

TWIN V2: Scaling Ultra-Long User Behavior Sequence Modeling for Enhanced CTR Prediction at Kuaishou

Zihua Si*
Renmin University of China
Beijing, China
zihua_si@ruc.edu.cn

Lin Guan
Kuaishou Technology Co., Ltd.
Beijing, China
guanlin03@kuaishou.com

Zhongxiang Sun*
Renmin University of China
Beijing, China
sunzhongxiang@ruc.edu.cn

Xiaoxue Zang
Kuaishou Technology Co., Ltd.
Beijing, China
zangxiaoxue@kuaishou.com

Jing Lu
Kuaishou Technology Co., Ltd.
Beijing, China
lvjing06@kuaishou.com

Yiqun Hui
Kuaishou Technology Co., Ltd.
Beijing, China
huiyiqun@kuaishou.com

Xingchao Cao
Kuaishou Technology Co., Ltd.
Beijing, China
caoxingchao@kuaishou.com

Zeyu Yang
Kuaishou Technology Co., Ltd.
Beijing, China
yangzeyu03@kuaishou.com

Yichen Zheng
Kuaishou Technology Co., Ltd.
Beijing, China
zhengyichen@kuaishou.com

Dewei Leng
Kuaishou Technology Co., Ltd.
Beijing, China
lengdewei@kuaishou.com

Kai Zheng
Kuaishou Technology Co., Ltd.
Beijing, China
zhengkai@kuaishou.com

Chenbin Zhang
Kuaishou Technology Co., Ltd.
Beijing, China
alczhang13@yeah.net

Yanan Niu
Kuaishou Technology Co., Ltd.
Beijing, China
niuyanan@kuaishou.com

Yang Song
Kuaishou Technology Co., Ltd.
Beijing, China
ys@sonyis.me

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

ABSTRACT

The significance of modeling long-term user interests for CTR prediction tasks in large-scale recommendation systems is progressively gaining attention among researchers and practitioners. Existing work, such as SIM and TWIN, typically employs a two-stage approach to model long-term user behavior sequences for efficiency concerns. The first stage rapidly retrieves a subset of sequences related to the target item from a long sequence using a search-based mechanism namely the General Search Unit (GSU), while the second stage calculates the interest scores using the Exact Search Unit (ESU) on the retrieved results. Given the extensive length of user behavior sequences spanning the entire life cycle, potentially reaching up to 10^6 in scale, there is currently no effective solution for fully modeling such expansive user interests. To overcome this issue, we introduced TWIN-V2, an enhancement of TWIN, where a *divide-and-conquer* approach is applied to compress life-cycle behaviors

and uncover more accurate and diverse user interests. Specifically, a hierarchical clustering method groups items with similar characteristics in life-cycle behaviors into a single cluster during the offline phase. By limiting the size of clusters, we can compress behavior sequences well beyond the magnitude of 10^5 to a length manageable for online inference in GSU retrieval. Cluster-aware target attention extracts comprehensive and multi-faceted long-term interests of users, thereby making the final recommendation results more accurate and diverse. Extensive offline experiments on a multi-billion-scale industrial dataset and online A/B tests have demonstrated the effectiveness of TWIN-V2. Under an efficient deployment framework, TWIN-V2 has been successfully deployed to the primary traffic that serves hundreds of millions of daily active users at Kuaishou.

*Work done during their internships at Kuaishou.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).
CIKM '24, October 21–25, 2024, Boise, ID, USA.

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0436-9/24/10...\$15.00
<https://doi.org/10.1145/3627673.3680030>

ACM Reference Format:

Zihua Si, Lin Guan, Zhongxiang Sun, Xiaoxue Zang, Jing Lu, Yiqun Hui, Xingchao Cao, Zeyu Yang, Yichen Zheng, Dewei Leng, Kai Zheng, Chenbin Zhang, Yanan Niu, Yang Song, and Kun Gai. 2024. TWIN V2: Scaling Ultra-Long User Behavior Sequence Modeling for Enhanced CTR Prediction at Kuaishou. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, October 21–25, 2024, Boise, ID, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3627673.3680030>

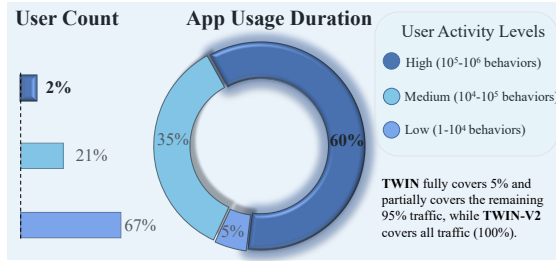


Figure 1: The Proportion of user count and app usage duration at Kuaishou. We grouped users based on the number of videos they have viewed over the past three years. Highly active users, though a small percentage (2%), contribute the most usage duration (60%).

1 INTRODUCTION

Click-Through Rate (CTR) prediction is crucial for internet applications. For instance, Kuaishou¹, one of China’s largest short-video sharing platforms, has employed CTR prediction as a core component of its ranking system. Recently, much effort [1, 2, 24] has been devoted to modeling users’ long-term historical behavior in CTR prediction. Due to the extensive length of behaviors across the life cycle, modeling life-cycle user behaviors presents a challenging task. Despite ongoing efforts to extend the length of historical behavior modeling in existing research, no method has yet been developed that can model a user’s entire life cycle, encompassing up to 1 million behaviors within an app.

Modern industrial systems [1, 13], to utilize as long a user history as possible within a controllable inference time, have adopted a two-stage approach. Specifically, in the first stage, a module called the General Search Unit (GSU) is used to filter long-term historical behaviors, selecting items related to the target item. In the second stage, a module called the Exact Search Unit (ESU) further processes the filtered items, extracting the user’s long-term interests through target attention. The coarse-grained pre-filtering by the GSU enables the system to model longer historical behaviors at a faster speed during online inference. Many approaches have tried different GSU structures to improve the accuracy of pre-filtering, such as ETA [3], SDIM [1] and TWIN [2].

