Regularized Adversarial Sampling and Deep Time-aware Attention for Click-Through Rate Prediction

Yikai Wang[1*] Liang Zhang[2*] Quanyu Dai[3] Fuchun Sun[1‖] Bo Zhang[2]
Yang He[2] Weipeng Yan[2] Yongjun Bao[2]
[1] Department of Computer Science and Technology, Tsinghua University
Beijing National Research Center for Information Science and Technology (BNRist)
[2] JD.COM [3] The Hong Kong Polytechnic University
{wangyk17@mails.,fcsun@}tsinghua.edu.cn,csqydai@comp.polyu.edu.hk
{zhangliang16,zhangbo35,landy,Paul.yan,baoyongjun}@jd.com

ABSTRACT

Improving the performance of click-through rate (CTR) prediction remains one of the core tasks in online advertising systems. With the rise of deep learning, CTR prediction models with deep networks remarkably enhance model capacities. In deep CTR models, exploiting users' historical data is essential for learning users' behaviors and interests. As existing CTR prediction works neglect the importance of the temporal signals when embed users' historical clicking records, we propose a time-aware attention model which explicitly uses absolute temporal signals for expressing the users' periodic behaviors and relative temporal signals for expressing the temporal relation between items. Besides, we propose a regularized adversarial sampling strategy for negative sampling which eases the classification imbalance of CTR data and can make use of the strong guidance provided by the observed negative CTR samples. The adversarial sampling strategy significantly improves the training efficiency, and can be co-trained with the time-aware attention model seamlessly. Experiments are conducted on real-world CTR datasets from both in-station and out-station advertising places.

CCS CONCEPTS

• **Information systems → Learning to rank**; **Recommender systems**; **Retrieval effectiveness**;

KEYWORDS

CTR Prediction; Time-aware Attention; Adversarial Sampling

ACM Reference Format:
Yikai Wang, Liang Zhang, Quanyu Dai, Fuchun Sun, Bo Zhang, Yang He, Weipeng Yan, and Yongjun Bao. 2019. Regularized Adversarial Sampling and Deep Time-aware Attention for Click-Through Rate Prediction. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3357936

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3357936

1 INTRODUCTION

Online advertising is widely used for delivering promotional products to users, due to its low ads displaying cost, easy customization, easy deployment, large coverage and fast delivery speed. In online advertising, cost per click (CPC) is one of the dominant methods, in which the advertisers pay for each click on their ads. Click-though rate (CTR) prediction, with the objective to estimate the probability of users' clicking behaviors, can directly influence the performance of both bidding and ranking in CPC advertising systems.

Improving users' CTR prediction performance in online advertising remains a hot research topic. In the early stage, FM[24] uses cross terms of user features and item features aiming to capture their combination relations. Recent years, inspired by the powerful capability of deep learning, deep CTR models like Deep Crossing[29], Wide&Deep[7], DeepFM[12] extend early works by replacing the transformation functions with complex networks, which enhance the model capacities. In these works, users' historical behaviors are integrally converted to low-dimensional embeddings without exploration of each individual historical item. Thus these models are limited to represent users' rich historical behaviors.

To further improve the performance of CTR prediction, a crucial part is to learn users' preferences on the basis of their past interactions with items, which are detailedly recorded over time. To do this, current works mainly use attention-based[3, 19, 25] models to exploit users' historical clicking records. DIN[36] and DIEN[35] capture users' relative interests by exploiting users' historical clicking items using the attention mechanism.

However, these attention-based models do not explicitly use the clicking temporal signals of users' historical data. The temporal signals, on the one hand, can reflect the users' periodic behavior trends. For example, a user is likely to be more active after the payday of each month, and tends to buy various seasonal clothes in various months of each year. On the other hand, temporal signals can express the extent of the influence of each historical item on the target recommended item. Users' behaviors present rich changes with time, as past interests may fade away and new interests may emerge. In order to take advantage of such temporal information in users' historical clicking records, we propose a time-aware attention model for CTR prediction. The time-aware attention model contains absolute temporal signals for periodicity representation and relative temporal signals for temporal relation representation. Comparison

[*] Both authors contribute equally to this research.
[‖] Fuchun Sun (fcsun@tsinghua.edu.cn) is the corresponding author.

results show that in CTR prediction tasks, the proposed time-aware attention model outperforms the current works by a large margin.

CTR prediction essentially aims to classify between the minority positive samples and the majority negative samples, and suffers from a serious data imbalance problem (such as 1:100). Adopting random sampling or down sampling on the negative samples can alleviate the classification imbalance to some extent[13]. However, these sampling strategies may miss informative negative samples, which are overwhelmed by the huge amounts of the total negatives. GAN-based[11] sampling, which is a state-of-the-art method in recommender systems, improves data efficiency by seeking competitive nonpositive samples for each positive sample to promote adversarial training, where the nonpositive samples stand for arbitrary combinations of users and items that are not positive and are often non-interactive. IRGAN[27] and AdvIR[22] both apply adversarial sampling models in the information retrieval area, and the authors demonstrate several applications containing item recommendation. [28] describes recommendation-specific adversarial sampling method in detail and acquires effective performance. A same weakness of the current adversarial sampling methods in recommender systems is that, they can only select the nonpositive samples to train against the positive samples, thus these GAN sampling works are not applicable to CTR prediction tasks where observed negative samples should be considered.

In CTR prediction tasks, each item is carefully selected and exposed to a user by the advertisement. Whether the exposed item acquires a click or not, it has already practically interacted with the user, and thus can provide us with strong user-item information. Recommended items that fail to be clicked, are treated as items that the user dislike, and these user-item pairs are labeled as negative samples. These observed negative samples are more negative compared with previously mentioned nonpositive samples, and thus can provide more guidance for pairwise learning. In order to utilize such guidance, we propose a regularized adversarial sampling model, which contains a distance-based discriminator and a probability-based generator. We reframe the adversarial sampling process in view of imbalanced classification and indicate that the selected negative sample needs to be competitive among all the negatives as well as correlative to the given positive sample. Therefore in the discriminator, we design a feedback that contains not only the sample score but also a regularization term. The regularization is a weighted Euclidean distance between the embeddings of the positive and negative samples calculated in the discriminator. The designed feedback will help the generator select proper negative samples that can promote the adversarial training. The difference between the adversarial sample structures of the current works for common recommender systems and ours specifically for CTR prediction tasks is illustrated in Figure 1.

Our regularized adversarial sampling strategy is specifically designed for CTR prediction, and can be seamlessly integrated with the proposed time-aware attention model. Besides, we provide a CTR calibration method to further deal with an over-estimation issue of the absolute CTR values in the negative sampling process.

The contributions of our works are summarized as follows:

(1) We propose a time-aware attention model for CTR prediction tasks, which can represent the users' periodic behaviors and the temporal relations between items, by taking absolute and relative temporal signals into consideration.
(2) We design a regularized adversarial sampling strategy for CTR prediction, which can use the strong information of the observed negative samples. The adversarial sampling model can be co-trained with the time-aware attention model.
(3) We conduct experiments on real-world CTR data from both in-station and out-station advertising places, and acquire comparison relative CTR results with recent state-of-the-art works. We obtain accurate absolute CTR results with the help of CTR calibration. We also provide an in-depth model exploration and sensitivity analysis of hyperparameters.

