TWIN: TWo-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou

Jianxin Chang
changjianxin@kuaishou.com
Kuaishou, China

Chenbin Zhang
zhangchenbin@kuaishou.com
Kuaishou, China

Zhiyi Fu
fuzhiyi@kuaishou.com
Kuaishou, China

Xiaoxue Zang
zangxiaoxue@kuaishou.com
Kuaishou, China

Lin Guan
guanlin03@kuaishou.com
Kuaishou, China

Jing Lu*
lvjing06@kuaishou.com
Kuaishou, China

Yiqun Hui
huiyiqun@kuaishou.com
Kuaishou, China

Dewei Leng
lengdewei@kuaishou.com
Kuaishou, China

Yanan Niu
niuyanan@kuaishou.com
Kuaishou, China

Yang Song
yangsong@kuaishou.com
Kuaishou, China

Kun Gai
gai.kun@qq.com
Unaffiliated, China

ABSTRACT

Life-long user behavior modeling, i.e., extracting a user's hidden interests from rich historical behaviors in months or even years, plays a central role in modern CTR prediction systems. Conventional algorithms mostly follow two cascading stages: a simple General Search Unit (GSU) for fast and coarse search over tens of thousands of long-term behaviors and an Exact Search Unit (ESU) for effective Target Attention (TA) over the small number of finalists from GSU. Although efficient, existing algorithms mostly suffer from a crucial limitation: the *inconsistent* target-behavior relevance metrics between GSU and ESU. As a result, their GSU usually misses highly relevant behaviors but retrieves ones considered irrelevant by ESU. In such case, the TA in ESU, no matter how attention is allocated, mostly deviates from the real user interests and thus degrades the overall CTR prediction accuracy. To address such inconsistency, we propose **TWo-stage Interest Network (TWIN)**, where our Consistency-Preserved GSU (CP-GSU) adopts the identical target-behavior relevance metric as the TA in ESU, making the two stages twins. Specifically, to break TA's computational bottleneck and extend it from ESU to GSU, or namely from behavior length 10^2 to length $10^4 - 10^5$, we build a novel attention mechanism by behavior feature splitting. For the video inherent features of a behavior, we calculate their linear projection by efficient pre-computing & caching strategies. And for the user-item cross features, we compress each into a one-dimentional bias term in the attention score calculation to save the computational cost. The consistency between two stages, together with the effective TA-based relevance metric in CP-GSU, contributes to significant performance gain in CTR prediction. Offline experiments on a 46 billion scale real production dataset from Kuaishou and an Online A/B test show that TWIN outperforms all compared SOTA algorithms. With optimized online infrastructure, we reduce the computational bottleneck by 99.3%, which contributes to the successful deployment of TWIN on Kuaishou, serving the main traffic of hundreds of millions of active users everyday.

ACM Reference Format:
Jianxin Chang, Chenbin Zhang, Zhiyi Fu, Xiaoxue Zang, Lin Guan, Jing Lu, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. TWIN: TWo-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

As one of the most popular short video sharing apps in China, *Kuaishou* strongly relies on its powerful recommendation system (RS). Every day, RS helps hundreds of millions of active users to filter out millions of uninterested videos and reach their interested ones, leaving tens of billions of click through logs. These tremendous data not only feed the training of RS, but also boost technique revolutions that keep lifting both the user experience and business effectiveness on this platform.

In modern RSs, a fundamental task is Click Through Rate (CTR) prediction which aims to predict the probability that a user would click an item / video [2, 10, 31]. Accurate CTR prediction directs RS to serve each user one's favourite contents and deliver each video to its interested audiences. To achieve this, CTR models should be highly personalized and make full use of scarce user information. Consequently, *life-long user behavior modeling*, i.e., extracting a user's hidden interests from rich long-term historical behaviors, usually acts as a key component in CTR models [7, 16, 33–35].

*Corresponding Author

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

Figure 1: Inconsistency between GSU & ESU in a Conventional Two-Stage Algorithm. Assume that an "Oracle" (blue) could afford to use the identical relevance metric as that in ESU on 10^4 behaviors, namely find "the real top-100". While GSU (orange) uses an ineffective and inconsistent coarse search. Among the top-100 returned by GSU (x axis), only 40 hit the real top-100 (y axis). This inconsistency (gray) indicates the potential improvement space left for TWIN.

Industrial life-long behavior modeling algorithms mostly follow the two cascading stages [19]: (1) a General Search Unit (GSU) that conducts a fast coarse search over tens of thousands of long-term behaviors and outputs a small number of most target-relevant ones, and (2) an Exact Search Unit (ESU) that performs effective *Target Attention* (TA) over the small number of finalists from GSU. The reason behind this two-stage design is twofold. On one hand, to precisely capture the user interest, TA is a proper choice for emphasizing target-relevant behaviors and suppressing target-irrelevant ones. On the other hand, the expensive computational cost of TA limits its applicable sequence length to at most a few hundreds. To this end, a simple and fast GSU as a pre-filter is essential for cutting-off industrial scale behavior sequences which could easily reach 10^4 in just a few months.

Recent years have witnessed a great many emerging studies on two-stage life-long behavior modeling, while their key difference lies in the GSU strategies that coarsely select target-relevant behaviors. For example, SIM Hard simply selects behaviors from the same category as the target item, while SIM Soft calculates the target-behavior relevance score from pre-trained item embeddings by the inner product and selects behaviors with the highest relevance [19]. ETA approximates the relevance score calculation using locality-sensitive hashing (LSH) and Hamming distance [3]. SDIM samples behaviors with the same hash signature as the target-behavior through multi-round hash collision [1], among others. Despite being extensively studied, existing two-stage life-long behavior modeling algorithms still suffer from a crucial limitation: the *inconsistency* between GSU and ESU (shown in Fig 1[1]). Specifically, the target-behavior relevance metric used in GSU is both coarse and inconsistent with the TA used in ESU. As a result, GSU may probably miss relevant behaviors, but retrieve ones considered irrelevant by ESU, wasting ESU's precious computational resources. In such case, the TA in ESU, no matter how attention is allocated, mostly deviates from the real user interests and thus degrades the overall CTR prediction accuracy.

To address such *inconsistency*, we propose TWIN: TWo-stage Interest Network for lifelong user behavior modeling, where a Consistency-Preserved GSU (CP-GSU) adopts the *identical* target-behavior relevance metric as the TA in ESU, making the two stages *twins*. To extend the expensive TA to CP-GSU, TWIN breaks TA's key computational bottleneck, namely the linear projection of 10^4 behaviors, by effective behavior feature split, simplified TA architecture and highly optimized online infrastructure. 1). Specifically, for the video inherent features of a behavior (e.g. video id, author, duration, topic) which are shared across users / behavior sequences, we accelerate their projection by efficient pre-computing & caching strategies. 2). And for the the user-video cross features of a behavior (e.g. user's click timestamp, play time, rating), where caching is not applicable, we simplify the TA architecture by compressing their projection into bias terms. With optimized online infrastructure, we successfully extend the applicable sequence length of TA from 10^2 in ESU to $10^4 - 10^5$ in CP-GSU. The consistency between two stages, together with the effective TA-based relevance metric in CP-GSU, contributes to significant performance gain in CTR prediction.