Despite their effectiveness, existing first-stage GSUs generally have length limitations and cannot model user behaviors throughout the life cycle. For instance, SIM, ETA, and SDIM can filter historical behaviors up to a maximum length of 10^3 . TWIN extends the maximum length at the 10^4 to 10^5 level. During deployment, TWIN utilizes the recent 10,000 behaviors as inputs for GSU. Unfortunately, these 10,000 behaviors only cover the user’s history for the last 3–4 months in the Kuaishou app, failing to encompass the entire life cycle of user behavior. As illustrated in Figure 1, we analyzed user behavior in the Kuaishou app over the past three years. The medium and high user groups can view between 10^4 to 10^6 videos over three years, contributing the majority of the app’s usage duration ($60\% + 35\% = 95\%$). Hence, modeling the full life-cycle behaviors may enhance user experience and boost the platform’s commercial gains.

To overcome this issue, we propose TWIN-V2, an enhancement of TWIN, enabling it to model the full life cycle of user behavior. TWIN-V2 employs a *divide-and-conquer* approach, decomposing the full life-cycle history into different clusters and using these clusters to model the user’s long-term interests. Specifically, we divide the model into two parts: offline and online. In the offline part, we employ hierarchical clustering to aggregate similar items in life-cycle behaviors into clusters and extract features from each cluster to form a virtual item. In the online part, we utilize cluster-aware target attention to extract long-term interests based on clustered behaviors. Here, the attention scores are reweighted based on the corresponding cluster size. Extensive experiments reveal that TWIN-V2 enhances performance across various types of users, leading to more accurate and diverse recommendation results.

The main contributions are summarized as follows:

- We propose TWIN-V2 to capture user interests from extremely long user behavior, which successfully extends the maximum length for user modeling across the life cycle.
- The hierarchical clustering method and cluster-aware target attention improve the efficiency of long sequence modeling and model more accurate and diverse user interests.
- Extensive offline and online experiments have validated the effectiveness of TWIN-V2 in scaling ultra-long user behavior sequences.
- We share hands-on practices for deploying TWIN-V2, including offline processing and online serving, which serve the main traffic of around 400 million active users daily at Kuaishou.

2 RELATED WORK

Click-Through Rate Prediction. Click-Through Rate (CTR) prediction is vital for modern Internet businesses. Pioneering studies leveraged shallow models, like LR [18], FM [17], and FFM [9], to model feature interactions. Wide&Deep [5], DeepFM [8], and DCN [20, 21] successfully combine deep and shallow models. Furthermore, several studies have explored more complex neural networks, such as AutoInt [19] leveraging the multi-head self-attention mechanism. Users’ behavioral data have gained much attention recently. YoutubeDNN [6] has employed average pooling across the entire sequence of past behaviors. Furthermore, DIN [24] leveraged a target attention mechanism to adaptively capture user interests from historical behaviors concerning the target item. This target attention approach has been widely used by practitioners [1–3, 23, 24]. Following this line, DIEN [23] and DSIN [7] incorporated temporal information into the target attention. BST [4] and TWIN [2] leveraged the multi-head attention for the target attention.

Long-Term User Behavior Modeling. Modeling lifelong historical behaviors is crucial in CTR prediction. Early efforts involved memory networks to capture and remember user interests, such as HPMN [16] and MIMN [12]. Several approaches attempt to directly shorten the length of user behaviors, such as UBCS [22] which samples sub-sequences from the entire history and clusters all candidate items to accelerate the sampling process, and DGIN [10] employs item IDs to deduplicate history, thereby compressing redundant items. Given the inconsistency between long-term and short-term interests, recent works commonly consider them separately. SIM [13] and UBR4CTR [14, 15] employ a two-stage approach, first retrieving a subset of behaviors related to the target

¹<https://www.kuaishou.com/en>

Table 1: Comparison with SOTA methods. The lower part lists the two-stage models. ‘Length’ means the maximum sequence length of user behaviors in original papers.

Method	Length	GSU Strategy
DIN [24]	$\sim 10^3$	N/A
DIEN [23]	$\sim 10^2$	N/A
MIMN [12]	$\sim 10^3$	N/A
DGIN [10]	$\sim 10^4$	N/A
UBR4CTR [14, 15]	$\sim 10^2$	BM25
SIM Hard [13]	$\sim 10^3$	Category Filtering
SIM Soft [13]	$\sim 10^3$	Inner Product
ETA [3]	$\sim 10^3$	LSH & Hamming
SDIM [1]	$\sim 10^3$	Hash Collision
TWIN [2]	$\sim 10^5$	Efficient Target Attention (TA)
TWIN-V2 (ours)	$\gg 10^5$	Hierarchical Clustering & TA

TWIN-V2: It can handle behavioral histories far exceeding 10^5 by adjusting the clustering hyper-parameters. The settings in this paper can process behaviors up to 10^6 , sufficiently covering the entire life cycle of users at Kuaishou.

item from long-term history, and then modeling long-term interests with retrieved behaviors and short-term interests with latest history. Following them, ETA [3] explored the locality-sensitive hashing method and the Hamming distance to retrieve relevant behaviors. SDIM [1] directly gathers behavior items that share the same hash signature with the candidate item. TWIN [2] improves the retrieval accuracy by leveraging an identical target-behavior relevance metric for both stages. TWIN [2] also extends the maximum length of lifelong behaviors around 10^3 to $10^4 - 10^5$ by accelerating the target attention mechanism.

This paper aims to extend long-term history modeling to the life-cycle level while maintaining efficiency. Table 1 summarizes the comparison between TWIN-V2 and previous works.

3 METHOD

This section elaborates on the proposed model, detailing the entire process from the overall workflow to the deployment framework. Notations are summarized in Appendix A.1.

3.1 Preliminaries

The core task of CTR prediction is to forecast the probability of a user clicking on an item. Let $\mathbf{x}_k \in \mathbb{R}^{d_0}$ denote the feature representation of k -th data sample and let $y_k \in \{0, 1\}$ denote the label of the k -th interaction. The process of CTR prediction can be written as:

$$\hat{y}_k = \sigma(f(\mathbf{x}_k)), \quad (1)$$

where $\sigma(\cdot)$ is the sigmoid function, f is the mapping function implemented as the CTR model $f: \mathbb{R}^{d_0} \rightarrow \mathbb{R}$, and \hat{y}_k is the predicted probability. The model f is trained by the binary cross entropy loss:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{k=1}^{|\mathcal{D}|} y_k \log(\hat{y}_k) + (1 - y_k) \log(1 - \hat{y}_k), \quad (2)$$

where \mathcal{D} denotes the training dataset.