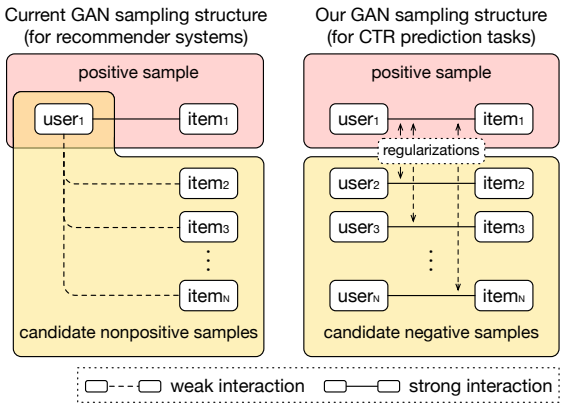


Figure 1: GAN-based sampling structures comparison

The rest of the paper is organized as follows. We propose the time-aware attention model in Section 2 and the regularized adversarial sampling strategy in Section 3. We provide experimental details, comparison results and in-depth analysis in Section 4. We summarize the related works in Section 5. Finally we conclude our works in Section 6.

2 TIME-AWARE ATTENTION

2.1 Data Description

Historical user-item interactions are largely recorded and play a pivotal role in many real-world CTR prediction tasks. In our model, a single sample is composed of a pair of a user and a target item (recommended item), where the user's information is represented by a series of user's recent L clicking items, and the target item is selected and exposed to the user by the advertisement. Each item has three parts of features, described as follows:

(1) Raw features embedding, denoted as $\boldsymbol{e}^r$, has 50 dimensions and is obtained by a pre-training process on the displayed image of the item using Telepath[30]. Telepath acts as a good feature extractor to our model as it extracts the key features with deep CNNs[15] by considering relations between the users' historical records and the target item.
(2) Interaction time, denoted as t, refers to the clicking time for historical items or the exposing time for the target item, containing an absolute time t^a in seconds, a month index t^m, a week index t^w, a day index t^d and an hour index t^h.

(3) Category index (cid3), denoted as c, is a single scalar value representing the category that the item belongs to.

In summary, each sample in our CTR data contains $L+1$ items in total, including L historical clicking items with features $(\boldsymbol{e}_l^r, t_l, c_l), l = 1 \cdots L$, and a target item with features $(\boldsymbol{e}_0^r, t_0, c_0)$.

Besides, each sample has several auxiliary features that represent the attributes of the user such as the gender, age, etc.

2.2 Time-aware Embedding Structure

The structure of the sample embedding model is illustrated in Figure 2. The sample embedding (blue block in Figure 2) is a concatenation of a history embedding, a (target) item embedding and an auxiliary embedding.

The structure of the item embedding is a nonlinear mapping to features $(\boldsymbol{e}_0^r, t_0, c_0)$ of the target item. The absolute time t_0 is spilt into t_0^m, t_0^w, t_0^d and t_0^h as defined in Section 2.1. The four time signals, as well as the cid3 c_0, are five scalars followed by the same network structure (dark green block in Figure 2) with independent weights. These scalars are converted to one-hot embeddings and then sent to a fully-connected layer for dimensional reduction. The five low dimensional embeddings together with the raw features embedding $\boldsymbol{e}_0^r$ are concatenated and converted using fully-connected layers to a d-dimensional item embedding, denoted as $\boldsymbol{e}_0^i$. The structure of the item embedding is formulated as follows:

$$\begin{aligned} \boldsymbol{e}_0^c &= \mathcal{R}\big(\boldsymbol{W}_c O(c_0)\big)\,, \\ \boldsymbol{e}_0^t &= \mathcal{R}\big(concat\big(\boldsymbol{W}_m O(t_0^m), \boldsymbol{W}_w O(t_0^w), \boldsymbol{W}_d O(t_0^d), \boldsymbol{W}_h O(t_0^h)\big)\big)\,, \\ \boldsymbol{e}_0^i &= \mathcal{F}\big(concat\big(\boldsymbol{e}_0^r, \boldsymbol{e}_0^c, \boldsymbol{e}_0^t\big)\big)\,, \end{aligned} \tag{1}$$

where $O(\cdot)$ is a projection of the one-hot embedding; $\mathcal{R}(\cdot)$ is a ReLU function; $\mathcal{F}(\cdot)$ is a series of fully-connected layers; $\boldsymbol{W}_*$ are trainable matrixes for dimensional reduction; $\boldsymbol{e}_0^c$ and $\boldsymbol{e}_0^t$ represent the embeddings of the cid3 and the absolute time respectively.

For L historical items in a sample, we get their corresponding item embeddings using the same network structure (light green block in Figure 2) as the embedding structure of the target item. Denote these item embeddings as $\boldsymbol{e}_l^i \in \mathbb{R}^d, l = 1 \cdots L$. To capture the sequential information, we apply GRU mechanism[8] to the historical item embeddings, which is formulated as:

$$\begin{aligned} \boldsymbol{r}_l &= \sigma(\boldsymbol{W}_{er}\boldsymbol{e}_l^i + \boldsymbol{W}_{hr}\boldsymbol{h}_{l-1} + \boldsymbol{b}_r)\,, \\ \boldsymbol{z}_l &= \sigma(\boldsymbol{W}_{ez}\boldsymbol{e}_l^i + \boldsymbol{W}_{hz}\boldsymbol{h}_{l-1} + \boldsymbol{b}_z)\,, \\ \widetilde{\boldsymbol{h}}_l &= tanh\big(\boldsymbol{W}_{ec}\boldsymbol{e}_l^i + \boldsymbol{W}_{hc}(\boldsymbol{r}_l \odot \boldsymbol{h}_{l-1}) + \boldsymbol{b}_c\big)\,, \\ \boldsymbol{h}_l &= (1 - \boldsymbol{z}_l) \odot \boldsymbol{h}_{l-1} + \boldsymbol{z}_l \odot \widetilde{\boldsymbol{h}}_l\,, \end{aligned} \tag{2}$$

where $\sigma(\cdot)$ is a sigmoid function; $\boldsymbol{h}_l \in \mathbb{R}^h$ is the hidden state; $\boldsymbol{r}_l$ is the reset gate controlling the input of the former hidden state $\boldsymbol{h}_{l-1}$; $\boldsymbol{z}_l$ is the update gate controlling the update ratio; $\boldsymbol{W}_{**}$ and $\boldsymbol{b}_*$ are trainable variables.

For exploration of each individual historical item, we apply the attention mechanism to the hidden states of GRU, by assigning proper weights to various hidden states. Instead of using the standard attention, given the speciality of CTR prediction, we provide the attention model with d-dimensional relative time embeddings. The relative time, specified in seconds, is an attribute of each historical clicking item representing the interval between its clicking time and the exposing time of the target item. The relative time embedding of the lth historical item, denoted as $\boldsymbol{e}_l^t$, is calculated as:

$$\begin{aligned} \boldsymbol{e}_l^t[2j] &= sin\big((t_0 - t_l)/10000^{2j/d}\big)\,, \\ \boldsymbol{e}_l^t[2j+1] &= cos\big((t_0 - t_l)/10000^{2j/d}\big)\,, \end{aligned} \tag{3}$$

where $2j$ and $2j+1$ are the indexes; t_0 and t_l are the absolute time in seconds of the target item and the lth historical item respectively; the coefficient 10000 is referring to the positional encodings in [26].