Overall, we make the following contributions:

- In our proposed TWIN, CP-GSU precisely and consistently retrieves behaviors that are not only target-relevant, but also considered important by ESU, maximizing the retrieval effectiveness of behavior modeling. To the best of our knowledge, we are the first to successfully address the inconsistency in the two-stage life-long behavior modeling problem.
- We validate the effectiveness of TWIN through extensive offline experiments on Kuaishou's 46 billion scale industrial dataset and online A/B tests. We verify our validity through ablation studies and show that TWIN brings significant online benefits.
- We build efficient industrial infrastructure to apply TWIN on the real online RS. Effective pre-computing & caching strategies are proposed to reduce the computational bottleneck of TWIN, i.e., the linear projection of behaviors in CP-GSU, by 99.3% and to meet the low latency requirements of the online serving system. TWIN has now been deployed on the RS of Kuaishou, serving the main traffic of 346 million active users every day.

2 RELATED WORK

2.1 Click-Through-Rate (CTR) Prediction

CTR prediction which aims to predict a user's personalized interests, is crucial for nowadays RSs. In early days, CTR models are shallow and mainly focus on exploiting feature interactions, such as factorization machines (FM) [21] and field-aware factorization machines (FFM) [12]. With the success of deep learning, deep CTR models are extensively studied and become mainstream choices. For example, Chen et al. [2] and Zhang et al. [32] first apply deep models for CTR tasks. Wide&Deep [5] combines a wide linear model and a deep model, which takes the advantages of both memorization of feature interaction and generalization of deep architecture. DeepFM [10] and DCN [26, 27] improve the wide part of Wide&Deep to increase the feature interaction ability. xDeepFM [15] and AFM [28] further exploit convolution-like layers and attention mechanism to improve the deep part and boost model performance.

[1]The Oracle is identically copied from the ESU of SIM-hard (trained from the top-100 behaviors retrieved by GSU), except that it ranks 10^4 rather than 10^2 behaviors. Experiments are conducted on a small demo dataset from Kuaishou, not feasible for normal datasets due to the extremely high cost of Oracle (details in Section 4.5).

Table 1: Comparison of SOTA user interest models. The bottom part lists the two-stage models. Length refers to the max sequence length of user behaviors in original papers.

Method	Length	GSU Stragegy	End2End	Consist
DIN [35]	$\sim 10^3$	N/A	N/A	N/A
DIEN [34]	$\sim 10^2$	N/A	N/A	N/A
MIMN [18]	$\sim 10^3$	N/A	N/A	N/A
UBR4CTR [20]	$\sim 10^2$	BM25	✗	✗
SIM Hard [19]	$\sim 10^3$	Category Filter	✗	✗
SIM Soft [19]	$\sim 10^3$	Inner Product	✗	✗
ETA [3]	$\sim 10^3$	LSH & Hamming	✓	✗
SDIM [1]	$\sim 10^3$	Hash Collision	✓	✗
TWIN (ours)	$\sim 10^4$	Target Attention	✓	✓

Table 2: Important Notations Used in Section 3

f	predictor	σ	sigmoid	$\hat{y}$	predicted CTR
$\mathcal{D}$	dateset	$\mathbb{R}$	real number set	d	feature dimension
ℓ	loss	$\mathbf{x}$	feature vector	y	ground truth label
E	embedding dictionary			$\mathbf{x}_{\text{emb}}$	embedded feature
$\mathbf{x}_{\text{hot}}$	one/multi hot coding			v	vocabulary size
K	behavior features			L	behavior length
J	cross feature number			a	head index
H	inherent feature dimension			K_h	inherent features
C	cross feature dimension			K_c	cross features
$\boldsymbol{\beta}$	cross feature weight			$\boldsymbol{\alpha}$	attention weight
d_k	dimension of projected inherent features				
d_{out}	dimension of projection in original MHTA				
W^h, W^c, W^v, W^o	linear projection parameters				
$\mathbf{w}_j^c$	a diagonal block of projection parameter W^c				

User behavior modeling, i.e., capturing a user's hidden interest from the summarization of one's historical behaviors, plays an important role in CTR prediction. Limited by computational resources, early algorithms are mostly in target-independent manners and thus can be efficiently pre-calculated offline [8, 23, 30]. To better extract a user's interest in specific items, various TA mechanisms are adopted. The most representative one is DIN [35], which represents the user interest by a weighted pooling over historical behaviors and the pooling weights are calculated from TA to emphasis target-relevant behaviors. DIEN [34] further introduces the temporal relationship of behaviors using ARGRU, an attention-based variant of classic GRU [6]. After that, several works try to improve DIN and DIEN from different aspects. DSIN [9] splits behaviors into multiple sessions and conducts self-attention inside each one to emphasise intra-session relationships. MIND [14] and DMIN [29] argue that representing user interest in one single vector is not enough. MIND uses the Capsule network [22] inside to represent user interests as multiple vectors while DMIN treats representations from different self-attention heads of Transformers [25] as different user interests. Akin to DMIN, BST [4], SASRec [13] and BERT4Rec [24] also use transformers to model user sequential behaviors, improving the model's performance and parallelism.

2.2 Long-Term User Behavior Modeling

As the effectiveness of TA and interest modeling have been confirmed in modern industrial RSs, researchers start to model increasing longer behaviors and try to dig out fine-grained interests from different user behavior stages. For example, Liu and Zamanian [16] proposes to combine long-term and short-term interests in CTR prediction. To process long-term user behaviors up to length of 10^3, MIMN [18] stores the user behaviors as a memory matrix at the user interest center (UIC) and update the memory when new user behavior comes. However, MIMN is hard to extend to longer behavior sequence and it generates the same memory matrix for different candidate items, carrying useless noise and impairing the TA. Recently, SIM [19] and UBR4CTR [20] introduce the two-stage cascaded framework to solve those challenges and achieved SOTA results on the CTR task at that time. SIM consists of a simple and fast GSU, which retrieves the top "similar" items for the target item from thousands of user behaviors, and an attentive ESU to perform TA. In the original SIM, there are two instances with different GSU designs. The GSU of SIM Hard retrieves related items by filtering items from the same category, while the GSU of SIM Soft uses the inner product with pre-trained item embeddings. UBR4CTR utilizes BM25 by regarding items and features as documents and terms.

