HORAE: Temporal Multi-Interest Pre-training for Sequential Recommendation

SHIRUI HU, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China
WEICHANG WU, Ant Group, Hangzhou, China
ZUOLI TANG, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China
ZHAOXIN HUAN, LIN WANG, XIAOLU ZHANG, and JUN ZHOU, Ant Group, Hangzhou, China
LIXIN ZOU and CHENLIANG LI, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China

The data sparsity problem has been a long-standing obstacle towards achieving better recommendation performance since it is miserable to estimate the user's interests from limited historical behaviors. The pre-training paradigm, i.e., learning universal knowledge across a wide spectrum of domains, has increasingly become a new de-facto practice in many fields, especially for adaption to new domains. The merit of this superior generalizability renders it a natural choice to tackle the data sparsity problem for various recommendation scenarios. Hence, several efforts mainly follow masked language modeling or simple data augmentation via contrastive learning to build a pre-trained recommendation model. Our recent work (namely MIRACLE) suggests that the common treatment utilizing the masked language modeling is not sufficient for pre-training

This work is an extended version of the paper [27] presented at the 46th Annual International ACM SIGIR Conference (SIGIR '23) (Taiwan, China, 2023).
This work was done when Shirui Hu was an intern at Ant Group.
This work was supported by National Natural Science Foundation of China (Nos. 62272349; 62302345; and U23A20305); Natural Science Foundation of Hubei Province under Grant No. 2023BAB160; the Ministry of Education Humanities and Social Sciences Project under Grant No. 24JDSZ3073; and CAAI-Ant Group Research Fund (No. CAAI-MYJJ2024-01). The numerical calculations in this article have been done on the supercomputing system in the Supercomputing Center of Wuhan University.
Authors' Contact Information: Shirui Hu, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: hushirui@whu.edu.cn; WeichangWu, Ant Group, Hangzhou, China; e-mail: jiuyue.wwc@antgroup.com; Zuoli Tang, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: tangzuoli@whu.edu.cn; Zhaoxin Huan, Ant Group, Hangzhou, China; e-mail: zhaoxin.hzx@antgroup.com; Lin Wang, Ant Group, Hangzhou, China; e-mail: fred.wl@antgroup.com; Xiaolu Zhang, Ant Group, Hangzhou, China; e-mail: yueyin.zxl@antfin.com; Jun Zhou, Ant Group, Hangzhou, China; e-mail: jun.zhoujun@antfin.com; Lixin Zou (corresponding author), Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: zoulixin@whu.edu.cn; Chenliang Li (corresponding author), Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: cllee@whu.edu.cn.

ACM 1558-2868/2025/5-ART88
https://doi.org/10.1145/3727645

a recommender system, since a user's intent could be more complex than predicting the next word or item. The encouraging results demonstrate that the multi-interest modeling could significantly push the frontier of recommender system pre-training. Nevertheless, how to accommodate the temporal dynamics of the user interests seems to be underexplored under both single vector representation and multi-interest schemes. In this article, we aim to incorporate sophisticated temporal information modeling with the current advance in this line. More specifically, we extend Miracle by further considering relative position information and two kinds of relative time interval information jointly when performing multi-interest learning. Then, a sequential process for interest refinement is proposed to learn the subtle nuances of how interests change and shift along the timeline, leading to a more precise representation of user interests. Our extensive experiments on multiple real-world datasets validate the effectiveness of the proposed solution, demonstrating a significant improvement over current state-of-the-art models on these benchmarks. The code is available at https://github.com/WHUIR/Horae.

CCS Concepts: • **Information systems** → **Recommender systems**;

Additional Key Words and Phrases: Pre-trained Model, Sequential Recommendation, Multi-Interest Learning, Temporal Modeling

ACM Reference format:
Shirui Hu, Weichang Wu, Zuoli Tang, Zhaoxin Huan, Lin Wang, Xiaolu Zhang, Jun Zhou, Lixin Zou, and Chenliang Li. 2025. HORAE: Temporal Multi-Interest Pre-training for Sequential Recommendation. *ACM Trans. Inf. Syst.* 43, 4, Article 88 (May 2025), 29 pages.
https://doi.org/10.1145/3727645

1 Introduction

Sequential recommendation, which predicts user's next interaction based on their historical behavior sequences, plays a pivotal role in multiple areas such as internet e-commerce, streaming platforms, and online advertising. In the past decades, the fundamental cornerstone underlying different methodologies is that the user's historical behaviors would reflect her interests. Hence, the data sparsity problem is a major challenge that hinders effective recommendation in different scenarios. Recently, the great success of the pre-training followed by fine-tuning in natural language processing [5] encourages the endeavor of leveraging the pre-training for sequential recommendation. Some pre-training frameworks have been proposed for better initializing the parameters in single-domain [25, 31] or transferring knowledge across different domains [8, 20]. Particularly, the cross-domain methods bridge different domains with the item's text information (e.g., product description, product name) for the universal item representation, which therefore leverages the knowledge of different domains and the semantic knowledge stored in the **pre-trained language model (PLM)** [5]. As expected, these methods achieve state-of-the-art performance on the sequential recommendation. However, these methods represent each user as a fixed vector and fail to capture the user's diverse interests. Therefore, it is noteworthy to consider the user's multiple interests when pre-training a recommender system.

As illustrated in Figure 1, the historical behavior sequences of user a and user b suggest that various interests are associated with them respectively. Specifically, user a has more focused interests in "cosmetics," "clothing," and "electronics," while user b shows fluctuating interests in "electronics," "sports," and "games." To address these issues, advanced methods such as MIND [12] and ComiRec [2] have shown that the multi-interest modeling can better capture the user's diverse interests.

In our recent work, a multi-interest pre-training framework with sparse capsule network (namely Miracle) is proposed to achieve recommender system pre-training under multi-interest scheme [27]. Specifically, for multi-interest modeling, Miracle utilizes a text-aware encoder for generating

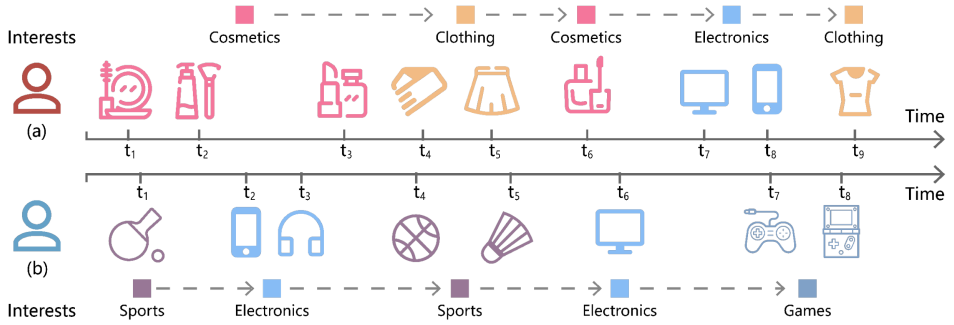

Fig. 1. The example illustrates the multi-interest nature that is dynamic, contextual, and shift along the timeline.

transferable item embeddings. Following earlier effort [8], this text-aware item encoder integrates cross-domain semantic information with a **mixture-of-experts (MoE)** adapter and a deeply contextual-aware bi-directional Transformer for item embedding generation. The sparse capsule network then derives the number of interests per user with a position-aware dynamic routing. Then, a new interest-level contrastive learning task is utilized in the pre-training stage to enhance universal multi-interest learning by establishing the connections between different interests across the domains.

Here, we argue that both single vector representation and multi-interest learning framework underexplore the temporal information of the user behaviors in a more fine-grained manner. As for the earlier pre-training models that generate a fixed vector representation with a Transformer architecture, the positional information in the form of behavior order is utilized with the positional embeddings. The same thing happens for MIRACLE where the position embeddings are used for both the text-aware encoder and the dynamic routing built inside the sparse capsule network respectively. Nevertheless, this type of positional encoding is absolute and coarse, as the temporal intervals between identical positions are not uniform. The fine-grained temporal information, including the time gap between consecutive historical items or towards the target, as well as their relative order, is predominantly overlooked. As illustrated in Figure 1, the short time interval between two electronic items at timestamp t_7 and t_8 of user (a), and their relative positions suggest that the corresponding interest is likely to be a temporary yet specific intent. Also, the historical items before timestamp t_7 of user (b) illustrate that this user is young and prefers amusement, which are validated by the interactions at the coming timestamp t_7 and t_8. We believe these signals are generalizable and could enable better universal preference transfer.