3.2 Overall Workflow

Figure 2 illustrates the overall framework of TWIN-V2, including *offline* and *online* parts. This paper focuses on life-cycle behavior modeling. Thus we emphasize this part and omit other parts.

Algorithm 1: Hierarchical Clustering Over Life Cycle

Input: History items $S_{1:T}$, Item embeddings \mathbf{K}_s , Item Playing Completion Ratio $P_{1:T}$, group number M , and maximum cluster size γ , where s_j is the j -th item, $\mathbf{K}_s \in \mathbb{R}^{T \times d}$, $p_i \in \mathbb{R}$, and $M, \gamma \in \mathbb{N}^+$

Output: Clustered items $C_{1:T}$

```

1 Function  $h_{\text{comp}}(S_{1:T})$  :
2   # Rule-based item grouping in lines 3-8
3   Initialize  $M$  empty lists  $L_1, L_2, \dots, L_M$ ;
4   Define a function  $\text{getGroup}(p)$  that returns the group number  $m$  for the  $j$ -th
      item considering its  $p_j$ ;
5   for  $j = 1$  to  $T$  do
6      $m = \text{getGroup}(p_j)$ ;
7      $L_m.$ Append( $s_j$ );
8   end
9   # Recursively adaptive clustering in lines 10-28
10  Initialize an empty list  $C$ ;
11  for  $m = 1$  to  $M$  do
12    Initialize an empty queue  $Q$ ;
13     $Q.$ enqueue( $L_m$ );
14    while  $Q$  is not empty do
15       $V \leftarrow Q.$ dequeue();
16      if  $|V| < \gamma$  then
17         $C.$ append( $V$ );
18      end
19      else
20         $\delta \leftarrow \lfloor |V|^{0.3} \rfloor$ ;
21        # Cluster items in  $V$  into  $\delta$  groups based on their
           corresponding embeddings in  $\mathbf{K}_s$ 
22         $J_{1:\delta} \leftarrow \text{Kmeans}(\delta, V, \mathbf{K}_s)$ ;
23        for  $i = 1$  to  $\delta$  do
24           $Q.$ enqueue( $J_i$ );
25        end
26      end
27    end
28  end
29  return  $C$ 

```

Since the behavior over the entire life cycle is too lengthy for online inferring, we initially compress it during the *offline* phase. User behavior often includes many similar videos, as users frequently browse their favorite topics. Therefore, we employ hierarchical clustering to aggregate items with similar interests in each user’s history into clusters. Online inferring captures the user’s long-term interests using these clusters and their representation vectors.

During *online inferring*, we adopt a two-stage approach for life-cycle modeling. First, we use GSU to retrieve the top 100 clusters related to the target item from the clustered behaviors, and then employ ESU to extract long-term interests from these clusters. In GSU and ESU, we implement cluster-aware target attention, adjusting weights for different clusters.

3.3 Life-Cycle User Modeling

Considering that the user histories over the life cycle have ultra-long sequences, we first compress these behaviors in a divide-and-conquer method and then extract users’ long-term interests from compressed behaviors.

3.3.1 Hierarchical Clustering Over Life Cycle. For a user u , we represent all her historical behaviors as a sequence of items $S = [s_1, s_2, \dots, s_T]$, where T is the number of life-cycle behaviors and s_j is the j -th interacted item. Users may watch many similar videos in S . For instance, a user fond of NBA games may have hundreds of

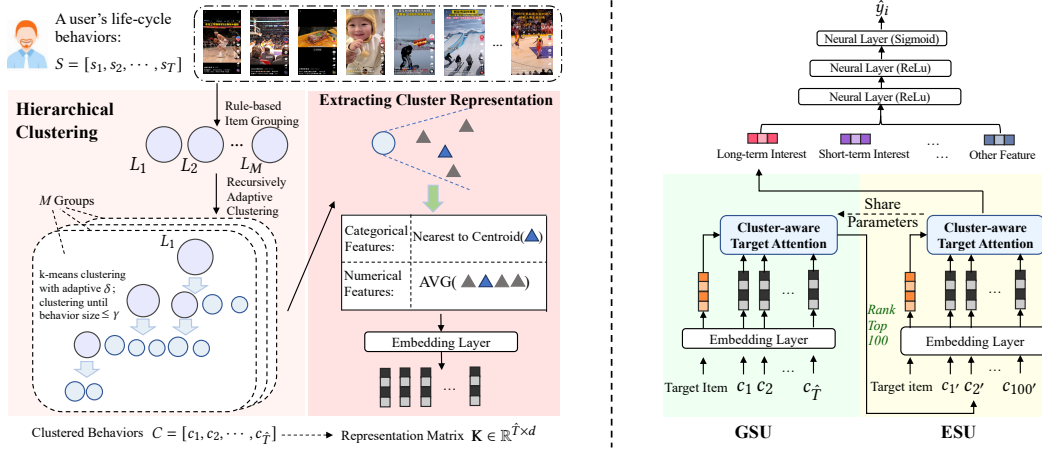


Figure 2: Overview of TWIN-V2 in Kuaishou's CTR prediction system. Left: the offline component compresses life-cycle behaviors and extracts features of clustered behaviors. Right: the online component captures user interests from life-cycle behaviors, which adopts a two-stage approach. GSU and ESU for both stages take clustered behaviors as input.

basketball-related videos in his history. Therefore, a natural idea is to aggregate similar items in S into clusters, using a single cluster to represent many similar items, thereby reducing the size of T .

Formally, we aim to use a compression function h_{comp} to reduce a behavior sequence S of length T to a sequence C of length \hat{T} :

$$C = h_{\text{comp}}(S), \quad (3)$$

where $\hat{T} \ll T$. Let C denote the clustered behaviors $C = [c_1, c_2, \dots, c_{\hat{T}}]$, where c_i denotes the i -th set containing clustered items.

Specifically, we employ a hierarchical clustering approach to implement h_{comp} , as illustrated in Algorithm 1. In the first step, we simply split the historical behaviors into groups. We first categorize historical items into M different groups based on the proportion of time they were played by the user u , which is denoted as $p_j = \frac{\text{playing time}}{\text{video duration}}$ for the j -th behavior item s_j . In the second step, we recursively cluster the life-cycle historical behaviors of each group until the number of items in each cluster does not exceed γ . We employ the widely used k-means method with adaptive cluster size γ to cluster the embeddings K_s of behaviors S obtained from the recommendation model. γ is a hyper-parameter to control the maximum cluster size.