Though the two-stage design takes a big step forward, the original GSU still faces high computational burden and has different retrieval metric with ESU, which brings the inconsistency problem. More recently, ETA [3] uses locality-sensitive hash (LSH) to encode item embeddings from ESU and retrieve similar items from long-term behaviors via Hamming distance (HD) based on those hash signatures. SDIM [1] samples behavior items with the same hash signature as the target item through multiround hash collision, whose ESU then linearly aggregates those sampled behavior items to obtain user interests. It is positive that ETA and SDIM adopt End2End training, so that the embeddings of the two stages are the same. Nevertheless, inconsistency still exists in the retrieval strategies of ETA and SDIM, specifically the network structure and parameters. Because they just approximate the similarity or relevance computation of ESU in the GSU, which probably misses important behaviors and retrieves unrelated ones, misleading the CTR model. In this paper, we propose to unify the way of similarity computation in GSU&ESU by equally using TA in GSU, and maintain the end-to-end train method, i.e., CP-GSU, whose parameters are updated from ESU. Theoretically, GSU and ESU share the same architecture and parameters. So compared to ETA and SDIM, our model is consistent in two stages, has less chance to miss "similar" items in stage 1, and improves the performance significantly. We detail the differences of our model with others in Table 1.

3 TWIN IN KUAISHOU CTR PREDICTION

At the beginning, we first review the general preliminaries of the CTR prediction problem in Section 3.1. Then we describe the model architecture of our CTR prediction system in Kuaishou in Section 3.2. We further dig into details of our proposed *consistency-preserved*

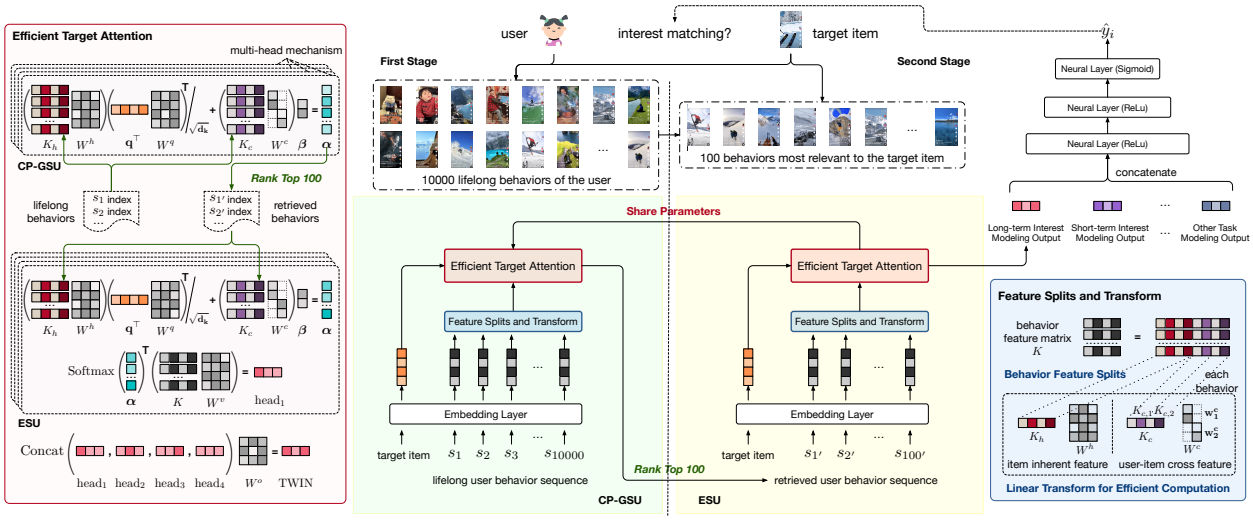


Figure 2: TWIN in Kuaishou's CTR prediction system. Different from conventional two-stage behavior modeling algorithms, TWIN adopts identical target-behavior relevance metric in CP-GSU and ESU, including not only identical network architecture (shown in the left) but also identical parameter values (shown in the middle bottom). This is challenging since MHTA was designed with high computational cost and thus only applicable to ESU (with 100 behaviors), not to CP-GSU (with 10^4 behaviors). We address this challenge by proposing: 1). efficient feature split and projection strategies that process item inherent features and user-item cross features in different manners (shown in the right bottom); 2). simplified target attention architecture that accelerates the efficiency of target attention through compressing cross features into bias terms (shown in the left).

lifelong user behavior modeling module, named TWo-stage Interest Network (TWIN), in Section 3.3. Finally, we introduce essential accelerating strategies which guarantee the successful online deployment of TWIN on the main traffic of Kuaishou in Section 3.4. The notations used are summarized in Table 2.

3.1 Preliminaries

The aim of CTR prediction is to predict the probability that a *user* would *click* an *item* given specific *contexts*. Accurate CTR prediction not only lifts the user experience by serving preferred contents, but also benefits the business effectiveness of content producers and platforms by reaching interested audiences. Consequently, CTR prediction has become the core component in various industrial RSs, especially short video recommendation platforms like Kuaishou.

CTR prediction is usually formulated as a binary classification problem, where the goal is to learn a predictor function $f : \mathbb{R}^d \to \mathbb{R}$ given a training dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{|\mathcal{D}|}, y_{|\mathcal{D}|})\}$. Specifically, $\mathbf{x_i} \in \mathbb{R}^{\mathbf{d}}$ is the feature vector of the i-th training sample (namely, the concatenation of the user, item and contexts features), and $y_i \in \{0, 1\}$ is the ground truth label denoting whether the user clicks (1) the item or not (0). The predicted CTR is calculated as:

$$\hat{y}_i = \sigma(f(\mathbf{x}_i)). \tag{1}$$

$\sigma(\cdot)$ is the sigmoid function that scales the prediction of f to $(0, 1)$.

The model is trained by minimizing the negative log-likelihood:

$$\ell(\mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i). \tag{2}$$

For conciseness, we omit the training sample index i in the following sections when no confusion is caused.

3.2 The Architecture of the CTR Prediction System in Kuaishou

We now illustrate the architecture of our CTR prediction system at Kuaishou. Details are shown in Fig 2.

3.2.1 The Embedding Layer. At the bottom, out model starts from a feature embedding layer that transforms raw features of a training sample to embedding vectors.

Without loss of generality, we assume that all features are in the categorical form after essential pre-processing. For a feature A with vocabulary size v_A, we first encode the categorical information into a one-hot / multi-hot code $\mathbf{x}_{\text{A,hot}} \in \{0, 1\}^{v_A}$. For example,

WeekDay=Mon $\Longrightarrow \mathbf{x}_{\text{WeekDay, hot}} = [1, 0, 0, 0, 0, 0, 0]^\top$,

Topic={Funny, Pet} $\Longrightarrow \mathbf{x}_{\text{Topic, hot}} = [..., 0, 1, 0, ..., 0, 1, 0...]^\top$.

Note that in most industrial systems, the vocabulary size (especially that of user / author / video ids) can easily scale to hundreds of millions. Thus, a common strategy is to transform the extreme high dimensional hot codes to low dimensional embeddings,

$$\mathbf{x}_{\text{A,emb}} = E_A \mathbf{x}_{\text{A,hot}}, \tag{3}$$

where $E_A \in \mathbb{R}^{d_A \times v_A}$ is the embedding dictionary of A, and d_A is the embedding dimension. In our system, we set the embedding dimension to 64 for id features with large vocabulary, and to 8 for others, such as video topic, video played timestamp.

In all the upper layers, we take the embedding vectors as input and thus omit the subscript "emb" for conciseness.