As previously mentioned, user interests evolve dynamically over time, and accurately capturing the subtle temporal shifts in user interests is crucial for improving multi-interest learning. To this end, in this article, we propose a temporal multi-interest pre-training framework on the basis of MIRACLE [27], named HORAE.[1] Building upon MIRACLE, this model refines the temporal dynamics of user interests. Specifically, we explicitly model three kinds of fine-grained temporal information: (1) the relative position order, (2) the time interval between two consecutive historical items, and (3) the time interval between each historical item and the next prediction timestamp. These richer temporal features complement each other collaboratively contributing to multi-interest learning, helping the model better understand the temporal structure and transition patterns of user interests. Moreover, to fully leverage fine-grained temporal information, and to deeply capture the evolving process of user interests, we introduce a sequential process for interest refinement. This process

[1]Named after the god of time in Greek mythology.

enables HORAE to capture the subtle changes in user interests over time, precisely modeling the temporal dependencies of each activated interest. Furthermore, to effectively guide the learning process of dynamic interest representations and improve the quality of interest modeling, we propose two interest-level contrastive learning objectives on the basis of MIRACLE: Anchor-based Contrastive Learning and Interest-Intra Learning. The Anchor-based Contrastive Learning objective aims to enhance the modeling of dynamic interest evolution, while the Interest-Intra Learning objective restricts fine-grained discrepancy in user preferences within the same activated interest, thereby improving the accuracy of interest modeling. Through these two methods, we can more precisely capture the temporal evolution of each interest, further improving performance and personalization. In summary, our contributions made in this article are as follows:

- We highlight the importance of temporal information modeling in a fine-grained manner, a method that is also significantly beneficial when building a pre-training recommender system for sequential recommendation. A new pre-training framework HORAE is proposed to model various kinds of temporal information from the user behavior sequences.
- We propose a new multi-interest pre-training model by incorporating the three kinds of temporal information when performing the multi-interest learning. Besides, a sequential process of interest refinement is also introduced to capture subtle correlations inside each interest. This is the first attempt to explore the transferability of fine-grained temporal information for pre-training driven sequential recommendation. Furthermore, two novel interest-level unsupervised tasks are proposed to enhance universal multi-interest learning.
- We conduct comprehensive experiments on several real-world datasets to verify the effectiveness of our method over nine SOTA baselines. Further experiments and analysis also demonstrate the positive impact of our design choices.

2 Related Work

This section provides a brief overview of representative efforts relevant to our work.

2.1 Sequential Recommendation

Sequential recommendation is a critical task aimed at predicting the next item a user might interact with based on their historical behavior data. In recent years, the field has witnessed the emergence of various models and methods to enhance recommendation performance. In traditional approaches, some models rely on the Markov chain assumption to estimate transition probability matrices between items for recommendation [21]. These methods mainly focus on statistical co-occurrence patterns between items but often overlook the temporal aspects of user behaviors. With the rise of deep learning, more powerful models have been introduced to better capture long-term and short-term dependencies in user behavior data. For instance, models based on Recurrent Neural Networks, like GRU4Rec [7], leverage recurrent structures to model user sequential behaviors. This enables them to better capture the transition patterns of user interests. On the other hand, recent introductions of Transformer architectures, such as SASRec [9] and BERT4Rec [25], have become the mainstream for sequential recommendation. These models employ the self-attention mechanism of Transformers to better capture high-order dependencies between items, leading to further improvement. Moreover, TiSASRec [13] and TiCoSeRec [4] strive to integrate temporal information into their modeling, further enhancing the capabilities of existing models. Specifically, traditional sequential recommendation models often overlook the time intervals between interactions, which can carry significant information about user preferences and habits. TiSASRec addresses this by incorporating the time intervals as an additional context in the self-attention mechanism, allowing the model to weigh more recent interactions more heavily and adjust the influence of

past interactions based on their temporal distance. This approach has been shown to improve the accuracy and relevance of recommendations by better capturing the temporal dependencies in user behavior sequences. TiCoSeRec introduces five data augmentation strategies to convert non-uniform sequences into uniform ones, taking into account the variance of time intervals. It applies contrastive learning to ensure that the enhanced sequences maintain a high level of similarity with the original sequences. The authors implement their method on a state-of-the-art recommendation model, CoSeRec [14], and propose TiCoSeRec. Furthermore, similarly, graph-based models, including FGNN [18], GAG [19], and PosRec [17], improve recommendation performance by modeling the global correlations between items through the user–item interactions. These graph-based models better understand dependencies between items, providing more accurate personalized recommendations.

However, while these models have made significant progress in improving recommendation performance, they typically rely predominantly on item IDs and often give less consideration to aggregation semantic correlations across domains. This limitation hampers knowledge sharing and transfer learning across different domains. Furthermore, it's important to note that user interests are dynamic and diverse over time. Therefore, using a single fixed representation is insufficient for user understanding.

2.2 Multi-Interest Recommendation

In recent years, recommendation systems have increasingly performed multi-interest learning to better understand users' historical behaviors precisely and comprehensively. For example, MIND [12] utilizes the dynamic routing [22] mechanism to extract users' multi-level interests such that more than one vector is derived to cover a user's diverse interests. ComiRec [2] refines the dynamic routing structure and introduces a novel multi-interest learning method based on self-attention. Similarly, SINE [26] employs attention mechanisms to construct user interest prototypes and then utilizes attention to extract multiple interests between user interactions and interest prototypes. UMI [3] leverages multiple attributes present in user profiles to bolster multi-interest learning. Recently, MGNM [28] integrates user embeddings into graph structures and combines graph neural networks with dynamic routing to understand users' multi-interest at multiple levels.

However, these techniques still face certain challenges. Firstly, they require fixing the number of interests for all users, which is globally difficult to optimize. This hinders their adaptability to diverse user needs, as different users are likely to have varying numbers of interests. Secondly, these methods are confined to single-domain recommendations, failing to tackle the data sparsity problem by knowledge transfer. Furthermore, they do not adequately account for temporal patterns in user historical behaviors, which limits their capacity to model the dynamic variation of interests. Consequently, the quest for more flexible and comprehensive multi-interest recommendation methods remains a challenging endeavor.

2.3 Recommender System Pre-training

By performing data augmentation or unsupervised contrastive learning, the pre-training paradigm can effectively address the data sparsity problem in recommendation systems. For instance, U-BERT [20] introduces a review encoder to learn universal domain-specific user behavior representations. Additionally, S3-Rec [31] employs a pre-training scheme to enhance data representations and utilizes the principle of maximizing mutual information (MIM) to learn the correlations between attributes, items, sub-sequences, and sequences. Furthermore, UniSRec [8] chooses to model universal item representations using a text-based PLM and excels in cross-domain scenarios. Our recent work, Miracle [27], further incorporates the multi-interest learning with the pre-training scheme for sequential recommendation. At first, Miracle utilizes a text-aware encoder for generating

transferable item embeddings. Then, a sparse capsule network is proposed to identify the number of interests per user and generate the user-level interest representation. The position-wise dynamic routing, with trainable position embeddings and an attention mechanism, helps highlight the importance of each historical item for multi-interest learning.

Although these efforts have delivered promising downstream recommendation performance, pre-training recommender system still remains at its early stage. There are many contextual factors that are underexplored under the pre-training paradigm. One important factor is the temporal information modeling, which is mainly overlooked by the above existing solutions. In this work, the proposed HORAE can be seen as a preliminary attempt to incorporate fine-grained temporal information with the pre-training scheme.

3 Methodology

In this section, we present the proposed temporal multi-interest pre-training framework in detail. We first present the problem definition. Then, as illustrated in Figure 2, we describe the three main components, with their structural details shown in the first row of the figure. Specifically, the raw sequence is first integrated into contextualized representation $[\mathbf{h}_1, \ldots, \mathbf{h}_5]$ through the *Text-Aware Temporal Item Embedding*, which integrates three kinds of temporal information and utilizes the **rotary position encoding (RoPE)** technique. Subsequently, two pre-refined interest representations, $\mathbf{u}_1$ and $\mathbf{u}_2$ are calculated via the *Temporal Sparse Interest Capsule Network*, which sequentially process the item embedding obtained in the previous step, activating, aggregating, and extracting interest capsules, resulting in pre-refined interest representations. Then, these two embedding are further refined in the *Sequential Interest Refinement*, resulting in the post-refined interest embedding $\overline{\mathbf{u}}_1$ and $\overline{\mathbf{u}}_2$. In the pre-training stage, the objective $\mathcal{L}_{pre-train}$ combines four distinct types of interest-level self-supervised learning tasks (three in the second row and one in the third row of Figure 2) along with the main recommendation task (presented in the third row of the same figure). During fine-tuning, the objective $\mathcal{L}_{fine-tune}$ incorporates one self-supervised task and one fine-tuning main task, as illustrated in the third row of the same figure.

3.1 Problem Definition

Given a user set U and an item set V, for each user $u \in U$, we have their historical interaction sequence $s^u = (v_1^u, v_2^u, \ldots, v_N^u)$ organized chronologically and the corresponding timestamp sequence $s_t^u = (t_1^u, t_2^u, \ldots, t_N^u)$, where N is the maximum sequence length, $v_i^u \in V$ and $t_i^u \in \mathbb{R}$ represent the ith item interacted with by user u and its corresponding timestamp, respectively. Additionally, for each item $v \in V$, we have its textual description, denoted as $C_v = (c_1, c_2, \ldots, c_{|C_v|})$. The goal of the sequential recommendation is to predict the next item v_{N+1}^u that user u will interact with based on s^u and s_t^u.

Due to the data sparsity problem, the available historical interactions could be insufficient, which is a common phenomenon in many real-world scenarios. Hence, there are two stages for pre-training a recommender system: pre-training and fine-tuning. Specifically, in the pre-training stage, the model is warmed up on the pre-training objective $\mathcal{L}_{pre-train}$ on the pre-training dataset D_p, which is typically a large-scale, semantically rich, multi-domain dataset. In the fine-tuning stage, the model fine-tunes its parameters to validate performance on the downstream dataset D_f with fine-tuning objective $\mathcal{L}_{fine-tune}$, which is typically smaller in scale. These two datasets, D_p and D_f, often exhibit some hidden correlations as well as semantic gap, making it a key challenge to utilize the knowledge from D_p and transfer it to enhance the model's recommendation performance on D_f effectively. The key variables of this section are defined in Table 1.