In the attention module, the weighted factor a_l of the lth hidden state $\boldsymbol{h}_l$ is formulated as:

$$\begin{aligned} u_l &= \boldsymbol{v}^\top tanh(\boldsymbol{W}_h\boldsymbol{h}_l + \boldsymbol{W}_i\boldsymbol{e}_0^i + \boldsymbol{W}_t\boldsymbol{e}_l^t)\,, \\ a_l &= softmax(u_l)\,, \end{aligned} \tag{4}$$

where u_l is an intermediate variable; $\boldsymbol{e}_0^i \in \mathbb{R}^d$ is the item embedding of the target item; $\boldsymbol{e}_l^t \in \mathbb{R}^d$ is the relative time embedding; $\boldsymbol{W}_h \in \mathbb{R}^{v \times h}$, $\boldsymbol{W}_i, \boldsymbol{W}_e \in \mathbb{R}^{v \times d}$, $\boldsymbol{v} \in \mathbb{R}^v$ are trainable variables.

The history embedding $\boldsymbol{e}^h$ of the sample, which is the output of the attention module, is a linear weighted sum formulated as:

$$\boldsymbol{e}^h = \sum_{l=1}^{L} a_l \boldsymbol{h}_l\,. \tag{5}$$

The auxiliary embedding $\boldsymbol{e}^a$ is a nonlinear mapping of the auxiliary features described in Section 2.1 using fully-connected layers.

Finally the sample embedding $\boldsymbol{e}^s$ (blue block in Figure 2) is a concatenation of the history embedding $\boldsymbol{e}^h$, the (target) item embedding $\boldsymbol{e}_0^i$ and the auxiliary embedding $\boldsymbol{e}^a$, as follows:

$$\boldsymbol{e}^s = concat\big(\boldsymbol{e}^h, \boldsymbol{e}_0^i, \boldsymbol{e}^a\big)\,. \tag{6}$$

After a user browsing through a recommended item, the user-item sample is labeled as positive or negative based on whether the user clicks the item or not. Given the labels, our model can be learned by minimizing either a pointwise or a pairwise loss. The pointwise loss function is a cross-entropy loss calculated as:

$$\mathcal{L}_{point} = \sum_{s \in \mathcal{T}} \log \sigma\big(f\big(\boldsymbol{e}^s(s)\big)\big) + \sum_{s \in \mathcal{T}'} \log\big(1 - \sigma\big(f\big(\boldsymbol{e}^s(s)\big)\big)\big)\,, \tag{7}$$

where $\mathcal{T}$ and $\mathcal{T}'$ are the sets of the whole positive and the negative samples respectively; $f(\cdot)$ is a score function that converts the sample embedding to a scalar score; $\sigma(\cdot)$ is a sigmoid function.

The pairwise loss function is a marginal hinge loss calculated as:

$$\mathcal{L}_{pair} = \sum_{s \in \mathcal{T}, s' \in \mathcal{T}'} \big[-f\big(\boldsymbol{e}^s(s)\big) + f\big(\boldsymbol{e}^s(s')\big) + \gamma\big]_+\,, \tag{8}$$

where $[\cdot]_+ = \max(0, \cdot)$ is the hinge function, and γ is the margin.

In CTR prediction tasks, the size of the positives $|\mathcal{T}|$ is far smaller than the size of the negatives $|\mathcal{T}'|$, and thus both loss functions suffer from the imbalanced problem. Because of this, usual works conduct a negative sampling process instead of directly using the whole negative samples. In the next section, to improve the performance of CTR prediction, we propose a GAN-based negative sampling strategy which is adaptable for the both loss functions. We take the pairwise loss for example and provide detailed derivations.

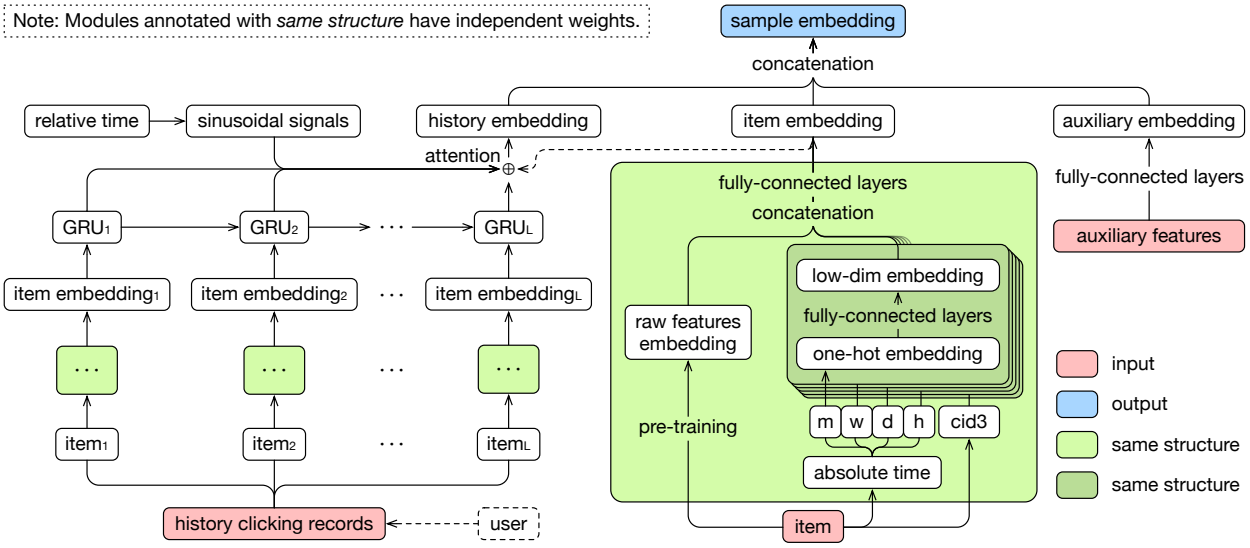


Figure 2: Structure of the time-aware attention model

3 REGULARIZED ADVERSARIAL SAMPLING

In this section we introduce our adversarial negative sampling model which is specifically designed for CTR prediction. The model contains a distance-based discriminator and a probability-based generator. The proposed adversarial sampling model can make use of the guidance brought by the observed negative samples.

3.1 Sampling Strategy

The sampling model trains a pairwise loss which is similar to Eq. (8). Specifically, at each step, a positive sample s and a negative sample s' are embedded using a time-aware attention module, and are both sent to the discriminator D. Networks in the discriminator convert the sample embeddings into scores, denoted as $f_D(\boldsymbol{e}_D^s(s))$ and $f_D(\boldsymbol{e}_D^s(s'))$ respectively. Then the optimizer in the discriminator updates to increase the gap between the scores. The score f_D measures the attraction of the item to the user. Therefore positive samples tend to have higher scores than the negatives.

When given a positive sample in each training step, a negative sample needs to be selected for training the pairwise loss function in the discriminator. One straightforward way is to uniformly choose a negative sample from all the negatives, and we call this method *uniform sampling*. Instead of using uniform sampling, we are seeking a better sampling strategy to enhance training efficiency and thus acquire better CTR prediction performance.

Supposing for a given positive sample s, a negative sample s' is sampled following a distribution conditioned on s. In our adversarial sampling model, We use a generator G to fit this conditional distribution, denoted as $p_G(s'|s)$. In the generator, we adopt another time-aware attention model for each sample to get the sample embedding, denoted as $\boldsymbol{e}_G^s$. The attention model in the generator shares the same structure with the attention model in the discriminator, but has independent network weights. The conditional distribution $p_G(s'|s)$ is expressed by an explicit union function of $\boldsymbol{e}_G^s(s)$ and $\boldsymbol{e}_G^s(s')$, which will be analyzed in later part of this section.