3.2.2 The Deep Networks. The overall architecture of our CTR prediction is shown in Fig 2. The upper module, consisting of many stacked neural networks and ReLUs, acts as a mixer that learns the interaction between the outputs of three intermediate modules:

- **TWIN**, the proposed consistency-preserved life-long user behavior modeling module. TWIN extracts user interest through two cascading stages of behavior modeling sub-modules: 1). Consistency-Preserved General Search Unit (**CP-GSU**) which performs a coarse search for 100 most relevant behaviors from tens of thousands of long term historical behaviors; 2). Exact Search Unit (**ESU**) which adopts an attention mechanism over the 100 finalists of CP-GSU to capture the exact user interest.
  Different from conventional algorithms that usually consist of a "light" GSU and a "heavy" ESU, our proposed CP-GSU follows the identical relevance evaluation metric as that of ESU, making the two cascading stages *TWINS*. Consequently, CP-GSU consistently retrieves items that are considered important by ESU, maxmizing the behavior modeling effectiveness.
- **Short-term behavior modeling** which extracts user interests from the 50 most recent behaviors. This module focuses on the user's short term interest in the latest few days, and acts as a strong complement to TIM.
- **Others Task Modelings**. Besides behavior modeling, we also concatenate the outputs of various other task modelings, which model the user's gender, age, occupation, location, the video's duration, topic, popularity, quality, and contexts features such as the played date, timestamp, page position, etc.

3.3 TWIN: TWo-stage Interest Network

We name the proposed algorithm TWIN to highlight that CP-GSU follows the identical relevance evaluation metric as that of ESU. Note that this consistency is nontrivial because:

- Effective behavior modeling algorithms are usually based on Multi-Head Target Attention (MHTA) [25], which precisely captures user interest by emphasising target relevant behaviors. Unfortunately, due to the high computational complexity, the applicable behavior sequence length of MHTA is mostly limited to a few hundreds.
- To exhaustively capture user's long-term interest, CP-GSU should cover user behaviors in the last several months, which could easily reach tens of thousands. This sequence length is far beyond the capacity of conventional MHTA given the strict low latency requirements of online systems.

This section aims to answer this key question: how to improve the efficiency of MHTA so that we can extend it from ESU to CP-GSU, or namely from a sequence length of hundreds to a sequence length of at least tens of thousands?

3.3.1 Behavior Feature Splits and Linear Projection. Following the standard notations of the MHTA [25], we define the features of a length L behavior sequence $[s_1, s_2, ..., s_L]$ as matrix K, where each row denotes the features of one behavior. In practice, the linear projection of K in the attention score computation of MHTA is the key computational bottleneck that hinders the application of MHTA on extremely-long user behavior sequences. We thus propose the followings to reduce its complexity.

We first split the behavior features matrix K into two parts,

$$K \triangleq \begin{bmatrix} K_h & K_c \end{bmatrix} \in \mathbb{R}^{L\times(H+C)}, \tag{4}$$

We define $K_h \in \mathbb{R}^{L\times H}$ as the *inherent* features of behavior items (e.g. video id, author, topic, duration) which are independent of the specific user / behavior sequence, and $K_c \in \mathbb{R}^{L\times C}$ as the user-item cross features (e.g. user click timestamp, user play time, clicked page position, user-video interactions). This split allows high efficient computation of the following linear projection $K_h W^h$ and $K_c W^c$.

For the inherent features K_h, although the dimension H is large (64 for each id feature), the linear projection is actually not costly. The inherent features of a specific item are shared across users / behavior sequences. With essential caching strategies, $K_h W^h$ could be efficiently "calculated" by a look up and gathering procedure. Details of online deployment will be introduced in Section 3.4.

For the user-item cross features K_c, caching strategies are not applicable because: 1). Cross features describe the interaction details between a user and a video, thus not shared across users behavior sequences. 2). Each user watches a video for at most once. Namely, there is no duplicated computation in projecting cross features. We thus reduce the computational cost by simplifying the linear projection weight.

Given J cross features, each with embedding dimension 8 (since not id features with huge vocabulary size). We have $C = 8J$. We simplify the linear projection as follows,

$$K_c W^c \triangleq \begin{bmatrix} K_{c,1}\mathbf{w}_1^c, & \dots & , K_{c,J}\mathbf{w}_J^c \end{bmatrix}, \tag{5}$$

where $K_{c,j} \in \mathbb{R}^{L\times 8}$ is a column-wise slice of K_c for the j-th cross feature, and $\mathbf{w}_j^c \in \mathbb{R}^8$ is its linear projection weight. Using this simplified projection, we compress each cross feature into one dimension, i.e., $K_c W^c \in \mathbb{R}^{L\times J}$. Note that this simplified projection is equivalent to restricting W^c to a diagonal block matrix.

3.3.2 Complexity Analysis. In the conventional MHTA, the time complexity of linear projection of K, namely from dimension $L \times (H + C)$ to $L \times d_{out}$, is $O(L \times (H + C) \times d_{out})$.

While in our MHTA for TWIN, the item inherent features $K_h W^h$ is pre-computed and efficiently gathered in $O(L)$, which is independent of the dimension H. And the user-item cross features $K_c W^c$ is reduced to low dimensional computation of $O(L \times C)$.

Since $C \ll H$, and $C \ll d_{out}$, it is this theoretical acceleration that allows the consistent implementation of MHTA in both CP-GSU and ESU.

3.3.3 Target Attention in TWIN. Based on the linear projection of behaviors $K_h W^h$ and $K_c W^c$, we now define the target-behavior relevance metric that is used uniformly in both CP-GSU and ESU.

Without loss of generality, we assume that there has been no interaction between the user and the target item and denote the target item's inherent features as $\mathbf{q} \in \mathbb{R}^H$. With proper linear projection W^q, the relevance score $\boldsymbol{\alpha} \in \mathbb{R}^L$ between the target item and the historical behaviors is calculated as:

$$\boldsymbol{\alpha} = \frac{(K_h W^h)(\mathbf{q}^\top W^q)^\top}{\sqrt{d_k}} + (K_c W^c)\boldsymbol{\beta}, \tag{6}$$

where d_k is the dimension of projected query and key. This relevance score is calculated by the inner product between the *query*, i.e., the inherent features of the target, and the *keys*, i.e., the inherent features of behaviors. Additionally, the cross features, since compressed to 1 dimension, serve as bias terms. We use $\boldsymbol{\beta} \in \mathbb{R}^J$ as the learnable parameter for the relative importance of cross features.