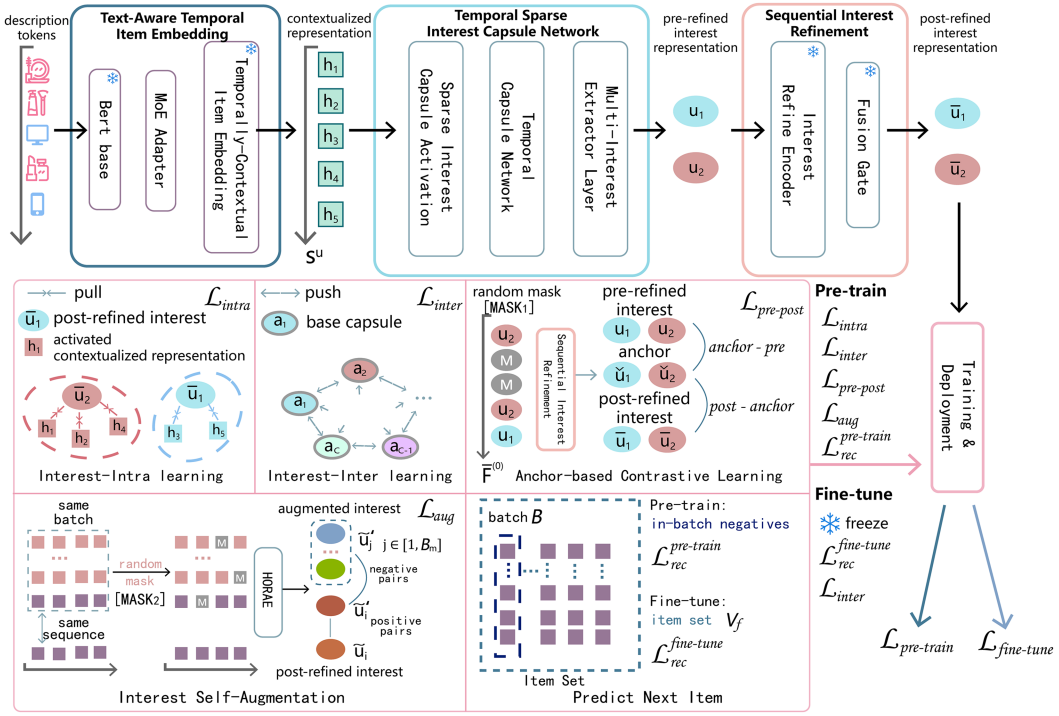

Fig. 2. The overview of the proposed HORAE. The model consists of three main components as shown in the first row of the figure. The raw sequence, consisting of five historical interaction items, is sequentially processed through each component of the framework: first by Text-Aware Temporal Item Embedding for temporal encoding, then by Temporal Sparse Interest Capsule Network for extracting temporal multi-interests, and finally by Sequential Interest Refinement to refine the interest representations. This process results in the post-refined interest representations $\bar{\mathbf{u}}_1$ and $\bar{\mathbf{u}}_2$. During the pre-training phase, the model uses in-batch negative sampling method, where each item in the batch treats itself as the positive sample and all other items in the batch as negative samples. During the fine-tuning phase, the model considers the fine-tuning item set V_f as the candidate set for the next-item prediction task. Besides, four interest-level unsupervised tasks are designed to enhance universal multi-interest learning: Interest-Intra Learning, Interest-Inter Learning, Anchor-based Contrastive Learning, and Interest Self-Augmentation.

Table 1. Key Variable Definition for Problem Definition

Variable	Definition
U	user set with each user $u \in U$.
V	item set with each item $v \in V$.
N	maximum sequence length.
s^u	historical interaction sequence for user u: $s^u = (v_1^u, v_2^u, \ldots, v_N^u)$.
s_t^u	timestamp sequence for user u: $s_t^u = (t_1^u, t_2^u, \ldots, t_N^u)$.
C_v	textual description of item v.
D_p	dataset for the pre-training stage.
D_f	dataset for the fine-tuning stage.
$\mathcal{L}_{pre-train}$	learning objective for the pre-training stage.
$\mathcal{L}_{fine-tune}$	learning objective for the fine-tuning stage.

Table 2. Key Variable Definition for Text-Aware Temporal Item Embedding

Variable	Definition
d_b	base dimension size of pre-trained text encoder $\text{BERT}_{\text{base}}$.
$\mathbf{c}_i$	textual embedding for item v_i.
d_n	hidden dimension size.
$\mathbf{x}_i$	MoE-enhanced textual representation of item v_i.
$\mathbf{g}$	gating weight vector: $\mathbf{g} = [g_1, \ldots, g_G]$.
$\mathbf{P}$	position embedding matrix: $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$.
d_i^a	adjacent temporal intervals for the ith interaction.
d_i^t	target temporal intervals for the ith interaction.
N_a	number of temporal bins for adjacent temporal intervals.
N_t	number of temporal bins for target temporal intervals.
τ	normalization scaling coefficient.
$\mathbf{D}$	time embedding matrix: $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_{N_a+N_t}]$.
$\mathbf{t}_i$	temporal interval embedding for the ith interaction.
$\mathbf{x}'_i$	temporal-textual embedding for the ith interaction.
$\mathbf{q}/\mathbf{k}$	query/key vectors in attention mechanism.
$\mathbf{R}_{\Theta}^{d_n}$	rotary positional projection matrix for d_n-dimensional space.
$\tilde{\mathbf{q}}/\tilde{\mathbf{k}}$	rotation-enhanced query/key vectors.
$\mathbf{H}$	contextualized representation: $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$.

3.2 Text-Aware Temporal Item Embedding

Following the existing works for pre-training recommender system [8, 27], we choose to encode the textual descriptions of items into a unified semantic space. Additionally, by utilizing *Temporally Contextual Item Embedding*, we generate item embeddings that comprehensively encode multiple kinds of fine-grained temporal information. In the following, we will delve into the details of this process; the key variables of this section are defined in Table 2.

3.2.1 Pre-trained Language Model-Based Text Encoder. We first adopt $\text{BERT}_{\text{base}}$ [5], a widely used PLM, to capture semantic features from the textual description C_{v_i} of item v_i. Specifically, $\text{BERT}_{\text{base}}$ is a 12-layer transformer, with each layer comprising 12 self-attention heads. Following common practice, we place a special token [CLS] in front of C_{v_i} and feed the resulting text sequence into $\text{BERT}_{\text{base}}$:

$$\mathbf{c}_i = \text{BERT}_{\text{base}}([[\text{CLS}], c_1, \ldots, c_{|C_{v_i}|}]), \tag{1}$$

where $\mathbf{c}_i \in \mathbb{R}^{d_b}$ represents the textual embedding of v_i, which is the latent feature output of the last transformer layer of token [CLS], and d_b is the base dimension size of pre-trained text encoder $\text{BERT}_{\text{base}}$. Note that $\text{BERT}_{\text{base}}$ is a pre-trained text encoder, and its parameters are frozen.

Since we merge multiple domain-specific datasets to form a pre-training dataset D_p, due to semantic gaps between different domains, the textual embeddings produced by $\text{BERT}_{\text{base}}$ for items in different domains are not aligned well under the same semantic space. This common issue is referred to as domain bias in existing literature [11].

Therefore, the MoE network [23] is utilized to mitigate domain bias as follows:

$$\mathbf{x}_i = \sum_{k=1}^{G} g_k \cdot \text{Expert}_k(\mathbf{c}_i), \tag{2}$$

where $\mathbf{x}_i$ denotes the MoE-enhanced textual representation of item v_i, G is the expert number, g_k represents the gating weight obtained by the kth expert based on the textual embedding $\mathbf{c}_i$.

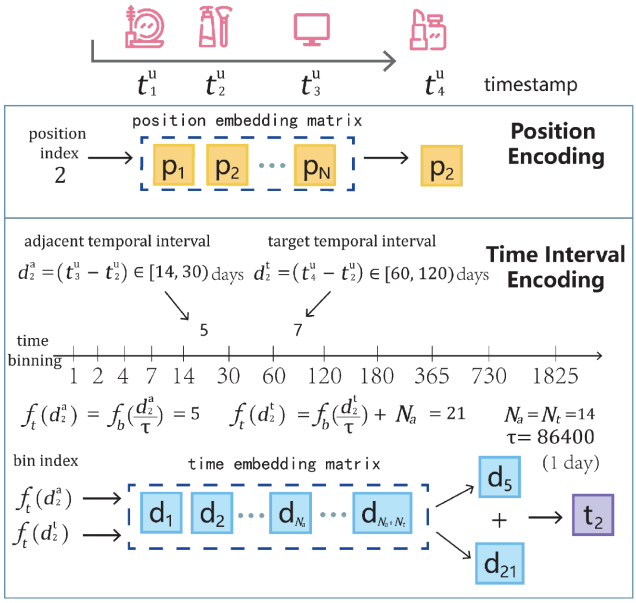

Fig. 3. The example illustrates Position Encoding and Time Interval Encoding for the second item in the sequence.