Next, we will elaborate on the rationales behind Algorithm 1:

- **Item Grouping:** In short video scenarios, the playing completion ratio of a video can be seen as an indicator of the user's level of interest in the video. We initially group behaviors by playing completion ratio to ensure the final clustering results have a relatively consistent playing time ratio. If this is not done, it would result in an imbalanced distribution within each cluster because k-means only considers item embeddings. The function $\text{getGroup}(p_j)$ is specific to the application scenarios, for example, dividing the proportional range into five equal parts. M is a constant, set to 5 in our practices.
- **Recursively Adaptive Clustering:** We opt for hierarchical clustering instead of single-step clustering because user history is personalized, making it impractical to set a universal number of clustering iterations. We also dynamically set the number of clusters (δ equals the 0.3 power of the number of items), allowing clustering to adapt to varying scales of behaviors. The clustering process

stops when the item size falls below γ , where γ is set to 20. In our practice, the average cluster size is approximately $\frac{T}{2}$, hence \hat{T} is about $\frac{T}{10}$, significantly reducing the length of life-cycle behaviors. Item embeddings K_s used for clustering are obtained from the recommendation model, which means the clustering is guided by collaborative filtering signals.

Thus, we compress the originally lengthy user behavior sequence $S = [s_1, s_2, \dots, s_T]$ into a shorter sequence $C = [c_1, c_2, \dots, c_{\hat{T}}]$ by aggregating similar items into clusters.

3.3.2 Extracting Cluster Representation. After obtaining clusters C , we need to extract features from items of each cluster. To minimize computational and storage overhead, we use a virtual item to represent the features of each cluster.

We divide item features into two categories (numerical and categorical) and extract them using different methods. Numerical features are typically represented using scalars, for example, video duration and user playing time for a video. Categorical features are commonly represented using one-hot (or multi-hot) vectors, such as video ID and author ID. Formally, given an arbitrary item v , its feature can be written as:

$$\mathbf{x}_v = [\underbrace{\mathbf{x}_{c,1}^{(v)}; \mathbf{x}_{c,2}^{(v)}; \dots; \mathbf{x}_{c,N_1}^{(v)}}_{\text{categorical (one-hot)}}, \underbrace{\mathbf{x}_{s,1}^{(v)}; \mathbf{x}_{s,2}^{(v)}; \dots; \mathbf{x}_{s,N_2}^{(v)}}_{\text{numerical (scalar)}}, \quad (4)$$

For simplicity, we use $\mathbf{x}_{1:N_1}^{(v)}$ to denote categorical features and $\mathbf{x}_{1:N_2}^{(v)}$ to denote numerical features. For a cluster c_i in C , we calculate the average of all numerical features of the items it contains as c_i 's numerical feature representation:

$$\mathbf{c}_{1:N_2}^{(i)} = \frac{1}{|c_i|} \sum_{v \in c_i} \mathbf{x}_{1:N_2}^{(v)}, \quad (5)$$

For categorical features, since their average is meaningless, we adopt the closest item to the centroid in cluster c_i to represent

cluster c_i :

$$\mathbf{c}_{1:N_1}^{(i)} = \mathbf{x}_{1:N_1}^{(v)} \quad (6)$$

$$v = \arg \min_{v \in c_i} \|\mathbf{k}_v - \mathbf{k}_{\text{centroid}}\|_2^2, \quad (7)$$

where $\mathbf{k}_v, \mathbf{k}_{\text{centroid}} \in \mathbb{R}^d$ are embeddings of the item v and centroid in c_i , respectively. These embeddings can be looked up from \mathbf{K}_s . Thus, the entire cluster c_i is represented by an aggregated, virtual item feature $\mathbf{c}_i = [\mathbf{c}_{1:N_1}^{(i)}; \mathbf{c}_{1:N_2}^{(i)}]$. After passing through embedding layers, each cluster's feature is embedded into a vector, e.g., $\mathbf{k}_i \in \mathbb{R}^d$ for c_i . The GSU and ESU modules estimate the relevance between each cluster and the target item based on these embedded vectors.

3.3.3 Cluster-aware Target Attention. Following TWIN [2], we employ an identical efficient attention mechanism in both ESU and GSU, which take representations of clusters C as input.

Given the clustered behaviors $C = [c_1, c_2, \dots, c_{\hat{T}}]$, we obtain a matrix $\mathbf{K} \in \mathbb{R}^{\hat{T} \times d}$ composed of representation vectors through the embedding layer, where $\mathbf{k}_i \in \mathbb{R}^d$ is the vector for i -th cluster's feature \mathbf{c}_i . Then we use a target attention mechanism to measure the relevance between the target item and historical behaviors. We initially apply the "Behavior Feature Splits and Linear Projection" technique from TWIN [2] to enhance the efficiency of target attention, details in Appendix A.2. This method splits item embeddings into inherent and cross parts. $\mathbf{q} \in \mathbb{R}^H$ denotes the vector of the target item's inherent features. $\mathbf{K}_h \in \mathbb{R}^{\hat{T} \times H}, \mathbf{K}_c \in \mathbb{R}^{\hat{T} \times C}$ are inherent and cross embeddings of clustered behaviors respectively, where $\mathbf{K} = [\mathbf{K}_h, \mathbf{K}_c]$. We can calculate the relevance scores $\alpha \in \mathbb{R}^{\hat{T}}$ between the target item and clustered behaviors as follows:

$$\alpha = \frac{(\mathbf{K}_h \mathbf{W}^h)(\mathbf{q}^\top \mathbf{W}^q)^\top}{\sqrt{d_k}} + (\mathbf{K}_c \mathbf{W}^c) \beta, \quad (8)$$

where $\mathbf{W}^h, \mathbf{W}^q, \mathbf{W}^c$ are linear projections, d_k is the projected dimension, and β is a learnable cross feature weight. The α inadequately represents the relationship between clusters and the target item, due to varying item counts in clusters. Assuming two clusters have the same relevance, the cluster with more items should be considered more important, as more items imply a stronger user preference. Thus, we adjust the relevance scores according to the cluster size:

$$\alpha' = \alpha + \ln \mathbf{n}, \quad (9)$$

where $\mathbf{n} \in \mathbb{N}^{\hat{T}}$ denote the size of all clusters in C . Each element n_i means the cluster size of c_i .