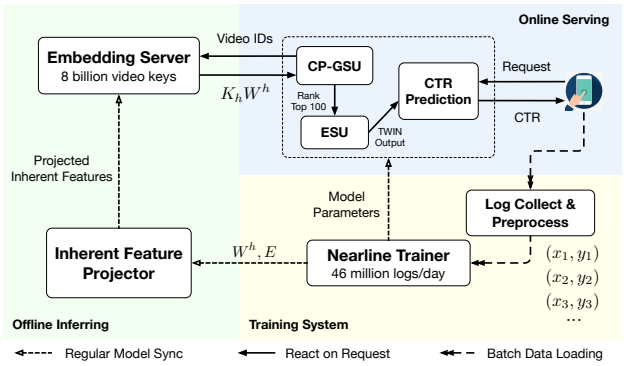


Figure 3: The Deployment of TWIN in Online CTR Prediction System. We propose essential precomputing & caching strategies to reduce the key computational bottleneck, the linear projection of 10^4 behaviors' inherent features. With proper frequency control, we cut off tailed videos and limit the size of candidate video pool to 8 billion. As a result, the *inherent feature projector* can cyclically refresh the linear projection of all candidate videos every 15 minutes, minimizing the accuracy loss from caching. And the *embedding server* which stores the projection of 8 billion candidate videos, can cover 97% of the online requests and achieve satisfactory effectiveness given limited computational resources.

In CP-GSU, this relevance score $\boldsymbol{\alpha}$ is used to cut-off the $L = 10^4$ long-term historical behaviors to 100 most relevant ones. And in ESU, we perform a weighted average pooling over the 100 finalists:

$$\text{Attention}(\mathbf{q}^\top W^q, K_h W^h, K_c W^c, KW^v) = \text{Softmax}(\boldsymbol{\alpha})^\top KW^v, \quad (7)$$

where W^v is a projection matrix. We slightly abuse the notation by setting $L = 100$. This projection KW^v is only performed over 100 behaviors and thus can be conducted efficiently online. We do not need to split K as we did when computing $\boldsymbol{\alpha}$ for 10^4 behaviors.

To jointly attend to information from different representation subspaces, we adopt 4 heads in our MHTA. Thus the final output of TWIN is defined as

$$\begin{aligned} \text{TWIN} &= \text{Concat}(\text{head}_1, ..., \text{head}_4) W^o, \\ \text{head}_a &= \text{Attention}(\mathbf{q}^\top W_a^q, K_h W_a^h, K_c W_a^c, KW_a^v), a \in \{1, ..., 4\}, \end{aligned} \quad (8)$$

W^o is a projection that learns relative importance between heads.

3.4 System Deployment

We deploy TWIN on the ranking system of Kuaishou, serving the main traffic of 346 million daily active users. In this section, we introduce our hands-on experience in the deployment. Details of our system architecture is shown in Fig 3.

3.4.1 Training System. Our TWIN module is trained jointly with the whole CTR prediction model on Kuaishou's large-scale distributed nearline learning system.

Every day, hundreds of millions of users visit Kuaishou, watch and interact with short videos, and leave 46 billion watch and interaction logs per day. Each log is collected, preprocessed in realtime and used for model training in less than 8 minutes. This nearline training system incrementally updates the model parameters using the latest knowledge from user-video interactions that take place in less than 8 minutes ago.

In addition, our message queue system continuously synchronizes the latest parameter values from the training system to the offline inferring and online serving systems, once every 5 minutes. This synchronization ensures that the online CTR prediction service is always based on the update-to-date model.

3.4.2 Offline Inferring. The offline inferring system aims to accelerate the online serving by providing a lookup service. When receiving lookup keys, a batch of video ids, this service returns the lookup values, the corresponding projected inherent features in concatenation, i.e. $K_h W_a^h$ for all heads $a \in \{1, ..., 4\}$.

Specifically, the offline inferring system consists of two parts. 1). *An inherent feature projector*, which cyclically pre-computes the linear projection of inherent features using the latest embeddings and TWIN parameters W_a^h synchronized from the training system. With proper frequency control, this projector can refresh the projected inherent features of an 8 billion scale candidate video pool every 15 minutes, minimizing the accuracy loss from caching 2). *An embedding server*, which stores results of the inherent feature projector into a key-value structure and provides the aforementioned key lookup service. By cutting off tailed videos, the 8 billion keys can cover 97% of online request, balancing efficiency and effectiveness.

3.4.3 Online Serving. Once a request is received, the online serving system queries the offline inferring system for the projected inherent features $K_h W_a^h$, and calculates the other parts of Eq 6 in realtime for the user-item relevance score $\boldsymbol{\alpha}$. We then sort the behaviors by the descending order of the scale in $\boldsymbol{\alpha}$, and select the top-100 behaviors for ESU. This design reduces the computational bottleneck of TWIN, i.e., linear projection of K_h with 10^4 rows, by 99.3% in practice.

Note that ESU with only 100 behaviors is light enough for all calculations to be conducted in realtime using up-to-date parameters synchronized from the training system. As the result, the $K_h W^h$ calculated by ESU is slightly more up-to-date than that in CP-GSU, which further lifts the performance of our TA mechanism.

With accelerated design, TWIN is successfully deployed on the ranking system of Kuaishou, serving the main traffic of 346 million active users, with a peak request of 30 million videos per second.

4 EXPERIMENT

In this section, we detail the offline and online experiments on real industrial data to evaluate our proposed method with the purpose of answering the following four research questions (RQs).

- **RQ1:** How does TWIN perform in the offline evaluations compared to other SOTAs in lifelong user behavior modeling?
- **RQ2:** How *consistent* can TWIN achieve, compared to other SOTAs? Or namely, why TWIN works?
- **RQ3:** How does the effectiveness of TWIN change as the length of the user behavior sequence grows?
- **RQ4:** What are the effects of key components and different implementations in the proposed method?
- **RQ5:** How does TWIN perform in real online RS?

Table 3: Statistics of the industrial dataset constructed on the daily collected user logs from Kuaishou.

Data	Field	Size
Daily Log Info	Users	345.5 million
	Videos	45.1 million
	Samples	46.2 billion
	Average User Actions	133.7 / day
Historical Behaviors	Average User Behaviors	14.5 thousand
	Max User Behaviors	100 thousand

4.1 Dataset

To evaluate TWIN in real-world situation for lifelong user behavior modeling, we need a large scale CTR prediction dataset with rich user historical behaviors that can ideally scale up to tens of thousands behaviors per user. Unfortunately, existing public datasets are either relatively small or lack of sufficient user historical behaviors. For example, in the widely used Amazon dataset [11, 13, 17] each user has less than ten historical behaviors on average. In the Taobao dataset [18, 19, 34–36] the average sequence length of user behaviors is at most 500. We thus collect an industrial dataset from Kuaishou, one of the top short-video sharing platforms in China.

We construct samples from daily user logs, with user's click-throughs as labels. As shown in Table 3, the size of daily active users on Kuaishou is around 346 million. Every day 45 million short videos are posted and these videos are played 46 billion times in total. On average, each user watches 133.7 short videos per day. To utilize rich behavior information, we collect full user historical behaviors from older logs back to months ago. On average, each user watched 14,500 videos in the past six months, which provides models a testbed with fertile user historical behaviors to learn from. We cutoff the maximum user behavior sequence length to 100,000, which is about the annual total number of views for heavy users.

4.2 Baselines

To demonstrate effectiveness, we compare TWIN with the following SOTA lifelong user behaviors modeling algorithms.