Referring to Eq. (8), the objective of the discriminator can be formulated as minimizing the following hinge loss function:

$$\mathcal{L}_D = \sum_{s\in\mathcal{T}} \left[-f_D\left(\boldsymbol{e}_D^s(s)\right) + f_D\left(\boldsymbol{e}_D^s(s')\right) + \gamma \right]_+, \; s' \sim p_G(s'|s), \quad (9)$$

where $\mathcal{T}$ is the set of the positive samples. Minimizing $\mathcal{L}_D$ tends to increase $f_D(\boldsymbol{e}_D^s(s))$ and decrease $f_D(\boldsymbol{e}_D^s(s'))$ to ensure a gap γ.

In order to design a better sampling strategy $p_G(s'|s)$, we take two aspects into consideration. Firstly, at a training step, if the selected negative sample s' has a low score, the gap between $f_D(\boldsymbol{e}_D^s(s))$ and $f_D(\boldsymbol{e}_D^s(s'))$ may close to or already larger than γ before any update, which leads to a useless training step. This will cause a low data efficiency and therefore affect the performance. Secondly, CTR prediction is a largely imbalanced classification problem. One-sided selection[4], a typical neighbor-based imbalanced classification approach, focuses on the boundary negatives and removes redundant or interferential majorities, achieving a better classification performance. Inspired by this, we consider a negative sample in CTR data to be redundant regarding to a given positive sample, if the negative sample locates too far away from the positive sample in some certain embedding spaces. The optimizer will acquire an unsatisfactory performance if it pays too much attention on the redundant negative samples, even though they have high scores.

To sum up, when given a positive sample s, a negative sample s' provided by the generator should have two desirable properties:

(1) Competitive: s' should have a high score, so as to be a strong impetus when training the discriminator.
(2) Correlative: Enough similarity needs to be ensured between the embeddings of s and s'. We experimentally find that using Euclidean distances on the embeddings calculated in the *discriminator embedding space* works better.

Detailed experimental results in subsection 4.4 compare the individual performance of either one of the properties and the overall performance with a good cooperation of the two properties.

We visualize the influence to be correlative in Figure 3, where the color shade reflects the score of the sample. A sample with darker shade in either positive region or negative region is more tend to be positive in the view of the discriminator. In each sub-figure, a negative sample with a high score is selected regarding to a given positive sample. The difference is, the first sampling seeks a tighter and more effective decision boundary compared with the second, giving the credit to the correlative negative sampling.

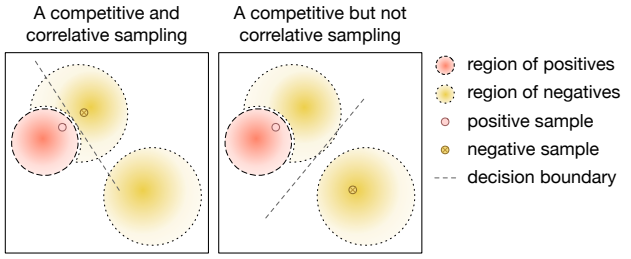


Figure 3: Effect visualization of the correlative sampling

Denote the item and history embedding functions in the discriminator as $\boldsymbol{e}_D^i$ and $\boldsymbol{e}_D^h$ respectively. In order to select a negative sample s' which keeps correlative to the positive sample s. We use Euclidean distances in the discriminator embedding space as a penalty $p(s, s')$, formulated as:

$$p(s, s') = \lambda_i \left\| \boldsymbol{e}_D^i(s) - \boldsymbol{e}_D^i(s') \right\|_2 + \lambda_h \left\| \boldsymbol{e}_D^h(s) - \boldsymbol{e}_D^h(s') \right\|_2 , \quad (10)$$

where $\lambda_i, \lambda_h > 0$ are penalty coefficients for the item embedding penalty and the history embedding penalty respectively.

The objective of the generator is designed to maximize the expectation of scores of the selected negative samples with penalties for correlative restriction. The loss function can be written as:

$$\mathcal{L}_G = \sum_{s \in \mathcal{T}} \underbrace{\mathbb{E}_{s' \sim p_G(s'|s)} \left[f_D\left(\boldsymbol{e}_D^s(s')\right) - p(s, s') \right]}_{\text{denote as } \mathcal{J}_G(s)} . \quad (11)$$

A typical technique to calculate the gradient of the function with expectations is to adopt the idea of policy gradient based reinforcement learning (REINFORCE)[31, 34]. Denote the parameters in the generator as θ_G. The gradient of $\mathcal{J}_G(s)$ is derived as:

$$\begin{aligned}
\nabla_{\theta_G} \mathcal{J}_G(s) &= \nabla_{\theta_G} \mathbb{E}_{s' \sim p_G(s'|s)} \left[f_D\left(\boldsymbol{e}_D^s(s')\right) - p(s, s') \right] \\
&= \sum_{n=1}^{N} \nabla_{\theta_G} p_G(s'_n|s) \left[f_D\left(\boldsymbol{e}_D^s(s'_n)\right) - p(s, s'_n) \right] \\
&= \sum_{n=1}^{N} p_G(s'_n|s) \nabla_{\theta_G} \log p_G(s'_n|s) \left[f_D\left(\boldsymbol{e}_D^s(s'_n)\right) - p(s, s'_n) \right] \\
&= \mathbb{E}_{s' \sim p_G(s'|s)} \left\{ \nabla_{\theta_G} \log p_G(s'|s) \left[f_D\left(\boldsymbol{e}_D^s(s')\right) - p(s, s') \right] \right\} \\
&\simeq \frac{1}{K} \sum_{k=1}^{K} \nabla_{\theta_G} \log p_G(s'_k|s) \left[f_D\left(\boldsymbol{e}_D^s(s'_k)\right) - p(s, s'_k) \right] . \quad (12)
\end{aligned}$$

The approximation in Eq. (12) is based on Monte Carlo method. To update the generator, at each step, we sample a mini-batch of positives and K corresponding negative samples for each positive sample. With the REINFORCE terminology, the term $[f_D(\boldsymbol{e}_D^s(s'_k)) - p(s, s'_k)]$, denoted as $r(s'_k)$, acts as the reward for the policy $p_G(s'_k|s)$ which takes an action s'_k given the environment state s.

Based on the above description, we can utilize the strong guidance brought by the observed negatives, by adversarially selecting the competitive negatives with embedding distance regularizations.

Our GAN-based sampling strategy is specifically designed for CTR prediction tasks or some of the recommender tasks where observed negative samples are available. We further unify our sampling strategy with the previous strategy illustrated in Figure 1, where no observed negatives are available and the nonpositive samples (randomly user-item combinations which are not positive) are used to train against the positive samples. In this case, we fix the user part of each positive sample, and combine the user to an item that are not interacted with the user to form a negative sample. With the same user part, the history embedding penalty in Eq. (10) is zero, thus $r(s'_k)$ can degenerate to an easier form. Summarizing the both conditions, the reward $r(s'_k)$ is formulated as:

$$r(s'_k) = \begin{cases} f_D\left(\boldsymbol{e}_D^s(s'_k)\right) - p(s, s'_k) & \text{with negatives} \\ f_D\left(\boldsymbol{e}_D^s(s'_k)\right) - \lambda_i \left\| \boldsymbol{e}_D^i(s) - \boldsymbol{e}_D^i(s'_k) \right\|_2 & \text{without negatives} \end{cases} . \quad (13)$$

As a common optimization in policy gradient to reduce variance, we can subtract a baseline b from the reward, where b equals to the average reward of the mini-batch $\mathcal{T}_{batch}$. The baseline b updates after each training step by the following equation:

$$b = \frac{1}{|\mathcal{T}_{batch}|} \sum_{s \in \mathcal{T}_{batch}} \frac{1}{K} \sum_{k=1}^{K} r(s'_k) . \quad (14)$$

With the technique of stochastic optimization, in a mini-batch $\mathcal{T}_{batch}$, the gradient of loss function $\mathcal{L}_G$ in the generator is:

$$\nabla_{\theta_G} \mathcal{L}_G \simeq \sum_{s \in \mathcal{T}_{batch}} \frac{1}{K} \sum_{k=1}^{K} \log p_G(s'_k|s) \left[r(s'_k) - b \right] , \quad (15)$$

where the baseline b is obtained from the previous mini-batch.