$\text{Expert}_k(\cdot)$ is a whitening transformation module, denoting the kth expert network, utilized to eliminate the bias of the textual representation in the semantic space. The calculation for the gating weight vector $\mathbf{g}$ and $\text{Expert}_k(\mathbf{c}_i)$ is as follows:

$$\mathbf{g} = \text{softmax}(\mathbf{c}_i \cdot \mathbf{W}_1 + \mathbf{b}_1), \tag{3}$$

$$\text{Expert}_k(\mathbf{c}_i) = (\mathbf{c}_i - \mathbf{b}_2^k) \cdot \mathbf{W}_2^k, \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_b \times G}, \mathbf{b}_1 \in \mathbb{R}^G$, and $\mathbf{W}_2^k \in \mathbb{R}^{d_b \times d_n}, \mathbf{b}_2^k \in \mathbb{R}^{d_b}$, d_n is the hidden size of each expert network, and $\text{softmax}(\cdot)$ refers to the softmax function.

3.2.2 Temporally Contextual Item Embedding. To better incorporate sequential patterns of a behavior sequence into a universal representation, we adopt Transformer [29] as the backbone network for modeling the long-short term dependency between items, jointly considering different kinds of fine-grained temporal information. *Position Encoding*. Specifically, considering the sequential nature of the historical behaviors, we then add a trainable position embedding matrix $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$, where $\mathbf{p}_i \in \mathbb{R}^{d_n}$ denotes the position embedding corresponding to the ith position in sequence s^u. Figure 3 illustrates a user's historical sequence. To obtain the position embedding corresponding to the second item, we need to take the position index of the item in the sequence (i.e., 2), and then use this index to retrieve the corresponding position embedding $\mathbf{p}_2$ from the position embedding matrix $\mathbf{P}$.

Time Interval Encoding. Based on the timestamp sequence $s_t^u = (t_1^u, t_2^u, \ldots, t_N^u)$ for user u, we can calculate the adjacent temporal interval d_i^a and the target temporal interval d_i^t. Specifically, $d_i^a = t_{i+1}^u - t_i^u$ represents the interval between adjacent timestamps, and $d_i^t = t_{N+1}^u - t_i^u$ signifies the interval between the target timestamp and the timestamp of the ith historical item. To simplify the notation, we use Δt to refer to both types of temporal intervals throughout the following discussions.

Next, we define a piece-wise function $f_b(\cdot)$ to discretize time intervals into specific bins, mapping time durations to corresponding bins as follows: $f_b(x)$ outputs a bin index based on the duration x in days, aligned with pre-defined thresholds [1, 2, 4, 7, 14, 30, 60, 120, 180, 365, 730, 1,825]. Here, number 7 represents a week, number 14 represents a half-month, number 30 represents a month, and number 365 represents a year, thereby reflecting periodicity. The time durations exceeding the maximum threshold are assigned to the final bin representing durations beyond 1,825 days (i.e., 5 years). It is important to note that the time span magnitudes can vary significantly across datasets from different sources. To address this issue, a preprocessing step is introduced, where a normalization scaling coefficient τ is applied for scale adjustment. After that, we personalize the time binning process by employing the time binning function $f_t(\cdot)$ as follows:

$$f_t(\Delta t) = \begin{cases} f_b\left(\frac{\Delta t}{\tau}\right), & \text{if } \Delta t = d_i^a \\ N_a + f_b\left(\frac{\Delta t}{\tau}\right), & \text{if } \Delta t = d_i^t \end{cases}. \tag{5}$$

Among these bins, N_a represents the number of bins for adjacent temporal interval, and N_t represents the number of bins for target temporal interval. A mapping from temporal interval to time embeddings via the time binning function $f_t(\cdot)$ can be easily established. Specifically, for the item at the ith position in the sequence, we calculate its corresponding adjacent time bin $f_t(d_i^a)$ and target time bin $f_t(d_i^t)$, as dictated by the time binning function. Similar to the position encoding, we define time embedding matrix $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_{N_a}, \ldots, \mathbf{d}_{N_a+N_t}]$, an embedding matrix that encompasses all time bins. Through the Embedding look-up operation, we obtain the time interval embedding $\mathbf{t}_i$ corresponding to the ith item as follows:

$$\mathbf{t}_i = \mathbf{d}_{f_t(d_i^a)} + \mathbf{d}_{f_t(d_i^t)}. \tag{6}$$

In summary, to calculate the time interval embedding for each item in the user's sequence, we first need to compute the adjacent temporal interval and the target temporal interval for each item. Based on the numerical ranges of these two values, we sequentially apply the functions $f_b(\cdot)$ and $f_t(\cdot)$ to determine the corresponding time bin index. Finally, we use the bin index to retrieve the corresponding time embedding from the time embedding matrix $\mathbf{D}$.

Specifically, to compute the time interval embedding for the second item in Figure 3, we first calculate d_2^a and d_2^t. In this example, we assume the scaling coefficient $\tau = 86400$, which corresponds to the number of seconds in a day, so that the timestamp differences are scaled to represent days. Additionally, suppose d_2^a falls within the range [14, 30) days, d_2^t falls within [60, 120) days, with $N_a = 14$. We then apply the pre-defined thresholds and the function $f_b(\cdot)$ to both intervals to determine the bin numbers. Specifically, we have $f_b(\frac{d_2^a}{\tau}) = 5$ and $f_b(\frac{d_2^t}{\tau}) = 7$. Next, applying the time binning function $f_t(\cdot)$, we calculate the corresponding time bin indices: $f_t(d_2^a) = 5$ and $f_t(d_2^t) = 21$. Finally, the time embeddings retrieved from the time embedding matrix $\mathbf{D}$ are $\mathbf{d}_5$ and $\mathbf{d}_{21}$.

Relative Position Encoding. Given contextual historical interaction sequence, we first combine the MoE-enhanced textual representation, positional embedding, and time interval embedding as follows:

$$\mathbf{x}_i' = \mathbf{x}_i + \mathbf{p}_i + \mathbf{t}_i. \tag{7}$$

Then, we model the contextual information for each item in the sequence with the L_t-layer Transformer as follows:

$$\mathbf{F}^{(0)} = [\mathbf{x}_1', \ldots, \mathbf{x}_N'] \tag{8}$$

$$\mathbf{F}^{(l)} = \text{FFN}(\text{RoMHA}(\mathbf{F}^{(l-1)})), \tag{9}$$

where FFN denotes Feed-Forward Network, and RoMHA denotes RoPE with Multi-Head Self-Attention [24]. RoPE utilizes a rotational approach to capture relative positional relationships between tokens in a sequence. The temporal-textual embedding is then transformed into query and key vectors using learnable weight matrices:

$$\mathbf{q}_m = \mathbf{W}_q \mathbf{x}'_m, \tag{10}$$

$$\mathbf{k}_n = \mathbf{W}_k \mathbf{x}'_n, \tag{11}$$

where $\mathbf{W}_q \in \mathbb{R}^{d_n \times d_n}$ and $\mathbf{W}_k \in \mathbb{R}^{d_n \times d_n}$ are the learnable parameter matrices. RoPE incorporates relative positional information via rotation matrices $\mathbf{R}^{d_n}_{\Theta,m}$. Specifically, for a d_n-dimensional embedding space, the rotation matrix is constructed by applying the following orthogonal transformation to each pair of elements along the embedding dimension:

$$\mathbf{R}^{d_n}_{\Theta,m} = \bigoplus_{j=0}^{\frac{d_n}{2}-1} \begin{pmatrix} \cos m\theta_j & -\sin m\theta_j \\ \sin m\theta_j & \cos m\theta_j \end{pmatrix} \tag{12}$$

Here, $\mathbf{R}^{d_n}_{\Theta,m}$ is the rotary positional projection matrix, which encodes the rotation operation applied to the query or the key vectors, where the parameters $\Theta = \left\{\theta_j = 10000^{-\frac{2j}{d_n}}, j \in [0, 1, \ldots, \frac{d_n}{2} - 1]\right\}$ are pre-defined to control the rotation's scale. $\bigoplus$ denotes block diagonal composition, for a d_n-dimensional vector, there are $d_n/2$ independent 2D rotation blocks. This design ensures the attention score satisfies:

$$\begin{aligned} \tilde{\mathbf{q}}_m^{\mathrm{T}} \tilde{\mathbf{k}}_n &= (\mathbf{R}^{d_n}_{\Theta,m} \mathbf{q}_m)^{\mathrm{T}} (\mathbf{R}^{d_n}_{\Theta,n} \mathbf{k}_n) \\ &= \mathbf{q}_m{}^{\mathrm{T}} \mathbf{R}^{d_n}_{\Theta,n-m} \mathbf{k}_n. \end{aligned} \tag{13}$$

The term $\mathbf{R}^{d_n}_{\Theta,n-m}$ explicitly encodes relative positions by leveraging the additivity of rotation matrices.

For conciseness, we omit the detail of Multi-Head Self-Attention and Feed-Forward Network, where more details can be found in [24]. Finally, we obtain the output of last Transformer layer $F^{(L_t)}$ as the contextualized representation, denoted as $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$.

3.3 Temporal Sparse Interest Capsule Network

Given these contextualized item embeddings, we design a sparse interest capsule network to extract multiple interests. Compared with the existing multiinterest-based methods that utilize vanilla capsule networks, we employ a sparse interest capsule activation mechanism to flexibly determine the number of interests and initialize the coupling coefficient in its dynamic routing stage. In the following, we first present the *Sparse Interest Capsule Activation* mechanism. Then, in the *Temporal Capsule Network*, we employ an attention mechanism to highlight the significance of different positions and time intervals. Finally, in the *Multi-Interest Extractor Layer*, a time-aware dynamic routing mechanism is utilized to adaptively aggregate the user's diverse interests; the key variables of this section are defined in Table 3.