- **Avg-Pooling**, the average pooling of user lifelong behaviors.
- **DIN** [35], the most widely adopted approach for short-term behavior modeling, which leverages TA for target-specific interests.
- **SIM Hard** [19]. GSU selects behaviors from the same category with the target item and ESU follows the TA in DIN. In our scenarios, the total number of the video categories is 37.
- **ETA** [3]. Locality-sensitive hash (LSH) is used to generate a hash signature for the target video and behaviors. Then GSU uses Hamming distance as the target-behavior relevance metric.
- **SDIM** [1]. GSU selects behaviors with the same hash signature as the target video through multi-round hash collision. In the original paper, ESU linearly aggregates sampled behaviors from multiple rounds to obtain user interests. In our experiments, a more powerful TA is adopted in ESU for fair comparison.
- **SIM Cluster**. Since "category" requires expensive human annotations and is usually unreliable in short video scenarios, we implement SIM Cluster as an improved variant of SIM Hard. We group videos into 1,000 clusters based on pre-trained embeddings. GSU retrieves behaviors from the same cluster as the target item.
- **SIM Cluster+** is a refinement of SIM Cluster, where the number of clusters is expanded from 1,000 to 10,000.
- **SIM Soft** [19]. GSU uses inner product score of videos' pre-trained embeddings to retrieve relevant behaviors. Inner product is a more refined retrieval method than Hamming distance and hash collision, but with higher computational cost.

In summary, ETA and SDIM adopt end-to-end training methods, but use rough retrieval methods to avoid high-complexity calculations. SIM Cluster (+), and SIM Soft use refined retrieval methods, but the compromise is that they have to use pre-trained embeddings and generate offline inverted index in advance. Note that SIM Hard and SIM Soft are simultaneously proposed [19], but SIM Soft has not yet been defeated by the follow-up work ETA and SDIM [1, 3]. We do not compare with UBR4CTR [20] because its iterative training is not suitable for our streaming scenario. Furthermore, UBR4CTR is confirmed performing worse than SIM Hard and ETA [3].

4.3 Experimental Setting

We use samples in 23 consecutive hours of a day as training data and ones in the following hour for test. We evaluate all algorithms in 5 consecutive days and report the averaged performance over days. For offline evaluation, we use two widely adopted metrics: AUC and GAUC. AUC signifies the probability that a positive sample's score is higher than that of a negative one, reflecting a model's ranking ability. GAUC performs a weighted average over all user's AUC, and the weights are set to the number of samples of this user. GAUC eliminates the bias among users and evaluates model performance at a finer and fair granularity.

For two-stage models, we use the recent 10,000 behaviors as input of GSU and retrieve 100 behaviors for TA in ESU. For DIN, we use the most recent 100 behaviors due to its bottleneck in processing long sequences. Though CP-GSU of TWIN uses four heads in the attention score computation, we recursively traverses the top ranked items by the four heads until it collects 100 unique behaviors. For all models, the embedding layer uses the AdaGrad optimizer and the learning rate is 0.05. DNN parameters are updated by Adam with learning rate 5.0e-06. The batch size is set as 8192.

4.4 Overall Performance (RQ1)

Table 4 shows the performance of all models. Note that due to the large amount of users and samples in our dataset, an improvement of 0.05% in AUC and GAUC in the offline evaluation is significant enough to bring online gains for the business.

First, TWIN significantly outperforms all baselines, especially two-stage SOTAs with inconsistent GSU. This validates the key advantage of TWIN in life-long behavior modeling, i.e., the *powerful* and *consistent* TA in CP-GSU. Specifically, CP-GSU precisely retrieves behaviors that are considered highly relevant by ESU, saving ESU's precious computational resources for the most important user information. While in others, the ineffective and inconsistent GSU may miss important behaviors but introduce noisy ones, degrading TA performance. In addition, the gain from Avg-pooling to DIN shows the ability of TA in retrieving effective

Table 4: Offline comparison with SOTAs (RQ1). We report the *mean* and standard deviation (*std*) over 5 consecutive days. The best and second-best results are highlighted in bold and underlined respectively . Note that the improvement of TWIN over the best compared model in AUC is 0.29% and that in GAUC is 0.51%. These improvements are much larger than 0.05%, a value enough to bring online benefits.

Method	AUC (*mean* ± *std*) ↑	GAUC (*mean* ± *std*) ↑
Avg-Pooling	0.7855 ± 0.00023	0.7168 ± 0.00019
DIN	0.7873 ± 0.00014	0.7191 ± 0.00012
SIM Hard	0.7901 ± 0.00016	0.7224 ± 0.00021
ETA	0.7910 ± 0.00004	0.7243 ± 0.00011
SIM Cluster	0.7915 ± 0.00017	0.7253 ± 0.00018
SDIM	0.7919 ± 0.00009	0.7267 ± 0.00006
SIM Cluster+	0.7927 ± 0.00009	0.7275 ± 0.00011
SIM Soft	<u>0.7939 ± 0.00014</u>	<u>0.7299 ± 0.00013</u>
TWIN	**0.7962 ± 0.00008**	**0.7336 ± 0.00011**
Improvement	+0.29%	+0.51%

information. And the gain from DIN to other two-stage SOTAs verifies the necessity of modeling long behaviors. These two together support our motivation: extending TA to long sequences.

Second, only end-to-end training is not enough. We observe that TWIN apparently outperforms ETA and SDIM, two strong baselines whose embeddings in GSU are also trained in end-to-end manner. Specially, ETA uses LSH & Hamming distance and SDIM uses multiround hash collision. Both GSU strategies are less precise compared to TA and inconsistent with the target-behavior relevance metric used in their ESUs. While CP-GSU in TWIN is not only trained end-to-end, but also consistent with TA in ESU. This shows that a precise relevance metric is crucial to GSU, validating our advantage over the existing end-to-end algorithms.

Third, modeling lifelong behaviors at a finer granularity is effective. We compare the variants of SIM: SIM Hard with 37 categories, SIM Cluster(+) with 1,000 / 10,000 clusters and SIM Soft where the target-behavior relevance score is calculated individually for each behavior. We observe consistent performance improvement as every finer granularity retrieval method is used in GSU. This is because GSU retrieves behaviors more accurately when it can capture the relevance score between videos more finely. From this perspective, we further attribute our advantage over SIM Soft to the fact that TWIN adopts more accurate relevance metric. Note that the poor performance of SIM Hard is partly because the video category used for retrieval is sometimes not reliable in short video RS, which is a little surprising for engineers from e-shop industry.

4.5 Consistency Analysis (RQ2)

As claimed, our superior behavior modeling ability comes from the consistent relevance metrics in CP-GSU and ESU. But how consistent we really achieve (Figure 4)?

For each well-trained two-stage model, we reuse the parameters of its ESU as its Oracle to retrieve "the real top-100" from 10,000 behaviors. In other words, these real top-100 are the ground-truth that ESU considers to be really important. We then traverse the GSU output size of all compared algorithms from 10 to 10^4 to examine how many outputs hit the real top-100. Note that each compared algorithm has its own Oracle and top-100. But we only plot one Oracle curve since all Oracles perfectly hit ground truth.