During the GSU stage, we use α' to select the top 100 clusters with the highest relevance scores from the clustered behaviors of length \hat{T} . Then, these 100 clusters are input into the ESU, where they are aggregated based on relevance scores, resulting in a representation of the user's long-term interests:

$$\text{Attention}(\mathbf{q}^\top \mathbf{W}^q, \mathbf{K}_h \mathbf{W}^h, \mathbf{K}_c \mathbf{W}^c, \mathbf{K} \mathbf{W}^v) = \text{Softmax}(\alpha')^\top \mathbf{K} \mathbf{W}^v, \quad (10)$$

where \mathbf{W}^v is a projection matrix and the notations in this equation are slightly abused for simplicity by setting $\hat{T} = 100$ for the ESU

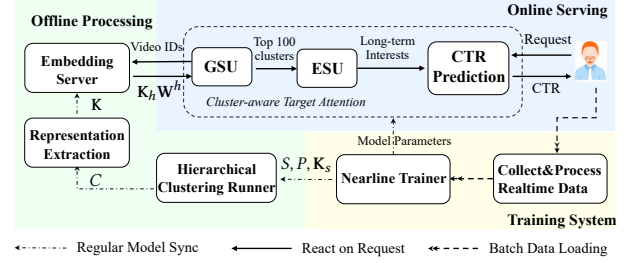


Figure 3: The deployment of TWIN-V2 at Kuaishou.

stage. Please note that the $\text{Softmax}(\alpha')$ can be written as:

$$\text{Softmax}(\alpha') = \sum_{i=1}^{\hat{T}} \frac{n_i \cdot \exp \alpha_i}{\sum_{j=1}^{\hat{T}} n_j \cdot \exp \alpha_j},$$

where the relevance score of each cluster α_i is reweighted by cluster size n_i . We employ multi-head attention with 4 heads to obtain the ultimate long-term interest:

$$\text{Long-term Interests} = \text{Concat}(\text{head}_1, \dots, \text{head}_4) \mathbf{W}^o,$$

$$\text{head}_a = \text{Attention}(\mathbf{q}^\top \mathbf{W}_a^q, \mathbf{K}_h \mathbf{W}_a^h, \mathbf{K}_c \mathbf{W}_a^c, \mathbf{K} \mathbf{W}_a^v), a \in \{1, \dots, 4\}, \quad (11)$$

where \mathbf{W}^o is a projection.

Remark. Utilizing behaviors derived from hierarchical clustering in GSU enables the model to accommodate longer historical behaviors, achieving life-cycle level modeling in our systems. Compared to TWIN, GSU's input of life-cycle behaviors encompasses a broader range of historical interests, thereby offering a more comprehensive and accurate modeling of user interests. Additionally, while existing works use the top 100 retrieved behaviors as input for ESU, TWIN-V2 utilizes 100 clusters. These clusters cover more behaviors beyond 100, allowing ESU to model a broader spectrum of behaviors. All these advantages lead to more accurate and diverse recommendation results, verified in Section 4.4.

3.4 Deployment of TWIN-V2

We divide our hands-on practices for deploying TWIN-V2 into two parts: online and offline, as shown in Figure 3.

Overview. In Figure 3, we illustrate with a single user example. When the system receives a user's request, it first extracts behavioral features from the offline-processed user life-cycle behaviors. Then, using GSU and ESU, it models these into long-term interest inputs for the CTR model to make predictions. We employ a nearline trainer for real-time model training, which incrementally updates model parameters using real-time user interaction data within an 8-minute window. For life-cycle behaviors, we apply hierarchical clustering and feature extraction to process them. This is conducted through periodic offline processing and updates.

Offline Processing. Offline processing aims to compress the user's entire life-cycle behaviors. Considering Kuaishou's user count is on a billion scale, we periodically compress their life-cycle behaviors. The hierarchical clustering runner performs a full update every 2 weeks. We set the maximum cluster size γ in hierarchical clustering to 20, leading to an average cluster size around 10. Through cluster representation extraction, each cluster is aggregated into a virtual item. In our practices, historical behavior is

Table 2: Statistics of the large-scale industrial dataset collected 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

compressed to 10% of its original length, resulting in a 90% reduction in storage costs. We employ the inherent feature projector proposed in TWIN for the embedding server, refreshing its parameters every 15 minutes from the latest CTR model.

Online Serving. After the user sends a request to the system, the offline system sends data features to the GSU, which calculates the behavior-target relevance scores, α' , according to Eq. (8) and (9). Then, the ESU selects and aggregates the top 100 clustered behaviors as long-term interests for the CTR model, which makes a prediction. To ensure efficiency in online inference, we maintain the precomputing and caching strategies from TWIN.

TWIN-V2 was incrementally deployed with TWIN. TWIN models the behaviors in recent months, while TWIN-V2 models the behaviors throughout the entire life cycle, allowing them to complement each other.

4 EXPERIMENT

In this section, we verify the effectiveness of TWIN-V2 by conducting extensive offline and online experiments.

4.1 Experimental Setup

4.1.1 Dataset. Since TWIN-V2 is designed for user behaviors with extremely long lengths, it's necessary to use datasets with ample user history. To the best of our knowledge, there is no existing public dataset with an average history length exceeding 10^4 . Therefore, we extracted user interaction data from the Kuaishou app over five consecutive days to serve as the training and testing sets. Samples were constructed from clicking logs with click-through as labels. To capture users' life-cycle histories, we further retraced each user's past behavior, covering the maximum history length for each user at 100,000. Table 2 shows the basic statistics of this industrial dataset. The first 23 hours of each day were used as the training set, while the final hour served as the testing set.

4.1.2 Baselines. Following common practices [1, 2], given that our focus is on modeling extremely long user histories, we compare TWIN-V2 with the following SOTA baselines:

- (1) **Avg-Pooling**: A naive method leverages average pooling.
- (2) **DIN** [24]: It introduces the target attention for recommendation.
- (3) **SIM Hard** [13]: Hard-search refers to the process where GSU filters long-term history based on its category.
- (4) **SIM Soft** [13]: Soft search involves selecting the top-k items by computing the vector dot product between the target item and items in the history.
- (5) **ETA** [3]: It employs Locality Sensitive Hashing to facilitate end-to-end training.
- (6) **SDIM** [1]: The GSU aggregates long-term history by gathering behaviors that share the same hashing signature with the target item.
- (7) **SIM Cluster**: Due to the reliance on

Table 3: Overall comparison. The best and the second-best performances are denoted in bold and underlined fonts, respectively. An 0.001-level improvement is considered significant for CTR prediction, as supported by [11, 21].