3.3.1 Sparse Interest Capsule Activation. First, we build a base capsule matrix $\mathbf{A} \in \mathbb{R}^{C \times d_n}$ to denote the universal interest space, where C is the interest number, and $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_C]$. Each row $\mathbf{a}_j$ of $\mathbf{A}$ represents a base interest vector, with $\mathbf{a}_j \in \mathbb{R}^{d_n}$ corresponding to a particular interest across all domains. Given a historical interaction sequence s^u, it is intuitive that only a subset of capsules will be activated to reflect the user's diverse interests. To measure the relevance of each

Table 3. Key Variable Definition for Temporal Sparse Interest Capsule Network

Variable	Definition
C	number of interest capsules.
$\mathbf{A}$	base capsule matrix: $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_C]$.
$\mathbf{m}_i$	activation vector for the ith interaction.
$\tilde{\mathbf{P}}$	position embedding matrix in capsule space.
$\tilde{\mathbf{D}}$	time embedding matrix in capsule space.
$\tilde{\mathbf{T}}$	temporal interval embedding matrix in capsules.
$\mathbf{o}$	position-aware routing weight vector.
$\mathbf{r}_i$	latent interest representation for the ith interaction.
$\mathbf{r}'_i$	temporal-aware interest vector.
b_{ij}	routing affinity between the ith interaction and capsule j.
$c_{ij}^{(l)}$	coupling coefficient between the ith interaction and capsule j.
$\mathbf{u}_j^{(l)}$	interest representation for capsule j at iteration l.

contextualized representation $\mathbf{h}_i$ to each base interest vector $\mathbf{a}_j$, we compute the similarity $s_{i,j}$ as follows:

$$s_{i,j} = \text{sim}(\mathbf{h}_i, \mathbf{a}_j) \cdot (1 + \epsilon), \tag{14}$$

where $\text{sim}(\cdot, \cdot)$ denotes the inner product, $\mathbf{a}_j$ is the jth base capsule vector, and ϵ is the Gaussian noise to enable exploration. Then, we activate one interest capsule for each item in the sequence, and generate the corresponding activation vector $\mathbf{m}_i$ as follows:

$$m_{ij} = \begin{cases} -\infty, & \text{if } j \textit{ not in any } \arg\max(\mathbf{s}_i) \\ 0, & \text{otherwise.} \end{cases}, \tag{15}$$

where $m_{ij} = 0$ means that the jth interest capsule is activated, and $\mathbf{s}_i = [s_{i,1}, \ldots, s_{i,C}]$.

3.3.2 Temporal Capsule Network. The utilization of sequential information has been consistently overlooked in many existing solutions when a capsule network is utilized for multi-interest extraction. Therefore, similar to Section 3.2.2, we first add another set of trainable position embedding matrix $\tilde{\mathbf{P}}$ and time embedding matrix $\tilde{\mathbf{D}}$, where $\tilde{\mathbf{P}} \in \mathbb{R}^{N \times d_n}$ and $\tilde{\mathbf{D}} \in \mathbb{R}^{(N_a+N_t) \times d_n}$.

Following Equation (6), we can obtain the corresponding temporal interval embeddings matrix for sequence s^u: $\tilde{\mathbf{T}} = [\tilde{\mathbf{t}}_1, \ldots, \tilde{\mathbf{t}}_N]$. Then, we utilize the attention mechanism with temporal information to calculate the importance of items at different positions:

$$\mathbf{o} = \text{softmax}(\tanh((\mathbf{H} + \tilde{\mathbf{P}} + \tilde{\mathbf{T}})\mathbf{W}_3 + \mathbf{b}_3)\mathbf{W}_4 + \mathbf{b}_4), \tag{16}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d_n \times d_n}$ and $\mathbf{W}_4 \in \mathbb{R}^{d_n \times N}$ are the weight matrices, $\mathbf{b}_3 \in \mathbb{R}^{d_n}$ and $\mathbf{b}_4 \in \mathbb{R}^N$ are the bias, and $\tanh(\cdot)$ denotes the non-linear activation function.

Then, we further utilize a **fully connected (FC)** layer including residual connection [6] and layer normalization [1] to derive the latent interest representation for each item:

$$\mathbf{r}_i = \text{LayerNorm}(\tanh(\mathbf{h}_i\mathbf{W}_5 + \mathbf{b}_5) + \mathbf{h}_i), \tag{17}$$

where $\mathbf{W}_5 \in \mathbb{R}^{d_n \times d_n}$ and $\mathbf{b}_5 \in \mathbb{R}^{d_n}$ are trainable parameters. At last, these latent interest representations are then weighted based on the temporal information as follows:

$$\mathbf{r}'_i = o_i \mathbf{r}_i, \tag{18}$$

where o_i is the ith element of $\mathbf{o}$, denotes the importance of the ith item.

3.3.3 Multi-Interest Extractor Layer. Based on Equation (15), each item in the sequence only activates one interest capsule. We then choose to aggregate the items that activate the same interest capsule via dynamic routing. Specifically, we can calculate the output of the jth capsule in iteration l as follows:

$$\mathbf{z}_j^{(l)} = \sum_{i=1}^{N} c_{ij}^{(l)} \mathbf{r}_i' \tag{19}$$

$$\mathbf{u}_j^{(l)} = \frac{\|\mathbf{z}_j^{(l)}\|^2}{\|\mathbf{z}_j^{(l)}\|^2 + 1} \frac{\mathbf{z}_j^{(l)}}{\|\mathbf{z}_j^{(l)}\|}, \tag{20}$$

where $c_{ij}^{(l)}$ is the coupling coefficient indicating the importance of the ith item towards the jth interest. Here, after the weighted aggregation, a non-linear squash function is used to obtain the jth capsule's interest representation in Equation (20). For the jth capsule that is not activated by any item in the sequence, we expect that the corresponding $c_{ij}^{(l)}$ should be zero.

Recall that we mark this relationship with an activation vector $\mathbf{m}_i$ in Equation (15). Hence, it is straightforward to calculate the coupling coefficients as follows:

$$c_{ij}^{(l)} = \frac{\exp(b_{ij}^{(l)} + m_{ij})}{\sum_{k=1}^{C} \exp(b_{ik}^{(l)} + m_{ik})}, \tag{21}$$

where b_{ij} denotes the routing affinity between the ith item and the jth interest capsule. We can see that m_{ij} helps restrict the dynamic routing scope of these activated interest capsules, and c_{ij} only works when the jth interest is activated (*i.e.,* $m_{ij} = 0$). Then, we use a residual connection to calculate $b_{ij}^{(l+1)}$ iteratively:

$$b_{ij}^{(l+1)} = b_{ij}^{(l)} + \mathbf{r}_i'^{\mathrm{T}} \mathbf{u}_j^{(l)}. \tag{22}$$

It is worth noting that we initialize $b_{ij}^{(0)} = s_{ij}$ to prevent interest coupling, instead of initializing it to zero or random values as done in previous work [2, 12]. We repeat the dynamic routing process L_c times, and denote the output of last iteration $\mathbf{u}_j^{(L_c)}$ as $\mathbf{u}_j$, which represents the user's pre-refined interest representation, where $j \in I_u$ and $I_u = \left\{ j \mid \exists\, i : m_{ij} = 0 \right\}$ includes all activated interest capsules.

3.4 Sequential Interest Refinement

In this section, we delve further into the refinement of user interests. After capturing the multi-interest representations of users, we propose an *Interest Refine Encoder* to explicitly model the variations of user interests at the sequence level. Following this, we implement a *Fusion Gate* mechanism to adaptively incorporate item embeddings into the adjustment of sequence-level interest representations, thereby enhancing the precision of our predictions regarding the variation of user interests. In the following, we will provide a detailed explanation of this process; the key variables of this section are defined in Table 4.

3.4.1 Interest Refine Encoder. Considering the limited ability of the sparse capsule network to model temporal information, we further devise a sequential process for interest refinement. Specifically, for historical interaction sequence s^u, we form a reordered interest representation as follows:

$$\overline{\mathbf{F}}^{(0)} = [\mathbf{u}_{a_1}, \ldots, \mathbf{u}_{a_N}], \tag{23}$$

Table 4. Key Variable Definition for Sequential Interest Refinement

Variable	Definition
$\mathbf{u}_{a_i}$	pre-refined interest representation for the ith interaction.
$\overline{\mathbf{F}}^{(0)}$	reordered interest representation: $\overline{\mathbf{F}}^{(0)} = [\mathbf{u}_{a_1}, \ldots, \mathbf{u}_{a_N}]$.
$\overline{\mathbf{H}}$	sequential interest representation: $\overline{\mathbf{H}} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$.
$\mathbf{H}'$	GRU-enhanced interest representation: $\mathbf{H}' = [\mathbf{h}'_1, \ldots, \mathbf{h}'_N]$.
$\mathbf{w}_i$	adaptive gating weight vector for the ith interaction.
$\hat{\mathbf{h}}_i$	latent refined representation for the ith interaction.
$\overline{\mathbf{u}}_i$	post-refined interest representation for the ith interaction.