Figure 4: Consistency between GSU & ESU (RQ2). Oracle uses ESU's relevance metric on 10^4 behaviors to get "the real top-100". Among the 100 return by GSU of SIM Hard, only 40 hits. TWIN lifts the hit number to 94. Theoretically, TWIN could achieve 100% hit rate since we adopt consistent GSU. However, our practical performance is limited by the deployment constrain, the 15 minutes refresh delay in caching.

SIM Soft achieves an improvement in retrieval consistency benefiting from a more precise retrieval strategy in GSU. Further, TWIN achieves 94 hits when returning 100 behaviors, which validates our advantage in preserving the consistency between two stages. Note that this is the most noteworthy value, since 100 is the upper bound for ESU input, given the constraints of inference time and computational complexity of TA. We did not reach our theoretical 100% consistency in practice due to refresh delay in caching, as described in Section 3.4.2. In addition, we show some case studies of retrieval consistency in the Appendix.

Based on the above results, we speculate that CP-GSU has a stronger ability to match user interests with the target video than conventional GSUs. This is attributed to the consistency. Via sharing the same structure and parameters with ESU, CP-GSU is able to accurately judge and retrieve videos with similar content. In addition, as the CP-GSU parameters are updated in real time, the model can capture the user's dynamically evolving interests.

4.6 Effects of Behavior Length (RQ3)

We aim to test the effectiveness of TWIN under different behavior sequence lengths and to further tap the potential of TWIN. Note that only input sequence lengths of GSUs are changed and the output length is kept 100. Results are shown in Figure 5.

We observe that 1). TWIN performs the best consistently, and 2) the performance gap between TWIN and other methods becomes larger as the sequence length increases. This indicates that TWIN has a better effectiveness in modeling extremely long sequences.

4.7 Ablation Study (RQ4)

We conduct ablation studies by applying different operations to TWIN to evaluate the contribution of our key model designs: 1) the consistency between two stages, and 2) efficient MHTA.

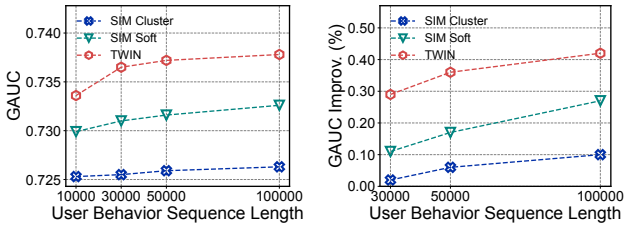


Figure 5: Performance when GSU Takes Different Behavior Sequence Lengths (RQ3). Left: GAUC values. Right: Relative GAUC improvements over length = 10,000. As the sequence length increases, all model's performance improves and the performance gap between TWIN and baselines increases.

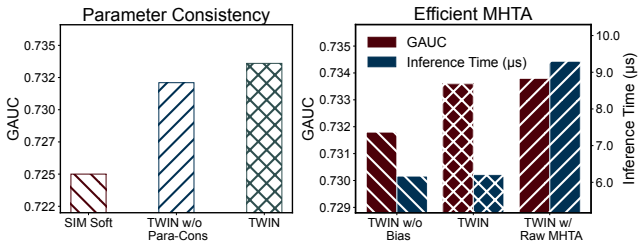


Figure 6: Effects of key Components in TWIN (RQ4). Left: *TWIN w/o Para-Cons* (consistent network structure + inconsistent parameters) outperforms SIM Soft (both inconsistent), but performs worse than TWIN (both consistent). Right: directly removing the user-item cross features (*TWIN w/o Bias*) saves little computation but leads to a significant drop in GAUC compared to TWIN. Compared to TWIN with the original MHTA (*TWIN w/ Raw MHTA*), compressing the item inherent features to biases for efficiency (TWIN) hardly harms the performance but greatly speeds up inference time.

TWIN preserves the consistency between two stages in two folds: the network structure and parameters. To study the benefits brought by each, we implement a variant named *TWIN w/o Para-Con*, that doesn't preserve the parameter consistency. Specially, we first train an auxiliary model *TWIN-aux* which uses the identical network structure and training data with TWIN but is trained separately. We then synchronize the GSU parameters from *TWIN-aux* to *TWIN w/o Para-Con*. This is to ensure that *TWIN w/o Para-Con* is still updated in real time and that the gap between TWIN and *TWIN w/o Para-Con* are all caused by parameter inconsistency.

As shown in Figure 6 (left), *TWIN w/o Para-Con* performs significantly better than SIM Soft (inconsistent in both structure and parameters) but slightly worse than TWIN. It indicates that both structure consistency and parameter consistency are beneficial, but network structure consistency contributes more.

To compute MHTA efficiently for the industrial deployment, we split user behavior features and compress each user-item cross feature into a one-dimensional bias term. To study the impact of such modification and the benefits of preserving user-item cross features in the attention computation, we implement two variants and compare their performances and inference time: TWIN with the original MHTA where a straightforward linearly projection KW is used and no feature split is conducted, and TWIN without using the user-item cross features in MHTA, which are respectively abbreviated as *TWIN w/ Raw MHTA* and *TWIN w/o Bias*.

As shown in Figure 6 (right), TWIN beats *TWIN w/o Bias* significantly and performs almost the same as *TWIN w/ Raw MHTA*, validating that our proposed modification to MHTA hardly compromises the performance. Regarding to the computational cost, as caching is inapplicable for *TWIN w/ Raw MHTA* when the user-item cross features are used in the linear projection of K (details in Section 3.3.2), the inference time of *TWIN w/ Raw MHTA* significantly increases. In contrast, removing the user-item cross features (TWIN w/o Bias) does not save much computation, yet harms performance.

Table 5: The relative Watch Time improvement of TWIN in online A/B test compared with SIM Hard and SIM Soft (RQ5). In Kuaishou's scenario, 0.1% increase is a significant improvement that brings great business effectiveness.

Scenarios	Featured-Video Tab	Discovery Tab	Slide Tab
v.s. SIM Hard	+4.893%	+3.712%	+6.249%
v.s. SIM Soft	+2.778%	+1.374%	+2.705%

4.8 Online Result (RQ5)

To evaluate the online performance of TWIN, we conduct strict online A/B tests on Kuaishou's short-video recommendation platform. Table 5 compares the performance of TWIN with SIM Hard and SIM Soft under three representative business scenarios in Kuaishou (**Featured-Video** Tab, **Discovery** Tab, and **Slide** Tab). Different from e-commerce, where the commonly used online evaluation metrics are CTR and GMV, the short-video recommendation scenarios usually use Watch Time, which measures the total amount of time users spend on viewing the videos. As shown, TWIN remarkably outperforms SIM Hard and SIM Soft in all scenarios. As 0.1% increase in Watch Time is considered as an effective improvement in Kuaishou, TWIN achieves significant business gain.