Method	AUC (<i>mean ± std</i>) ↑	GAUC (<i>mean ± std</i>) ↑
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	0.7939 ± 0.00014	0.7299 ± 0.00013
TWIN	<u>0.7962 ± 0.00008</u>	<u>0.7336 ± 0.00011</u>
TWIN-V2	0.7975 ± 0.00010	0.7360 ± 0.00009
Improv.	+0.16%	+0.33%

annotated categories in SIM Hard, we improved it by the clustering of video embeddings. GSU retrieves history behaviors within the same cluster with the target item. Here we clustered all items into 1,000 groups. (8) **SIM Cluster+**: This is an advanced variant of SIM Cluster with increasing cluster number to 10,000. (9) **TWIN** [2]: It proposes an efficient target attention for GSU and ESU stages.

To ensure a fair comparison, we maintained consistency in all aspects of the models except for the long-term interest modeling.

4.1.3 Evaluation Metrics & Protocol. We split the dataset into training and testing sets based on timestamps. We utilized data from 23 consecutive hours each day for the training set and the remaining 1 hour's data for the test set. For all models, we evaluated their performance over five consecutive days and reported the average results. As for evaluation metrics, we adopted AUC (Area under the ROC curve) and GAUC (Group AUC), which are widely used. We calculated the *mean* and standard deviation (*std*) of both metrics over 5 consecutive days.

4.1.4 Implementation Details. For all models, we utilize the same item and user features in our industrial context, including ample attributes such as video ID, author ID and user ID. TWIN-V2 limits the length of individual user history to a maximum of 100,000 items for clustering life-cycle user behavior. As discussed in Section 3.4, TWIN-V2 empirically compresses historical behavior to about 10% of its original size, resulting in a maximum input length of approximately 10,000 in the GSU. For other two-stage models, the maximum length of historical behavior input into the GSU is limited to 10,000 behaviors. For DIN and Avg-Pooling, the maximum length of recent history is 100 since they are not designed for long-term behaviors. All the two-stage models leverage the GSU to retrieve the top 100 historical items and feed them into the ESU for interest modeling. The batch size is set to 8192. The learning rate is $5e-6$ for Adam.

4.2 Overall Performance

From the results in Table 3, we have the following observations:

- **TWIN-V2 significantly outperforms other baselines by a large margin.** As supported by existing literature [11, 21], an improvement of 0.001 in AUC is considered significant for CTR prediction and is sufficient to yield online benefits. TWIN-V2 achieves

Table 4: Statistical analyses for different hierarchical clustering methods. ‘Adaptive’ denotes our method. ‘Binary’ denotes the variant with $\delta = 2$. ‘Balanced&Binary’ denotes the variant with $\delta = 2$ and enforced balanced k-means.

Method	Cluster number δ	Cluster Accuracy	Running Time (sec)
Balanced&Binary	2	0.765	1.010
Binary	2	0.789	1.201
Adaptive (ours)	dynamic	0.802	0.750

Cluster accuracy is measured by the average cosine similarity of each item to the centroid within each cluster. Running time denotes the average time taken for each user to complete hierarchical clustering.

an improvement over the second-best performing TWIN model, with an increase of 0.0013 in AUC and 0.0024 in GAUC. These improvements verify the effectiveness of TWIN-V2.

•TWIN-V2 exhibited a higher relative improvement in GAUC compared to its relative improvement in AUC. GAUC provides a more fine-grained measure of models. The larger improvement in GAUC indicates that TWIN-V2 achieves enhancements across various types of users, rather than just performing better on the overall samples, and it also excels in specific subsets. Since TWIN-V2 incorporates longer behavioral sequences, we believe it achieves greater improvements among highly active user groups, a hypothesis validated in Appendix A.3.

4.3 Ablation Study

We conducted an ablation study to investigate the role of core modules in TWIN-V2 and the rationale behind our design.

4.3.1 Comparison of Different Hierarchical Clustering Methods. Our hierarchical clustering method has two key features: 1. It utilizes a dynamic clustering number δ to adapt to varying sizes of behaviors; 2. The k-means clustering is non-uniform, resulting in clusters of different sizes in the end. We first verify the effectiveness of adaptive δ , creating a variant that uses a fixed cluster number $\delta = 2$ for all cases, denoted as ‘Binary’. Furthermore, to test the effect of uniform cluster sizes, we create a variant that enforces balanced k-means clustering results when $\delta = 2$, denoted as ‘Balanced&Binary’. In this variant, after each k-means iteration, a portion of items from the larger cluster, which are closer to the centroid of the other cluster, are moved to the other cluster to ensure equal sizes of the final two clusters.

Table 4 reports the statistical analyses on TWIN-V2 and these two variants. Among the three methods, ours achieves the highest cluster accuracy. This suggests that our method creates clusters where items have more closely matched representation vectors, resulting in higher similarity among items within each cluster. Our method also achieves the shortest running time, confirming the efficiency of adaptive δ . Furthermore, we reported the performance of these methods on the test set, as depicted in the left part of Figure 4. These results validate that the adaptive method can assign more similar items within each cluster and speed up the hierarchical clustering process compared with other methods.

4.3.2 Effectiveness of Cluster-aware Target Attention. In the proposed cluster-aware target attention, as compared to TWIN, the main difference lies in using clustered behavior as input and reweighting the attention scores based on cluster size. We separately removed the cluster size reweighting part from GSU and ESU to

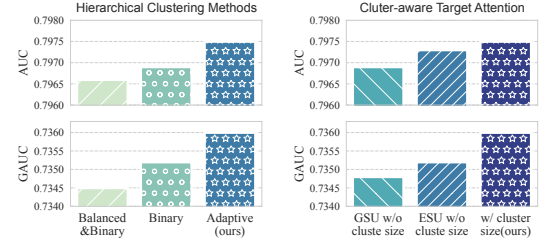


Figure 4: Effects of core modules in TWIN-V2. Left: Comparisons between different hierarchical clustering methods. Right: Impact of omitting cluster size in the reweighting of relevance scores.