The sampling policy $p_G(s'|s)$ for the negative sample s' regarding to a positive sample s, is modeled as a union function of the generator sample embeddings $\boldsymbol{e}_G^s(s)$ and $\boldsymbol{e}_G^s(s')$:

$$p_G(s'|s) = softmax_{s' \in Neg(s)} \frac{\boldsymbol{e}_G^s(s')^\top \boldsymbol{e}_G^s(s)}{T \left\| \boldsymbol{e}_G^s(s') \right\|_2} , \quad (16)$$

where $Neg(s)$ is the set of candidate negative samples for sample s; T is a temperature for sensitivity control; dividing $\|\boldsymbol{e}_G^s(s')\|_2$ is to eliminate the unfairness brought by the scale of $\boldsymbol{e}_G^s(s')$.

As the negative samples in practical CTR tasks are noisy, negative samples with the highest scores are likely to be false negatives, i.e. neglected positive samples, which will impact the training performance. To tackle this issue, we generate the set $Neg(s)$ by uniformly sampling C negative samples as candidates, instead of the whole negatives, where C is a hyperparameter. Denote the size of the positives is $|\mathcal{T}|$. For each positive sample s, calculating its corresponding policy needs $O(C)$ time complexity, and the total policy calculation expense in an epoch is $O(|\mathcal{T}| \cdot C)$. Such a complexity can be reduced to $O(|\mathcal{T}| \cdot \log C)$ with hierarchical softmax[21].

GAN can suffer from the mode collapse issue where the generator collapses to some certain modes[2]. In our sampling process,

Algorithm 1 Training the adversarial sampling network

Require: positive samples $\mathcal{T}$ and negative samples $\mathcal{T}'$
Ensure: adversarially trained discriminator
1: Pre-train the discriminator D and the generator G
2: **repeat**
3: Sample a mini-batch positive samples $\mathcal{T}_{batch} \in \mathcal{T}$;
4: **for** each positive sample $s \in \mathcal{T}_{batch}$ **do**
5: Uniformly select C candidate negative samples from $\mathcal{T}'$
6: In G, sample a negative sample s' from the C candidates according to the policy $p_G(s'|s)$ in Eq. (16)
7: In D, optimize $\mathcal{L}_D$ by a stochastic gradient descent
8: In D, calculate the score $f_D(s')$ and the reward $r(s')$ according to Eq. (13)
9: In G, optimize $\mathcal{L}_G$ by a stochastic gradient ascent according to Eq. (15)
10: **end for**
11: **until** convergence

the generator relies on $p_G(s'|s)$ to sample the negatives, and the collapse of $p_G(s'|s)$ to some certain negatives will overfit on these negatives. Thus we should balance the exploration process referring to traveling through the negatives, and the exploitation process referring to sampling the negatives unfairly regarding to the rewards.

In Eq. (16), the sensitivity of softmax can be adjusted by adjusting the temperature T. A larger T will lead to a lower sampling sensitivity, which encourages the generator to explore, and a lower T encourages the generator to exploit. The generator should explore at the early training to meet more negative samples, while exploit later to take advantage of the adversarial sampling. Thus we give a large initial temperature and decay it epoch by epoch with a decay rate. Sensitivity analysis of T can be seen in Section 4.4.

We experimentally set the parameter K in Eq. (12) to one and find it sufficient to deliver promising results. The main framework of our adversarial training algorithm is described in Algorithm 1.

3.2 Output Calibration

The proposed adversarial sampling strategy brings better relative CTR values which benefit the ranking process in CPC advertising. But during the negative sampling, the proportion of the positives and the negatives in the constructed training data will not match the real data proportion. Such mismatching will lead to an inaccurate absolute CTR estimates which is bad for the bidding process. Inspired by the calibration method in [16], we adopt a similar output calibration by taking into account the real data proportion.

The real CTR of a sample s for training can be expressed as:

$$\mathrm{CTR}(s) = p\big(Y(s) = 1 | f_D\big(\boldsymbol{e}_D^s(s)\big)\big)\,, \tag{17}$$

where $Y(s) = 1$ indicates that the sample s is a positive sample.

We apply a sigmoid function on the score $f_D(\boldsymbol{e}_D^s(s))$ for $[0, 1]$ normalization, and denote the normalized score as $\sigma(s)$. We divide the region $[0, 1]$ into n equal-sized buckets, where $0 \le v_1 < v_2 < , \cdots , < v_{n+1} \le 1$ and n is large enough. Assuming that $\sigma(s)$ locates in $[v_j, v_{j+1}),\ j = 1, \cdots, n$, an approximation of $\mathrm{CTR}(s)$ is:

$$\mathrm{CTR}(s) \simeq p\big(Y(s) = 1 | v_j \le \sigma(s) < v_{j+1}\big)\,, \tag{18}$$

Denote the right part of Eq. (18) as $p(v_j),\ j = 1, \cdots, n$. An approximation of $p(v_j)$ is the proportion of the positive training samples in all training samples, written as:

$$p(v_j) \simeq \frac{\#\text{ Positive training samples with } \sigma(s) \in [v_j, v_{j+1})}{\#\text{ All training samples with } \sigma(s) \in [v_j, v_{j+1})}\,. \tag{19}$$

An isotonic regression algorithm should be applied to keep $p(v_j)$ monotonically increasing with $j = 1, \cdots, n$. We experimentally choose Pool Adjacent Violators Algorithm[17] for regression.

After obtaining the estimates of $p(v_j),\ j = 1, \cdots, n$. For a single sample $\hat{s}$ in the testing data, its CTR is calibrated by:

$$\begin{aligned}\mathrm{CTR}(\hat{s}) &\simeq p\big(Y(\hat{s}) = 1 | v_j \le \sigma(\hat{s}) < v_{j+1}\big) \\ &= \alpha p(v_j) + (1 - \alpha) p(v_{j+1})\,,\end{aligned} \tag{20}$$

where we suppose $\sigma(\hat{s}) \in [v_j, v_{j+1})$, and $\alpha = \frac{v_{j+1} - \sigma(\hat{s})}{v_{j+1} - v_j}$.

The final calibrated CTR value of the testing data is the average calibrated CTR of all testing samples. Experimental results of the output calibration are shown in Section 4.3.

4 EXPERIMENTS

We conduct experiments on three real-world industrial datasets provided by JD.COM, one of which is from in-station advertising places (ads are displayed in the JD mobile app) and the other two are from out-station advertising places (ads are displayed in the third-party platforms). The datasets we choose have various magnitudes and CTR values for distinction considerations. A summary of the datasets is provided in Table 1.

We use AUC (area under the receiver operating characteristic curve) for performance evaluation. AUC is a commonly-used evaluation metric for CTR prediction, which measures the quality of the order by ranking all the samples with predicted CTR[10].