5 CONCLUSION

To address the inconsistency issue of conventional life-long behavior modeling algorithms, we proposed a consistency-preserved TWo-stage Interest Model, which successfully extended the effective but computational expensive MHTA from ESU to CP-GSU, or namely from a sequence length of 100 to length $10^4 - 10^5$. Specifically, we designed novel MHTA mechanism as well as highly efficient infrastructure, including behavior features splits & compression, pre-computing & caching, online training & parameter synchronization. We accelerated the computational bottleneck by 99.3%, which contributed to the successful deployment of TWIN on Kuaishou, serving the main traffic of hundreds of millions of active users. The consistency between two stages, together with the effective TA-based relevance metric in CP-GSU, maximized the retrieval effectiveness of behavior modeling and significantly lifted the performance of CTR prediction. To the best of our knowledge, TWIN is the first to achieve consistency in the two-stage life-long behavior modeling problem.

REFERENCES

[1] Yue Cao, XiaoJiang Zhou, Jiaqi Feng, Peihao Huang, Yao Xiao, Dayao Chen, and Sheng Chen. 2022. Sampling Is All You Need on Modeling Long-Term User Behaviors for CTR Prediction. *arXiv preprint arXiv:2205.10249* (2022).

[2] Junxuan Chen, Baigui Sun, Hao Li, Hongtao Lu, and Xian-Sheng Hua. 2016. Deep ctr prediction in display advertising. In *Proceedings of the 24th ACM international conference on Multimedia*. 811–820.

[3] Qiwei Chen, Changhua Pei, Shanshan Lv, Chao Li, Junfeng Ge, and Wenwu Ou. 2021. End-to-End User Behavior Retrieval in Click-Through RatePrediction Model. *arXiv preprint arXiv:2108.04468* (2021).

[4] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[9] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep session interest network for click-through rate prediction. *arXiv preprint arXiv:1905.06482* (2019).

[10] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[11] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*. 161–169.

[12] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM conference on recommender systems*. 43–50.

[13] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[14] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-interest network with dynamic routing for recommendation at Tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2615–2623.

[15] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1754–1763.

[16] Hongche Liu and MS Zamanian. 2007. Framework for selecting and delivering advertisements over a network based on combined short-term and long-term user behavioral interests. US Patent App. 11/225,238.

[17] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[18] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2671–2679.

[19] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2685–2692.

[20] Jiarui Qin, Weinan Zhang, Xin Wu, Jiarui Jin, Yuchen Fang, and Yong Yu. 2020. User behavior retrieval for click-through rate prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2347–2356.

[21] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[22] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *Advances in neural information processing systems* 30 (2017).

[23] Yang Song, Ali Mamdouh Elkahky, and Xiaodong He. 2016. Multi-rate deep learning for temporal recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 909–912.

[24] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[26] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 1–7.

[27] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the Web Conference 2021*. 1785–1797.

[28] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv preprint arXiv:1708.04617* (2017).

[29] Zhibo Xiao, Luwei Yang, Wen Jiang, Yi Wei, Yi Hu, and Hao Wang. 2020. Deep multi-interest network for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2265–2268.

[30] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 729–732.

[31] Li Zhang, Weichen Shen, Jianhang Huang, Shijian Li, and Gang Pan. 2019. Field-aware neural factorization machine for click-through rate prediction. *IEEE Access* 7 (2019), 75032–75040.

[32] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep learning over multi-field categorical data. In *European conference on information retrieval*. Springer, 45–57.

[33] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[34] 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 *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

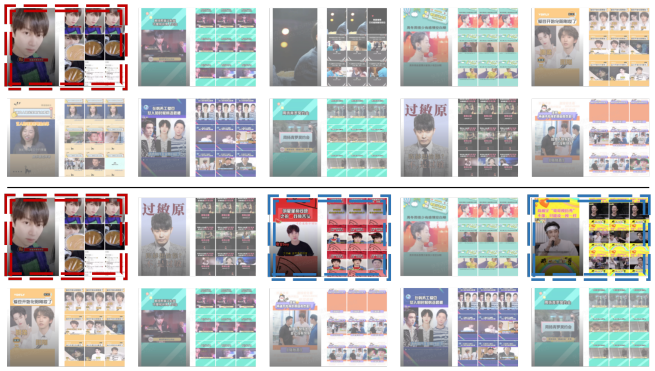
[35] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.

[36] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1079–1088.

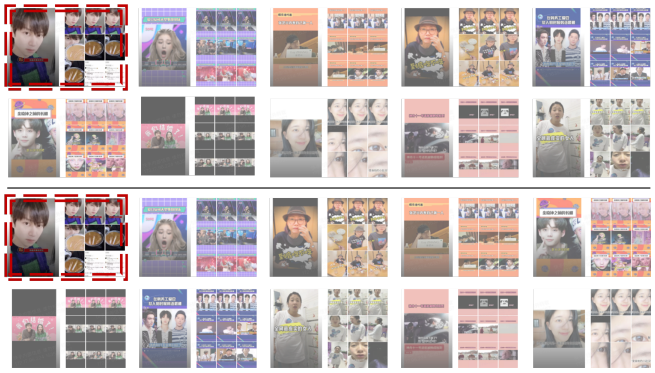
A APPENDIX

A.1 Case Study

We conduct case studies as Figure 7 and 8. The red box in Figure 7 shows a video of an actor performing a funny joke. We observe that when using it as the target item, GSU in SIM Soft only retrieves the videos about actors in Figure 7(a). Moreover, the most relevant ones judged by ESU are thought as not the most relevant in GSU of SIM Soft, which are marked in blue box. This indicates that the top results by GSU may miss the really relevant ones. Besides the videos about actors, CP-GSU of TWIN is able to retrieve videos of the funny topic and rank the history behaviors consistently with ESU, as shown in 7(b). The red box in Figure 8 presents a target item about mukbang. Figure 8(a) and Figure 8(b) show the most relevant items retrieved by SIM Soft and TWIN from the user behavior history respectively. The results show that the most relevant video retrieved by GSU in SIM Soft is of the mukbang topic, whereas CP-GSU in TWIN retrieves the item relevant in content and plot, yet not exactly of the mukbang topic.



(a) GSU (upper part) and ESU (bottom part) of SIM Soft



(b) CP-GSU (upper part) and ESU (bottom part) of TWIN

Figure 7: Differences between the Top-9 relevant items retrieved by GSU and ESU (Red box for the target item). In the results of SIM Soft, the most relevant ones judged by ESU are thought as not relevant in GSU (marked in blue box). This indicates that the top results by GSU may miss the really relevant ones. Differently, TWIN retrieves consistent items in CP-GSU and ESU stages thanks to the identical measurement metric and shared parameters. These local mis-orderings are caused by our deployment constrains (e.g. the delayed caching strategies), where the average difference of relevance score α is less than 4%.

(a) SIM Soft

(b) TWIN

Figure 8: Differences between the most relevant item retrieved by SIM Soft and TWIN. When the target item is a video about mukbang (photos on the left) of a couple, the most relevant video retrieved by GSU of SIM Soft is mukbang of a man. In contrast, the most relevant one retrieved by CP-GSU of TWIN (highlighted in the blue box) shows a couple eating lunch, which is more relevant in content to the target item.