GRU, gated recurrent unit.

where subscript $\mathbf{u}_{a_i}$ denotes the pre-refined interest representation activated by the ith item (*i.e.,* $a_i = \arg\max(\mathbf{s}_i)$). Next, we introduce a module that employs another L_t-layer Transformer network with RoPE to explicitly model the sequential patterns between the activated interests in the sequence, thus deepening the model's understanding of user interest trends over time.

$$\overline{\mathbf{F}}^{(l)} = \text{FFN}(\text{RoMHA}(\overline{\mathbf{F}}^{(l-1)})). \tag{24}$$

Finally, we obtain the output of last Transformer layer $\overline{\mathbf{F}}^{(l)}$, denoted as $\overline{\mathbf{H}} = [\overline{\mathbf{h}}_1, \ldots, \overline{\mathbf{h}}_N]$, where $\overline{\mathbf{h}}_i \in \mathbb{R}^{d_n}$. In this way, the activated interest for each historical item is refined according to their sequential patterns as a whole.

3.4.2 Fusion Gate. Although the interest-level sequential patterns are used above for interest refinement, the same interest activated at different positions remains the same in $\overline{\mathbf{F}}^{(0)}$. To address this, we propose a *Fusion Gate* to further refine the interest representation by integrating features between interests and items. Taking into account the **gated recurrent units (GRU)**'s superiority in encoding temporal information, we employ the GRU to refine the contextualized representation $\mathbf{H} \in \mathbb{R}^{N \times d_n}$ as follows:

$$[\mathbf{h}'_1, \ldots, \mathbf{h}'_N] = \text{GRU}([\mathbf{h}_1, \ldots, \mathbf{h}_N]), \tag{25}$$

where $\mathbf{h}'_i \in \mathbb{R}^{d_n}$ denotes the GRU-enhanced interest representation at the ith position. To adaptively fuse temporal dynamics with content features, we design a gating mechanism. This mechanism utilizes a FC layer to generates an adaptive gating weight vector $\mathbf{w}_i \in \mathbb{R}^{d_n}$ for each item i, which helps to extract the importance of each hidden feature:

$$\mathbf{w}_i = \text{sigmoid}(\mathbf{h}'_i \mathbf{W}_6 + \mathbf{b}_6), \tag{26}$$

where $\mathbf{W}_6 \in \mathbb{R}^{d_n \times d_n}$ is the weight matrix, $\mathbf{b}_6 \in \mathbb{R}^{d_n}$ is the bias. Subsequently, an element-wise multiplication is performed, resulting in feature-wise refinement as follows:

$$\hat{\mathbf{h}}_i = \mathbf{w}_i \odot \overline{\mathbf{h}}_i. \tag{27}$$

After the sequential refinement, for each activated interest, we select the refined representation by the latest item that activates this interest as the post-refined interest representation as follows:

$$j_{max} = \max(\{i |\ \arg\max(\mathbf{s}_i) = j,\ i \in [1, N]\}) \tag{28}$$

$$\overline{\mathbf{u}}_j = \hat{\mathbf{h}}_{j_{max}}. \tag{29}$$

3.5 Model Training and Deployment

In this section, we outline the training and inference protocol for our HORAE. As shown in Figure 2, we first introduce the primary training objective for recommending the next item. Secondly, we design four novel interest-level unsupervised tasks during the pre-training stage to enhance universal multi-interest learning: *Interest-Intra Learning*, *Interest-Inter Learning*, *Anchor-based Contrastive Learning*, and *Interest Self-Augmentation*.

3.5.1 Predict Next Item. For a given interaction sequence, its corresponding target item v_i is treated as a positive sample, while the target items from other interaction sequences within the same batch are considered negative samples. We optimize the model using the following cross-entropy loss function:

$$\mathcal{L}_{rec}^{pre-train} = -\log \frac{\exp(\hat{y}_{u,v_i})}{\sum_{j=1}^{B} \exp(\hat{y}_{u,v_j})}, \tag{30}$$

where B represents the training batch size, $\hat{y}_{u,v_j}$ represents the recommendation score of user u for item v_j. Since a user could hold multiple interests, the recommendation score between a user and a candidate item v_j is calculated by applying the max-pooling as follows:

$$\hat{y}_{u,v_j} = \max(\overline{\mathbf{u}}_i^{\mathrm{T}} \mathbf{x}_j | i \in I_u), \tag{31}$$

where the representation $\mathbf{x}_j$ of item v_j is derived according to Equation (2).

3.5.2 Interest-Intra Learning. Given that each item activates one interest, the items that activate the same interest should be semantically similar to each other. This inspires us that the interest representation should be closer to the items that activate it. Specifically, we design the interest-intra training objective as follows:

$$\mathcal{L}_{intra} = \sum_{i=1}^{N} \|\hat{\mathbf{h}}_i - \mathbf{h}_i\|_2, \tag{32}$$

where $\|\cdot\|_2$ represents the Euclidean distance, $\hat{\mathbf{h}}_i$ and $\mathbf{h}_i$ represent the post-refined interest representation and the contextualized representation at the ith position in the sequence for user u, respectively.

3.5.3 Interest-Inter Learning. Since our proposed HORAE is devised to achieve universal multi-interest learning for superior generalization across different domains, we expect that the base universal interests could cover a broad and diverse semantic scope. Therefore, we choose to ensure orthogonality between interest capsules and encourage a balanced distribution among them. From a regularization perspective, we introduce two auxiliary loss terms. To enforce orthogonality between base interest vectors, we impose the following constraints on the base capsule vectors in $\mathbf{A}$:

$$\mathcal{L}_c = \sum_{i=1}^{C} -\log \frac{\cos(\mathbf{a}_i, \mathbf{a}_i)}{\sum_{j=1}^{C} \cos(\mathbf{a}_i, \mathbf{a}_j)}, \tag{33}$$

where $\cos(\cdot,\cdot)$ represents cosine similarity function. Additionally, we choose to maintain a uniform distribution of activated capsules to ensure comprehensive coverage of fine-grained interests. Here, we decide to initially compute the similarity distribution $\mathbf{w}'$ between items and base vectors:

$$\mathbf{w}' = \frac{1}{B} \sum_{i=1}^{B} \mathbf{s}_i. \tag{34}$$

Then, the capsule activation distribution of the same batch is calculated as follows:

$$\mathbf{b}_c = \frac{1}{B}\sum_{i=1}^{B}\mathbf{m}_i. \tag{35}$$

The coverage regularization term is then computed as:

$$\mathcal{L}_b = C \cdot \mathbf{w}'^T \mathbf{b}_c. \tag{36}$$

Finally, we define the training objective of Interest-Inter as follows:

$$\mathcal{L}_{inter} = \mathcal{L}_c + \mathcal{L}_b. \tag{37}$$

3.5.4 Anchor-Based Contrastive Learning. Additionally, we consider an anchor-based contrastive learning process to guide the refinement of interest representations. In simple terms, we consider the recommendation scores obtained based on partially refined interests as *anchor*. Inspired by [16], an effective refinement process could ensure that the recommendation scores obtained using fully refined interest representations should be higher than the *anchor*, while those obtained using pre-refined interest representations should be lower than the *anchor*. Hence, for a given user u and a target item v_i, we can calculate two recommendation scores as follows:

$$\mathcal{S}_i^{pre} = \frac{\exp(\tilde{y}_{u,v_i})}{\sum_{j=1}^{B}\exp(\tilde{y}_{u,v_j})} \tag{38}$$

$$\mathcal{S}_i^{post} = \frac{\exp(\hat{y}_{u,v_i})}{\sum_{j=1}^{B}\exp(\hat{y}_{u,v_j})} \tag{39}$$

$$\tilde{y}_{u,v_j} = \max(\mathbf{u}_i^T \mathbf{x}_j | i \in I_u), \tag{40}$$

where $\mathcal{S}_i^{pre}$ and $\mathcal{S}_i^{post}$ are the likelihood scores by using pre-refined and fully post-refined interest representations respectively. $\hat{y}_{u,v_j}$ is calculated by Equation (31). We also mask some of the interests randomly in the reordered interest representation $\overline{\mathbf{F}}^{(0)}$ with the special token $[\text{MASK}_1]$ and denote the resulting post-refined interest representation as $\check{\mathbf{u}}_*$. Similarly, we calculate the likelihood score as follows:

$$\mathcal{S}_i^{a} = \frac{\exp(\check{y}_{u,v_i})}{\sum_{j=1}^{B}\exp(\check{y}_{u,v_j})} \tag{41}$$

$$\check{y}_{u,v_j} = \max(\check{\mathbf{u}}_i^{\mathrm{T}} \mathbf{x}_j | i \in I_u). \tag{42}$$

As discussed above, the anchor-based contrastive learning objective can be formulated as follows:

$$\begin{aligned}\mathcal{L}_{pre-post} = &-\log(\text{sigmoid}(\mathcal{S}_i^{post} - \mathcal{S}_i^{a})) \\ &-\log(\text{sigmoid}(\mathcal{S}_i^{a} - \mathcal{S}_i^{pre})) \\ &+\beta\mathcal{L}_{rec}^{pre}\end{aligned} \tag{43}$$

$$\mathcal{L}_{rec}^{pre} = -\log\frac{\exp(\tilde{y}_{u,v_i})}{\sum_{j=1}^{B}\exp(\tilde{y}_{u,v_j})}. \tag{44}$$

Here, we add $\mathcal{L}_{rec}^{pre}$ at the end of Equation (43) to ensure the stability of the lower bound, and β is the hyperparameter to control its strength.