Table 5: Results from online A/B test. Both watch time and the diversity of recommended results demonstrated improvements compared with TWIN.

Scenarios	Featured-Video Tab	Discovery Tab	Slide Tab
Watch Time	+0.672%	+0.800%	+0.728%
Diversity	+0.262%	+0.740%	+0.005%

0.1% increase is a significant improvement that brings great business gains in Kuaishou.

assess its impact on performance. In specific, we replaced the α' with α separately for both GSU and ESU in Eq. (9). The right part of Figure 4 shows the experimental results. Omitting reweighting leads to performance decline, validating the effectiveness of adjusting attention scores through cluster size.

4.4 Online Experiments

We conducted an online A/B test to validate the performance of TWIN-V2 in our industrial system. Table 5 illustrates the relative improvements of TWIN-V2 in terms of watch time and diversity of recommended results in three representative scenarios in Kuaishou (Featured-Video Tab, Discovery Tab, and Slide Tab). Watch Time measures the total amount of time users spend. Diversity refers to the variety in the model’s recommended outcomes, such as the richness in types of videos. From the results, it is evident that TWIN-V2 can better model user interests, leading to improved watch time. Additionally, by modeling longer historical behaviors, TWIN-V2 uncovers a more diverse range of user interests, resulting in more varied and diverse recommended results.

5 CONCLUSION

In this paper, we propose the TWIN-V2, which effectively extends the maximum length of user history to the life-cycle level, accommodating up to 10^6 behaviors in Kuaishou. The offline hierarchical clustering and feature extraction methods compress ultra-long behaviors into shorter clusters, significantly reducing the storage and computational overhead for life-cycle behaviors by 90%. The cluster-aware target attention for online inferring captures comprehensive and multi-faceted user interests, leading to more accurate and diverse recommendation results. Extensive offline and online experiments have demonstrated the effectiveness of TWIN-V2 over SOTA baselines. TWIN-V2 has been successfully deployed in Kuaishou, serving the main traffic of around 400 million active users daily.

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. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) (CIKM '22). Association for Computing Machinery, New York, NY, USA, 2974–2983. <https://doi.org/10.1145/3511808.3557082>
- [2] 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 the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (KDD '23). Association for Computing Machinery, New York, NY, USA, 3785–3794. <https://doi.org/10.1145/3580305.3599922>
- [3] Qiwei Chen, Changhua Pei, Shanshan Lv, Chao Li, Junfeng Ge, and Wenwu Ou. 2021. End-to-End User Behavior Retrieval in Click-Through Rate Prediction Model. *arXiv:cs.LR/2108.04468*
- [4] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior Sequence Transformer for E-commerce Recommendation in Alibaba. *CoRR* abs/1905.06874 (2019). [arXiv:1905.06874](https://arxiv.org/abs/1905.06874) <http://arxiv.org/abs/1905.06874>
- [5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishikesh Aradhya, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Isipir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems* (Boston, MA, USA) (DLRS 2016). Association for Computing Machinery, New York, NY, USA, 7–10. <https://doi.org/10.1145/2988450.2988454>
- [6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) (RecSys '16). Association for Computing Machinery, New York, NY, USA, 191–198. <https://doi.org/10.1145/2959100.2959190>
- [7] 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. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (Macao, China) (IJCAI'19). AAAI Press, 2301–2307.
- [8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) (IJCAI'17). AAAI Press, 1725–1731.
- [9] 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* (Boston, Massachusetts, USA) (RecSys '16). Association for Computing Machinery, New York, NY, USA, 43–50. <https://doi.org/10.1145/2959100.2959134>
- [10] Qi Liu, Xuyang Hou, Haoran Jin, Jin Chen, Zhe Wang, Defu Lian, Tan Qu, Jia Cheng, and Jun Lei. 2023. Deep Group Interest Modeling of Full Lifelong User Behaviors for CTR Prediction. *arXiv:cs.LR/2311.10764*
- [11] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) (SIGIR '18). Association for Computing Machinery, New York, NY, USA, 1137–1140. <https://doi.org/10.1145/3209978.3210104>
- [12] 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* (Anchorage, AK, USA) (KDD '19). Association for Computing Machinery, New York, NY, USA, 2671–2679. <https://doi.org/10.1145/3292500.3330666>
- [13] 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* (Virtual Event, Ireland) (CIKM '20). Association for Computing Machinery, New York, NY, USA, 2685–2692. <https://doi.org/10.1145/3340531.3412744>
- [14] Jiarui Qin, Weinan Zhang, Rong Su, Zhirong Liu, Weiwen Liu, Guangpeng Zhao, Hao Li, Ruiming Tang, Xiuqiang He, and Yong Yu. 2023. Learning to Retrieve User Behaviors for Click-through Rate Estimation. *ACM Trans. Inf. Syst.* 41, 4, Article 98 (apr 2023), 31 pages. <https://doi.org/10.1145/3579354>
- [15] 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* (Virtual Event, China) (SIGIR '20). Association for Computing Machinery, New York, NY, USA, 2347–2356. <https://doi.org/10.1145/3397271.3401440>
- [16] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, and Kun Gai. 2019. Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) (SIGIR'19). Association for Computing Machinery, New York, NY, USA, 565–574. <https://doi.org/10.1145/3331184.3331230>
- [17] Steffen Rendle. 2010. Factorization Machines. In *2010 IEEE International Conference on Data Mining*. 995–1000. <https://doi.org/10.1109/ICDM.2010.127>
- [18] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-through Rate for New Ads. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada) (WWW '07). Association for Computing Machinery, New York, NY, USA, 521–530. <https://doi.org/10.1145/1242572.1242643>
- [19] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) (CIKM '19). Association for Computing Machinery, New York, NY, USA, 1161–1170. <https://doi.org/10.1145/3357384.3357925>
- [20] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *Proceedings of the ADKDD'17* (Halifax, NS, Canada) (ADKDD'17). Association for Computing Machinery, New York, NY, USA, Article 12, 7 pages. <https://doi.org/10.1145/3124749.3124754>
- [21] 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* (Ljubljana, Slovenia) (WWW '21). Association for Computing Machinery, New York, NY, USA, 1785–1797. <https://doi.org/10.1145/3442381.3450078>
- [22] Yuren Zhang, Enhong Chen, Binbin Jin, Hao Wang, Min Hou, Wei Huang, and Runlong Yu. 2022. Clustering Based Behavior Sampling with Long Sequential Data for CTR Prediction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (<conf-loc>, <city>Madrid</city>, <country>Spain</country>, </conf-loc>) (SIGIR '22). Association for Computing Machinery, New York, NY, USA, 2195–2200. <https://doi.org/10.1145/3477495.3531829>
- [23] 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. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 5941–5948. <https://doi.org/10.1609/aaai.v33i01.33015941>
- [24] 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* (London, United Kingdom) (KDD '18). Association for Computing Machinery, New York, NY, USA, 1059–1068. <https://doi.org/10.1145/3219819.3219823>