Referring to [32, 36] , we adopt RelaImpr metric to measure the relative enhancement of AUC scores over base models. As the AUC score of a random guess is 0.5, RelaImpr is defined as:

$$\mathrm{RelaImpr} = \left(\frac{\text{AUC of measured model} - 0.5}{\text{AUC of base model} - 0.5} - 1\right) \times 100\%\,. \tag{21}$$

4.1 Baselines

For verifying the time-aware attention model, we select the following base models for comparison, and we uniformly adopt the regularized adversarial sampling for the sampling process.

(1) **Two-layer GRU**[9]: We implement a two-layer GRU network to represent users' historical records.
(2) **DIN**[36]: DIN adopts an attention-based model (without GRU) for activating related user behaviors and obtains the relative interests to a target item.
(3) **GRU Attention**: We discard the temporal components of our time-aware attention model, using a two-layer GRU to model users' sequential behaviors and a regular attention module for activating inner relations between items.

For verifying the regularized adversarial sampling (rGAN), we select the following models for comparison, and we uniformly adopt the time-aware attention model for the embedding process.

(1) **Logistic Regression**[20]: Logistic regression is an ordinary pointwise training method which uses all available samples.

Table 1: Summary of the data structure (Year: 2018)

Dataset	Items	Categories	Training Data				Testing Data			
			Date	Positives	Negatives	CTR	Date	Positives	Negatives	CTR
In-station-Sep.	1,680,735	3,137	Sep.14th	126,016	8,233,656	1.507%	Sep.15th	138,676	9,173,364	1.489%
Out-station-Jul.	3,411,957	4,574	Jul.5th~Jul.6th	35,262	2,868,936	1.214%	Jul.7th	17,630	1,352,200	1.287%
Out-station-May.	3,734,557	4,389	May.6th	18,309	1,502,377	1.204%	May.7th	18,780	1,514,364	1.220%

(2) **1:5 Under Sampling**[13, 18]: Under sampling is a common sampling method to balance distributions for CTR prediction. The ratio 1:5 is a tuned choice that realizes better results among various under sampling ratio for our datasets.
(3) **User-fixed Sampling**[33]: User-fixed sampling is a regular sampling method for CTR prediction which selects negative samples that share the same users with the positives. Uniform sampling is used as a complementary choice if there are not enough negative samples that meet requirements.
(4) **Uniform Sampling**[13]: Uniform sampling, where the negatives are selected uniformly given any positive sample, is a commonly-used CTR sampling method, and can be seen as a degradation of our rGAN strategy when T is large or $C = 1$.
(5) **IRGAN**[27]: IRGAN is a state-of-the-art model which applies adversarial sampling to select non-interactive user-item samples to train against the positive samples. However, IRGAN cannot use the information of the practically observed negative samples in CTR prediction tasks.
(6) **IRGAN++**: We modify IRGAN to sample from the practically observed negatives. As directly sampling from the negatives with IRGAN fails to consider the correlation (illustrated in Figure 3) between samples, we explore some modifications of the original IRGAN by narrowing down the sampling scope. We denote the modification with the best result as IRGAN++: given a positive sample, IRGAN is applied to sample from the negatives that share the same target item with the positive sample; and if there are not sufficient negative samples, we sample from the negatives of which the target items belong to the same category as the target item of the positive sample.

4.2 Implementation Details

The structural details of the embedding networks illustrated in Figure 2 are described as follows:

(1) Item embedding: For an item, we convert its cid3 and four absolute time signals to five one-hot embeddings, each of which is further sent to a fully-connected layer with tanh activation function. Every obtained output has eight dimensions, and is concatenated with the item's 50-dimensional raw features embedding obtained by pre-training using Telepath[30]. The fully-connected network between the concatenation and the item embedding has three 90-dimensional layers, with ReLU for the hidden layers and tanh for the output layer.
(2) History embedding: Each historical record in a sample contains the user's 10 latest clicking items, and their corresponding item embeddings are obtained using the same structure described in (1). The number 10 is selected by experimental results. These item embeddings are sent to a two-layer GRU network with 90 RNN size as inputs. The dimension of the relative time embedding is 90. We further apply the time-aware attention module on GRU. The history embedding, which is the output of the attention, has 90 dimensions.
(3) Auxiliary embedding: We apply a one-layer fully-connected network with an eight-dimensional output layer on the auxiliary features. The layer uses tanh as the activation function.

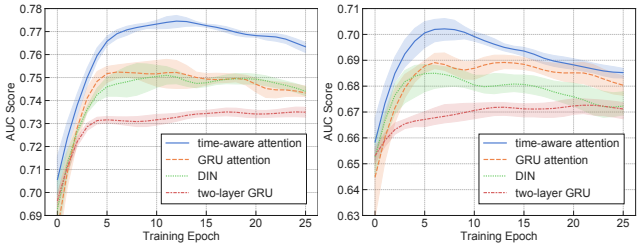
In the discriminator, the score net is a linear single-layer fully-connected network with one-dimensional outputs.

We use Adam[14] for both trainings in discriminator and generator. Parameters follow $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.

The model is trained for 50 epochs, where the first 25 are shown in results. For each sample batch, we set one step for training either the discriminator or the generator. We adjust the batch size to keep 30 steps per epoch. The initial learning rate for the discriminator and the generator are respectively set to 0.02 and 0.01, which both decay 50% every 10 epochs. K in Eq. (15) is set to one. The margin of the loss function is 1.0. The penalty coefficients for the item embedding and the history embedding are 3.0 and 5.0 respectively. The candidate size C is set to 35 for the In-station-Sep. dataset and set to 20 for the other two datasets. The initial temperature T is constantly set to 20.0, with a decay rate 0.98.

4.3 Comparison Results

We tune key hyperparameters individually for each baseline method, and collect the results in two aspects: comparison of the time-aware attention with other embedding models, and comparison of the regularized adversarial sampling with other sampling strategies.



(a) In-station-Sep. dataset (b) Out-station-Jul. dataset

Figure 4: Performance of various embedding models

Figure 4 illustrates the AUC scores of the time-aware attention and several embedding baselines for two datasets. For each model, we plot the corresponding average curves of five results. We adopt the same regularized adversarial sampling for all the models so as to observe the effect brought by the time-aware attention individually.

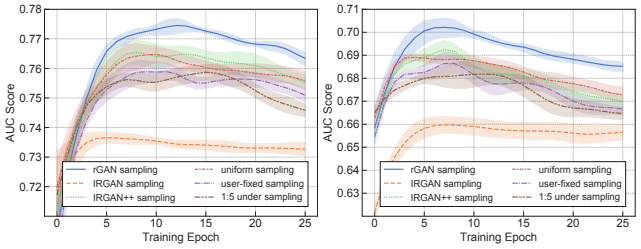
Table 2 shows the average optimal AUC scores of various embedding models for the three datasets. Observe that the proposed embedding model itself brings 8%~9% relative AUC improvements. The comparison between the time-aware attention and GRU attention proves the effect of the temporal components.

Table 2: Results of various embedding models

Model	in-station-Sep.		out-station-Jul.		out-station-May.	
	AUC	RelaImpr	AUC	RelaImpr	AUC	RelaImpr
Two-layer GRU	0.7350	-6.33%	0.6726	-6.70%	0.6271	-4.36%
DIN*	0.7509	0.00%	0.6850	0.00%	0.6329	0.00%
GRU Attention	0.7523	0.56%	0.6892	2.27%	0.6322	-0.53%
Time-aware Attention	**0.7745**	**9.41%**	**0.7021**	**9.24%**	**0.6439**	**8.28%**

In each table, * indicates the baseline model for calculating RelaImpr.
rGAN sampling is applied for all models in Table 2.