3.5.5 Interest Self-Augmentation. Furthermore, we augment each interaction sequence by randomly masking some items using a special token $[\text{MASK}_2]$. For the original sequence, we use $\widetilde{\mathbf{u}}_*$ to represent the user's post-refined interest representation corresponding to the activated interest of the masked item, for example, the ith masked item within the same batch. Consequently, the corresponding augmented interest representation $\widetilde{\mathbf{u}}'_i$, is considered as a positive sample. In the comparison, we select the other masked item from the same batch sequence as negative samples. However, due to the activation mechanism controlled by the sparse capsule activation, the capsules activated in the augmented sequence may not be consistent with the original capsules. Therefore, we enforce the augmented sequence of capsule activation vectors to remain consistent with the original sequence of capsule activation vectors. In this sense, we can perform the following interest self-augmentation:

$$\mathcal{L}_{aug} = \sum_{i=1}^{B_m} -\log \frac{\exp(\widetilde{\mathbf{u}}_i^{\mathrm{T}} \widetilde{\mathbf{u}}'_i)}{\sum_{j=1}^{B_m} \exp(\widetilde{\mathbf{u}}_i^{\mathrm{T}} \widetilde{\mathbf{u}}'_j)}, \tag{45}$$

where B_m represents the total number of masked items in a batch. Specifically, we randomly mask the interaction items in each sample according to a certain ratio. Notably, we do not mask samples with a sequence length shorter than 5 nor we mask the last item of each sample in the training set.

3.5.6 Pre-training and Fine-Tuning. In the pre-training stage, the final objective is defined as a linear combination of these five objectives:

$$\begin{aligned}\mathcal{L}_{pre-train} = \mathcal{L}_{rec}^{pre-train} + \theta_1 \mathcal{L}_{intra} + \theta_2 \mathcal{L}_{inter} \\ + \theta_3 \mathcal{L}_{pre-post} + \theta_4 \mathcal{L}_{aug},\end{aligned} \tag{46}$$

where $\theta_1, \theta_2, \theta_3, \theta_4$ are the hyperparameters to control the impact of different tasks.

In the fine-tuning stage, we freeze the following temporal encoder and the parameters of the sequential encoders, which include $[\mathbf{P}, \text{FFN}, \text{RoMHA}, \text{GRU}]$, corresponding to the modules marked with a snowflake in Figure 2. We then fine-tune the other modules' parameters and use cross-entropy to calculate the recommendation loss. It should be noted that in the fine-tuning stage, we create an item embedding table $\mathbf{V} \in \mathbb{R}^{|V_f| \times d_n}$ for the items in the downstream dataset D_f, where V_f and $|V_f|$ denote the item set and the number of items in the downstream dataset, respectively. For item v_i, we sum the MoE-enhanced textual representation $\mathbf{x}_i$ and the corresponding ID embedding $\mathbf{e}_i$ and feed it into the transformer encoder for generating the contextualized representation. The fine-tuning loss is then formulated as follows:

$$\mathcal{L}_{rec}^{fine-tune} = -\log \frac{\exp(\hat{y}_{u,v_i})}{\sum_{v_j \in V_f} \exp(\hat{y}_{u,v_j})}, \tag{47}$$

$$\mathcal{L}_{fine-tune} = \mathcal{L}_{rec}^{fine-tune} + \theta_5 \mathcal{L}_{inter}, \tag{48}$$

where θ_5 is the hyperparameter.

We conduct a detailed time complexity analysis of the model components, and the specific time complexity for each component in a single batch is shown in Table 5. Since we precompute and store the embeddings for all items using a fixed BERT model, during both training and inference, the corresponding embeddings can be retrieved via indexing. In the pre-training stage, as the batch size B is much larger than other variables, as shown in Equation (46), the model's time complexity is primarily dominated by $O(B^2 C d_n)$, with most of the computational time spent on calculating $\mathcal{L}_{rec}^{pre-train}$, $\mathcal{L}_{pre-post}$ and $\mathcal{L}_{aug}$. In the fine-tuning stage, since both B and $|V_f|$ are significantly larger than other variables, the model's time complexity is primarily dominated by $O(BC|V_f|d_n)$.

Table 5. Time Complexity Analysis of Different Components

Component	Complexity
Text-Aware Temporal Item Embedding	
$\text{BERT}_{\text{base}}$	$O(\lvert V\rvert))$
MoE Adapter	$O(Bd_bG + Bd_bd_n)$
Temporally Contextual Item Embedding	$O(BN^2d_n + BNd_n{}^2)$
Temporal Sparse Interest Capsule Network	
Sparse Interest Capsule Activation	$O(BNCd_n)$
Temporal Capsule Network	$O(BNd_n{}^2 + BNd_n)$
Multi-Interest Extractor Layer	$O(BNCd_nL_c)$
Sequential Interest Refinement	
Interest Refine Encoder	$O(BN^2d_n + BNd_n{}^2)$
Fusion Gate	$O(BN^2d_n + BNd_n)$
Model Training and Deployment	
Interest-Intra Learning $\mathcal{L}_{intra}$	$O(BNd_n)$
Interest-Inter Learning $\mathcal{L}_{inter}$	$O(BC + C^2d_n)$
Anchor-based Contrastive Learning $\mathcal{L}_{refine}$	$O(BN^2d_n + BNd_n{}^2 + B^2Cd_n + B^2)$
Interest Self-Augmentation $\mathcal{L}_{aug}$	$O(BN^2d_n + BNd_n{}^2 + B_m{}^2d_n + B_m{}^2)$
Pre-training main task $\mathcal{L}_{rec}^{pre-train}$	$O(B^2Cd_n + B^2)$
Fine-tuning main task $\mathcal{L}_{rec}^{fine-tune}$	$O(BC\lvert V_f\rvert d_n + B\lvert V_f\rvert)$

4 Experimental Setup

We conduct a series of experiments on real-world benchmark datasets to validate the effectiveness of HORAE. Specifically, we focus primarily on the following **research questions (RQs)**:

- *RQ1* Does the proposed HORAE achieve state-of-the-art performance? (ref Section 5.1)
- *RQ2* What impacts do individual components within HORAE have on its performance? (ref Section 5.2)
- *RQ3* How sensitive is HORAE to different hyperparameters? (ref Section 5.3)
- *RQ4* How does HORAE achieve universal multi-interest learning for knowledge transfer? (ref Section 5.4)

4.1 Dataset

For the pre-training stage, we initially select five domain datasets with a substantial volume from the Amazon Review dataset [15] and combine them to create the pre-training dataset D_p. For the fine-tuning stage, we choose an additional five relatively smaller domain datasets from the Amazon Review dataset as well as the Online Retail dataset as downstream datasets D_f. We fine-tune and evaluate HORAE on each dataset within D_f. The details of the datasets are as follows:

Pre-training Dataset. We select a cross-domain dataset consisting of five domains from the Amazon Review dataset where the minimum interaction time interval of this dataset is one day. These five domains include "Grocery and Gourmet Food" (Food), "CDs and Vinyl" (CD), "Kindle Store" (Kindle), "Movies and TV" (Movies), and "Home and Kitchen" (Home).

Fine-Tuning Dataset. We select another five domain datasets from the same Amazon Review dataset. These five domains include "Industrial and Scientific" (Scientific), "Prime Pantry" (Pantry),

Table 6. Dataset Statistics after Preprocessed

Dataset		Users	Items	Interactions	Sparsity
Pre-training	Food	115,349	39,670	1,027,413	-
	CDs	94,010	64,439	1,118,563	-
	Kindle	138,436	98,111	2,204,596	-
	Movies	281,700	59,203	3,226,731	-
	Home	731,913	185,552	6,451,926	-
	D_p	1,361,408	446,975	14,029,229	-
Fine-Tuning	Scientific	8,442	4,385	59,427	1.6‰
	Pantry	13,101	4,898	126,962	1.9‰
	Instruments	24,962	9,964	208,926	0.8‰
	Arts	45,486	21,019	395,150	0.4‰
	Office	87,436	25,986	684,837	0.3‰
	Online Retail	16,520	3,469	519,906	9.0‰

"Musical Instruments" (Instruments), "Arts, Crafts and Sewing" (Arts), and "Office Products" (Office). Furthermore, we select the Online Retail dataset as a cross-platform dataset to validate the generalization ability of HORAE. The Online Retail[2] dataset encompasses all transactional information of non-store online retail that occurred in the UK between 01/12/2010 and 09/12/2011. The minimum interaction time interval of this dataset is 1 minute.

As mentioned in Section 3.1, the pre-training dataset D_p is relatively extensive and semantically rich, whereas the fine-tuning dataset D_f is comparatively smaller and exhibits a semantic gap with the pre-training dataset, particularly in cross-platform scenarios. To ensure consistency in metrics, the dataset selection follows the settings outlined in [8]. We accomplish this by removing items with fewer than 5 interactions, using the 5-core setting for users or items across all datasets. Subsequently, we organize each user's interactions in ascending order based on timestamps, with the last interaction in the sequence representing the most recent interaction. For evaluation, we employ a leave-one-out strategy for dataset splitting. The latest interaction is designated for testing, the second-to-last interaction for validation, and the remaining interactions for training. Table 6 presents the statistics of all preprocessed datasets. It is noteworthy that interactions from the same user across different domains are not combined into a single sequence. For all datasets, we use the title information corresponding to each item as the text input for $\text{BERT}_{\text{base}}$ in Equation (1), which helps to enrich the item representations in HORAE.