A APPENDIX

A.1 Summary of Notations

We summarize the important notations used in Section 3 in the following table:

Table 6: Important Notations Used in Section 3.

f	predictor	σ	sigmoid	\mathbf{x}	feature vector
\mathcal{D}	dataset	\mathbb{R}	real number set	\hat{y}	predicted CTR
\mathcal{L}	loss	d_0	feature dimension	y	ground truth label

S	historical behaviors	\mathbf{K}	cluster embeddings of C
C	clustered behaviors	P	playing completion ratio of S
s_j	j -th behavior in S	p_j	playing completion ratio of s_j
c_i	i -th cluster in C	M	group number
T	length of S	L_m	m -th item group
\hat{T}	length of C	γ	maximum cluster size
\mathbf{K}_s	item embeddings of S	δ	adaptive cluster number

\mathbf{x}_v	feature vector of item v	\mathbf{c}_i	feature vector of cluster c_i
$\mathbf{x}_{1:N_1}^{(v)}$	categorical part of \mathbf{x}_v	$\mathbf{c}_{1:N_1}^{(i)}$	categorical part of \mathbf{c}_i
$\mathbf{x}_{1:N_2}^{(v)}$	numerical part of \mathbf{x}_v	$\mathbf{c}_{1:N_2}^{(i)}$	numerical part of \mathbf{c}_i
\mathbf{k}_i	embedding vector of c_i	d	embedding dimension

\mathbf{q}	target item's inherent embedding	β	learnable cross feature weight
\mathbf{K}_h	inherent feature part of \mathbf{K}	\mathbf{K}_c	cross feature part of \mathbf{K}
H	inherent feature dimension	C	cross feature dimension
α	attention weight	α'	adjusted attention weight
n	cluster size of C	\mathbb{N}	natural number set

$\mathbf{W}^q, \mathbf{W}^h, \mathbf{W}^c, \mathbf{W}^v, \mathbf{W}^o$	linear projection parameters
--	------------------------------

A.2 Behavior Feature Splits and Linear Projection

Following TWIN [2], we define the feature representations of a length \hat{T} clustered behavior sequence $[c_1, c_2, \dots, c_{\hat{T}}]$ as matrix \mathbf{K} , where each row denotes the features of one behavior. In practice, the linear projection of \mathbf{K} in the attention score computation of MHTA is the key computational bottleneck that hinders the application of multi-head target attention (MHTA) on ultra-long user behavior sequences. We thus propose the following to reduce its complexity.

We first split the behavior features matrix \mathbf{K} into two parts,

$$\mathbf{K} = [\mathbf{K}_h, \mathbf{K}_c] \in \mathbb{R}^{\hat{T} \times (H+C)}, \quad (12)$$

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

For the inherent features \mathbf{K}_h , although the dimension H is large (64 for each id feature), the linear projection is not costly. The inherent features of a specific item are shared across users/behavior sequences. With essential caching strategies, $\mathbf{K}_h \mathbf{W}^h$ could be efficiently “calculated” by a look-up and gathering procedure.

For the user-item cross features \mathbf{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

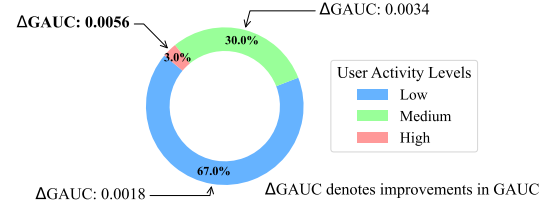


Figure 5: GAUC improvement among users of three activity levels with different proportions. The improvement in GAUC is calculated by subtracting the GAUC of TWIN from the GAUC of TWIN-V2. TWIN-V2 demonstrates more improvements for users with a longer history.

sequences. 2). Each user watches a video 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,

$$\mathbf{K}_c \mathbf{W}^c \triangleq \begin{bmatrix} \mathbf{K}_{c,1} \mathbf{w}_1^c & \dots & \mathbf{K}_{c,J} \mathbf{w}_J^c \end{bmatrix}, \quad (13)$$

where $\mathbf{K}_{c,j} \in \mathbb{R}^{\hat{T} \times 8}$ is a column-wise slice of \mathbf{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., $\mathbf{K}_c \mathbf{W}^c \in \mathbb{R}^{\hat{T} \times J}$. Note that this simplified projection is equivalent to restricting \mathbf{W}^c to a diagonal block matrix.

A.3 Analysis of User Activity Levels

We postulate that enhancing recommendation model performance can be achieved by extending the length of user history input. Given that users with different activity levels exhibit varied lengths of historical behavior, the effect of extending long-term interest modeling to the life-cycle level is likely to differ among them. Consequently, we grouped users by different history lengths and reported the performance improvement across these groups.

We categorized users in the dataset into three groups based on the number of their historical behaviors: Low, Medium, and High. Additionally, we calculated the GAUC for TWIN and TWIN-V2 models across these groups. The improvements in GAUC for TWIN-V2 in different groups and their respective proportions of the total user count are shown in Figure 5. It is observable that TWIN-V2 achieves performance improvements across all user groups, validating the effectiveness of our approach. It is also evident that the absolute increase in GAUC is greater in user groups with a higher number of historical behaviors. This occurs as users with a greater number of historical actions possess a broader spectrum of interests within their life-cycle behaviors, thereby presenting more significant opportunities for enhancement. Additionally, TWIN-V2 also achieves performance improvements in groups of users with shorter histories. This is due to the incorporation of clustered behavior features in the cluster-aware target attention mechanism. These features represent the aggregated characteristics of all items within the cluster. Consequently, the data fed into GSU and ESU reflects a more comprehensive scope of behaviors compared to TWIN, leading to improved performance.