We conduct experiments to verify the effect of regularized adversarial (rGAN) sampling. For each model, we adopt the same time-aware attention model for the embedding process, aiming to observe the individual effect of rGAN sampling. Figure 5 shows six pairwise training methods, where rGAN sampling model promotes the training performance with better AUC scores.



(a) In-station-Sep. dataset (b) Out-station-Jul. dataset

Figure 5: Performance of various sampling models

Table 3 shows the average optimal AUC scores of one pointwise model and six pairwise sampling models on three datasets. During the training process, we find that with proper parameter settings, rGAN stably outperforms the other models. It is worth mention that the rGAN sampling is not only suitable for our time-aware attention model, but also can help to improve the AUC scores for other embedding models, such as the GRU model and DIN.

Table 3: Results of various sampling models

Model	in-station-Sep.		out-station-Jul.		out-station-May.	
	AUC	RelaImpr	AUC	RelaImpr	AUC	RelaImpr
Logistic Regression	0.7643	0.15%	0.6790	-5.34%	0.6251	-7.81%
1:5 Under Sampling	0.7587	-1.97%	0.6818	-3.86%	0.6270	-6.41%
User-fixed Sampling	0.7589	-1.89%	0.6866	-1.32%	0.6379	1.62%
Uniform Sampling*	0.7639	0.00%	0.6891	0.00%	0.6357	0.00%
IRGAN Sampling	0.7366	-10.34%	0.6597	-15.55%	0.6165	-14.15%
IRGAN++ Sampling	0.7655	0.61%	0.6924	1.75%	0.6380	1.69%
rGAN Sampling	**0.7745**	**4.02%**	**0.7021**	**6.87%**	**0.6439**	**6.04%**

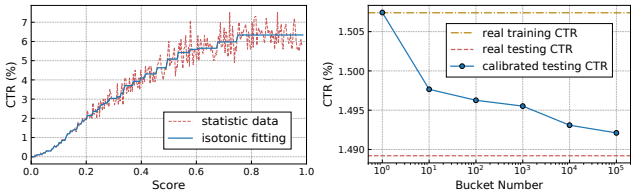
Time-aware attention is applied for all models in Table 3.

As either of the previous results shows the individual effect of the time-aware attention or the regularized adversarial sampling, we now provide their overall improvements in Table 4. We make comparison between our overall model and DIN with uniform sampling. As shown in the table, our model acquires 11%~15% relative improvements of the AUC scores, which is more remarkable than many state-of-the-art CTR prediction works like [12, 36] in real-world datasets.

Table 4: Overall results of the proposed work

Model	in-station-Sep.		out-station-Jul.		out-station-May.	
	AUC	RelaImpr	AUC	RelaImpr	AUC	RelaImpr
DIN+Uniform*	0.7405	0.00%	0.6754	0.00%	0.6296	0.00%
Ours	**0.7745**	**14.14%**	**0.7021**	**15.22%**	**0.6439**	**11.03%**

Experiments above indicate that the proposed models bring markedly better relative CTR values which will benefit the ranking process in CPC advertising. Considering the bidding process, we now follow subsection 3.2 to calibrate the absolute CTR values. For the In-station-Sep. dataset, Figure 6(a) illustrates the curves of $p(v_j)$ in Eq. (19) with 300 buckets (denote as n) and the corresponding isotonic regression fitting using PAVA[17]. We add a small slope $\epsilon = 0.1$ on the isotonic fitting curve to make it strictly increasing. Figure 6(b) illustrates absolute CTR calibrated results on the testing data of the In-station-Sep. dataset w.r.t. various bucket numbers. With 10^5 buckets, the calibration error drops to as low as 0.19%. We also verify the performance on the out-station-Jul. and out-station-May. datasets where the error are 0.20% and 0.22% respectively. When calibrated CTR values of some certain categories (cid3) are required, we can directly apply the same calibration procedure specifically on these certain categories.



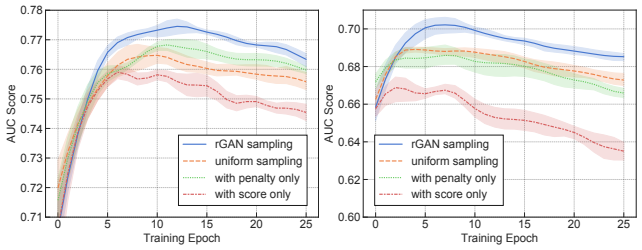
(a) $p(v_j)$ and its isotonic fitting (b) Calibrated results w.r.t. bucket numbers

Figure 6: Performance of the absolute CTR calibration

4.4 In-depth Model Analysis

In this part, we dive into an in-depth model analysis and quantify the benefits of the components that build-up the proposed regularized adversarial (rGAN) sampling strategy.

In Eq. (13), the reward function of rGAN sampling contains a score term and a distance penalty term. To show the importance of the two components, we omit either one of them and conduct comparison experiments. Figure 7 illustrates that a good cooperation of the score and the penalty will improve the CTR prediction performance, and the two terms are both indispensable.



(a) In-station-Sep. dataset (b) Out-station-Jul. dataset

Figure 7: Performance of various reward structures

We further explore the relation between scores and penalties during training. For each positive sample s, we rank the selected C candidate negatives s'_j according to their scores $f_D(\boldsymbol{e}^s_D(s'_j))$ and their negative penalties $-p(s, s'_j)$ respectively, where $j = 1, \cdots, C$. Thus we get two permutations for s both with length C. We apply Kendall Tau correlation coefficient[1] to measure the similarity between the permutations, where a higher Tau coefficient indicates more similarity between the permutations. We average over the Tau coefficients of the mini-batch positive samples for each training step. Figure 8 illustrates the Tau coefficient curve in out-station-Jul. dataset. We also provide Tau curve between the permutation of score and a random permutation for comparison. The corresponding AUC score curve is together plotted in the figure, which reaches its maximum where the permutations have high Tau coefficient.

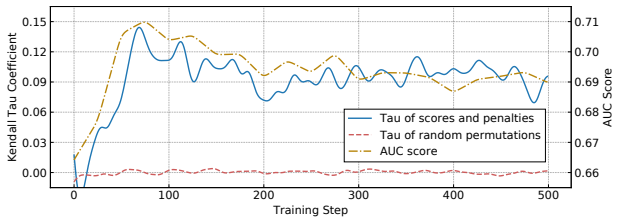
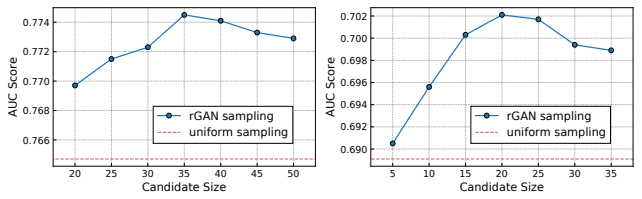


Figure 8: Tau coefficient of scores and negative penalties

We provide the sensitivity analysis of two key hyperparameters in the regularized adversarial (rGAN) sampling, the candidate size and the initial temperature. Other parameters follow the default settings described in Section 4.2.



