prediction. (2017).