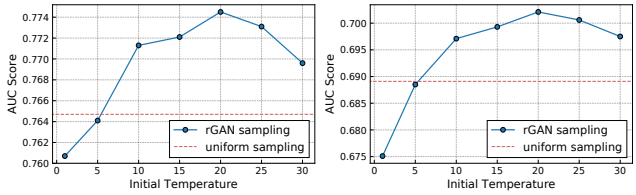
(a) In-station-Sep. dataset (b) Out-station-Jul. dataset

Figure 9: Sensitivity analysis of the candidate size

Impact of candidate size. The candidate size C in rGAN sampling is an essential hyperparameter. rGAN with a small candidate size will degenerate into the uniform sampling, and with a large candidate size will be vulnerable to the false negatives (as analyzed in Section 3.1). We test the sensitivity of the candidate size using In-station-Sep. and Out-station-Jul. datasets, and show the average results in Figure 9. Candidate size has obvious impact on the performance and needs adjusting according to both the data amount and the noise level.

Impact of initial temperature. As analyzed in Section 3.1, the temperature T is used to adjust the sensitivity of Eq. (16). We conduct experiments to observe the results with various initial temperatures shown in Figure 10.

Results illustrate that the AUC score increases firstly and then decreases as temperature increases. The two datasets both reach the best performance when $T = 20$, which shows that the temperature has good generalization to datasets.



(a) In-station-Sep. dataset (b) Out-station-Jul. dataset

Figure 10: Sensitivity analysis of the initial temperature

5 RELATED WORK

We summarize the related works according to the deep CTR prediction models and the adversarial negative sampling methods.

With the rise of deep learning, CTR prediction models have evolved from shallow to deep, aiming to improve the model capacities. Typical models like Wide&Deep[7] which combines low-order logistic regression components and high-order features with neural networks for better representation of feature interactions. PNN[23] imposes networks based on inner and outer products for stronger expression of cross features. DCN[29] introduces a novel cross network which is more efficient in learning feature interactions explicitly by apply feature crossing at each layer. These deep CTR works greatly enhance model capacities, while lack the deeper exploration of the users' historical behaviors. With the evolution of the recommender systems, user-item interactions are detailedly recorded. DIN[36] activates historical behaviors regarding to the target item locally with attention mechanism to capture users' relative interests. DIEN[35] designs an attention-based interest extractor layer to capture diverse interests from users' historical records. However, these state-of-the-art deep CTR prediction models neglect the importance of the temporal signals in users' historical records. Our proposed time-aware attention model aims to improve the performance of CTR prediction giving credit to the temporal signals, which reflect the users' periodic trends and measure the temporal influence of each historical item to the target recommended item.

Generative adversarial nets (GANs), which are originally proposed to fit continuous data distributions[11], have been recently used for negative sampling in discrete data to promote the training efficiency. IRGAN[27] unifies generative and discriminative models of information retrieval into a discrete GAN framework. AdvIR[22] expands IRGAN by adding additional generated adversarial examples for joint training. [5] learns by contrasting observed and fictitious samples with an adversarially learned sampler. KBGAN[6] designs an adversarial framework with dual KGE components for improving knowledge graph embedding models. [28] further extends the idea to recommender systems, and proposes an adaptive adversarial negative sampling scheme to generate negative samples for each user, where the item in each negative sample is selected from all items not interacted with the user. A common point of these works is that, each negative sample is a combination of two components with few interactions. In many application domains such as CTR prediction tasks, however, observed negative interactions are available as well, which provide stronger negative guidance than nonpositive user-item combinations. Our regularized adversarial sampling is designed for CTR prediction tasks and unlike these common adversarial sampling strategies, it can make use of the strong information of the observed negative samples.

6 CONCLUSION

In this paper, we focus on designing a temporal embedding model and an adversarial sampling strategy that can promote the performance of CTR prediction tasks. The proposed attention-based model is capable to represent the users' periodic behaviors and the temporal relations between historical items and target items, by considering absolute and relative temporal signals. In addition, the proposed regularized adversarial negative sampling is able to make use of the stronger guidance provided by the negative CTR samples, which is different from existing adversarial sampling methods in recommender systems. And the idea of regularization in adversarial sampling can be potentially extend to other fields. We test our models in three real-world CTR datasets. Comparison results show that the collaboration of the time-aware attention and the regularized adversarial sampling strategy outperforms the state-of-the-art CTR prediction models.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China under Grants 61621136008 and 91848206.

REFERENCES

[1] H. Abdi. 2007. Kendall Rank Correlation. In *Encyclopedia of Measurement and Statistics*.
[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. In *ICML*. 214–223.
[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
[4] Gustavo E. A. P. A. Batista, André Carlos Ponce Leon Ferreira de Carvalho, and Maria Carolina Monard. 2000. Applying One-Sided Selection to Unbalanced Datasets. In *MICAI*. 315–325.
[5] Avishek Joey Bose, Huan Ling, and Yanshuai Cao. 2018. Adversarial Contrastive Estimation. In *ACL*. 1021–1032.
[6] Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial Learning for Knowledge Graph Embeddings. In *NAACL-HLT*. 1470–1480.
[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *DLRS*. 7–10.
[8] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*. 1724–1734.
[9] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *RecSys*. 152–160.
[10] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognition Letters* 27, 8 (2006), 861–874.
[11] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NIPS*. 2672–2680.
[12] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *IJCAI*. 1725–1731.
[13] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *ADKDD*. 5:1–5:9.
[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*. 1106–1114.
[16] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating Conversion Rate in Display Advertising from Past Performance Data. In *SIGKDD*. 768–776.
[17] Jan De Leeuw, Hornik Kurt, and Mair Patrick. 2009. Isotone Optimization in R: Pool-Adjacent-Violators Algorithm (PAVA) and Active Set Methods. *Journal of Statistical Software* 32, 5 (2009), 1–24.
[18] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2006. Exploratory Under-Sampling for Class-Imbalance Learning. In *ICDM*. 965–969.
[19] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*. 1412–1421.
[20] H. Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad click prediction: a view from the trenches. In *SIGKDD*. 1222–1230.
[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.
[22] Dae Hoon Park and Yi Chang. 2019. Adversarial Sampling and Training for Semi-Supervised Information Retrieval. In *WWW*. 1443–1453.
[23] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-Based Neural Networks for User Response Prediction. In *ICDM*. 1149–1154.
[24] Steffen Rendle. 2010. Factorization Machines. In *ICDM*. 995–1000.
[25] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS*. 3104–3112.
[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*. 6000–6010.
[27] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *SIGIR*. 515–524.
[28] Qinyong Wang, Hongzhi Yin, Zhiting Hu, Defu Lian, Hao Wang, and Zi Huang. 2018. Neural Memory Streaming Recommender Networks with Adversarial Training. In *SIGKDD*. 2467–2475.
[29] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *ADKDD*. 12:1–12:7.
[30] Yu Wang, Jixing Xu, Aohan Wu, Mantian Li, Yang He, Jinghe Hu, and Weipeng P. Yan. 2018. Telepath: Understanding Users from a Human Vision Perspective in Large-Scale Recommender Systems. In *AAAI*. 467–474.
[31] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* (1992), 229–256.
[32] Ling Yan, Wu-Jun Li, Gui-Rong Xue, and Dingyi Han. 2014. Coupled Group Lasso for Web-Scale CTR Prediction in Display Advertising. In *ICML*. 802–810.
[33] Yan Yan, Wentao Guo, Meng Zhao, Jinghe Hu, and Weipeng P. Yan. 2017. Optimizing Gross Merchandise Volume via DNN-MAB Dynamic Ranking Paradigm. *CoRR* abs/1708.03993 (2017). arXiv:1708.03993 http://arxiv.org/abs/1708.03993
[34] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI*. 2852–2858.
[35] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep Interest Evolution Network for Click-Through Rate Prediction. In *AAAI*. 5941–5948.
[36] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *SIGKDD*. 1059–1068.