4.2 Evaluation Metrics

We utilize two widely used evaluation metrics as follows: (1) *Recall* measures the proportion of relevant items that are retrieved out of the total number of relevant items; (2) **normalized discounted cumulative gain (*NDCG*)** considers both relevance and ranking order of retrieved items. It calculates the quality of a ranked list by assigning higher scores to relevant items that are placed higher in the list.

To prevent evaluation bias caused by sampling, we calculate the recommendation scores for each user against all items, and sort them in descending order to generate a top-K recommendation list. Metric scores are represented as the average for all users in the test set, with K being 10 and 50,

[2]https://doi.org/10.24432/C5BW33.

respectively. For each method, we repeat the experiment 5 times and report the average results. The statistical significance test is conducted by Student *t-test*.

4.3 Baselines

We compare the proposed HORAE against the various state-of-the-art sequential recommendation methods. These methods can be divided into groups with and without pre-training.

Without pre-training:

- *GRU4Rec* [7] utilizes GRU to model user interaction sequences.
- *SASRec* [9] employs the self-attention mechanism to capture long-range dependencies in user-item interaction sequences.
- *FDSA* [30] utilizes a self-attention mechanism to precisely model user behavior preference patterns from the dimensions of items and features.
- *MIND* [12] utilizes a multi-interest capsule network to capture diverse interests from the user interaction sequences.
- *ComiRec* [2] introduces a controllable multi-interest framework for sequential recommendation, featuring a dynamic routing-based multi-interest extractor and an aggregation module, designed to balance the accuracy and diversity of recommendations.
- *TiSASRec* [13] introduces a learnable matrix for each user that captures the item-time interval relationship, leading to effectively encoding temporal information. TiSASRec not only exploits the absolute positions of items but also the time intervals between sequential items.
- *TiCoSeRec* [4] introduces five data augmentation strategies to convert non-uniform sequences into uniform ones, taking into account the variance of time intervals.

With pre-training:

- *S^3-Rec* [31] proposes four self-supervised optimization objectives to maximize the mutual information of context information in different forms or granularities.
- *BERT4Rec* [25] uses a Cloze objective loss and bidirectional self-attention mechanism to model user interaction sequences.
- *UniSRec* [8] employs textual description to learn cross-domain item representations, and subsequently leverages these representations to acquire a universal sequential representation.
- *Miracle* [27] is our recent work, which includes a text-aware encoder and a sparse interest capsule network. As described in Section 2.3, it performs a universal multi-interest modeling with a position-wise dynamic routing, which highlights the importance of each item in the historical interactions.

Here, it is noteworthy that TiSASRec and TiCoSeRec are temporal encoding-based methods.

4.4 Hyperparameter Settings

For a fair comparison, all methods are implemented in Pytorch and learned with Adam [10] optimizer, and their hyperparameters are configured according to the original suggestion from their papers.

Specifically, as to HORAE, the maximum sequence length is set to $N = 20$, the expert number is set to $G = 8$, the number of bins for adjacent temporal differences is set to $N_a = 14$, the number of bins for target temporal differences is set to $N_t = 14$, the transformer layers L_t is selected from $\{1, 2, 3, 4\}$, the embedding dimension is set to $d_n = 300$, the interest capsule number C is selected from $\{16, 24, 32, 40\}$, the capsule layer L_c is set to $\{1, 2, 3, 4\}$. The normalization scaling coefficient is set to $\tau = 86400$ in the Amazon Review dataset, while in the Online Retail dataset, τ is set to 60. In

Table 7. Performance Comparison of Different Methods across the Six Datasets

Dataset	Metrics	Models												
		Without Pre-training							With Pre-training				Ours	
		GRU4Rec	SASRec	FDSA	MIND	ComiRec	TiSASRec	TiCoSeRec	S^3-Rec	BERT4Rec	UniSRec	Miracle	Horae	Improve
Scientific	Recall@10	0.0602	0.1003	0.0881	0.0632	0.0313	0.1181	0.1077	0.0684	0.0650	0.1237	0.1343	**0.1467***	+9.23%
	Recall@50	0.1407	0.2010	0.1656	0.1375	0.0913	0.2484	0.2159	0.1464	0.1471	0.2416	0.2562	**0.2688***	+4.92%
	NDCG@10	0.0330	0.0550	0.0589	0.0402	0.0176	0.0637	0.0695	0.0373	0.0345	0.0663	0.0722	**0.0816***	+13.02%
	NDCG@50	0.0503	0.0769	0.0757	0.0561	0.0302	0.0921	0.0928	0.0541	0.0522	0.0919	0.0988	**0.1082***	+9.51%
Pantry	Recall@10	0.0396	0.0488	0.0406	0.0384	0.0305	0.0507	0.0518	0.0426	0.0385	0.0751	0.0822	**0.0893***	+8.64%
	Recall@50	0.1282	0.1339	0.1159	0.1125	0.1016	0.1605	0.1398	0.1243	0.1281	0.1862	0.2027	**0.2153***	+6.22%
	NDCG@10	0.0197	0.0218	0.0215	0.0199	0.0155	0.0251	0.0279	0.0187	0.0188	0.0352	0.0395	**0.0428***	+8.35%
	NDCG@50	0.0385	0.0400	0.0376	0.0358	0.0306	0.0488	0.0467	0.0363	0.0381	0.0591	0.0656	**0.0700***	+6.71%
Instruments	Recall@10	0.0876	0.1089	0.1099	0.0988	0.0769	0.1212	0.1232	0.1052	0.0984	0.1260	0.1335	**0.1450***	+8.61%
	Recall@50	0.1609	0.1988	0.1958	0.1752	0.1528	0.2251	0.2147	0.1939	0.1862	0.2375	0.2468	**0.2616***	+6.00%
	NDCG@10	0.0610	0.0631	0.0809	0.0722	0.0462	0.0706	0.0883	0.0678	0.0580	0.0699	0.0776	**0.0911***	+3.17%
	NDCG@50	0.0769	0.0826	0.0994	0.0887	0.0626	0.0932	0.1080	0.0870	0.0770	0.0914	0.1021	**0.1164***	+14.01%
Arts	Recall@10	0.0757	0.1045	0.1015	0.0906	0.0507	0.1147	0.1117	0.0970	0.0854	0.1270	0.1362	**0.1465***	+7.56%
	Recall@50	0.1472	0.1924	0.1832	0.1591	0.1125	0.2233	0.1956	0.1846	0.1660	0.2377	0.2530	**0.2643***	+4.47%
	NDCG@10	0.0492	0.0599	0.0705	0.0634	0.0287	0.0664	0.0708	0.0572	0.0493	0.0678	0.0791	**0.0863***	+9.10%
	NDCG@50	0.0647	0.0789	0.0882	0.0783	0.0419	0.0900	0.0871	0.0763	0.0668	0.0919	0.1046	**0.1120***	+7.07%
Office	Recall@10	0.0941	0.1081	0.1132	0.0971	0.0641	0.1200	0.1112	0.1075	0.0934	0.1268	0.1417	**0.1515***	+6.92%
	Recall@50	0.1445	0.1686	0.1740	0.1415	0.1087	0.1983	0.1665	0.1645	0.1447	0.2025	0.2219	**0.2331***	+5.05%
	NDCG@10	0.0711	0.0656	0.0862	0.0739	0.0444	0.0766	0.0857	0.0715	0.0662	0.0775	0.0918	**0.1043***	+13.62%
	NDCG@50	0.0821	0.0787	0.0994	0.0836	0.0540	0.0937	0.0977	0.0839	0.0773	0.0939	0.1093	**0.1221***	+11.71%
Online Retail	Recall@10	0.1495	0.1465	0.1503	0.1463	0.1175	0.1599	0.1495	0.1461	0.1530	0.1554	0.1729	**0.1857***	+7.40%
	Recall@50	0.3784	0.3700	0.3756	0.3331	0.2996	0.3814	0.3212	0.3841	0.3951	0.3906	0.4478	**0.4590***	+2.50%
	NDCG@10	0.0720	0.0703	0.0729	0.0778	0.0578	0.0698	0.0839	0.0676	0.0711	0.0715	0.0795	**0.0911***	+8.58%
	NDCG@50	0.1218	0.1190	0.1218	0.1185	0.0972	0.1184	0.1213	0.1195	0.1221	0.1228	0.1398	**0.1512***	+8.15%

The best and second-best results are highlighted in boldface and underlined, respectively. $*$ indicates that the performance difference against the best result is statistically significant at 0.05 level. "Improve": the relative improvement gain of Horae against best-performing baselines.

the pre-training stage and the fine-tuning stage, the batch size is set to $B = 2048$, the learning rate is set to $\theta = 0.001$. For the multi-interest learning tasks, the hyperparameters are chosen from different sets: $\theta_1 \in \{0.005, 0.01, 0.05, 0.1, 0.5\}$, $\theta_2 \in \{0.5, 1.0, 10, 50, 100\}$, $\theta_3 \in \{0.005, 0.01, 0.05, 0.1, 0.5\}$, $\theta_4 \in \{0.005, 0.01, 0.05, 0.1, 0.5\}$, and $\theta_5 = 10$.

5 Result and Analysis

5.1 Overall Performance (RQ1)

We compare the Horae with the baseline methods in the five cross-domain downstream datasets and one cross-platform downstream dataset. The overall performance is shown in Table 7, from which we have the following observations:

Auxiliary text information and self-attention modeling mechanisms prove beneficial for enhancing the performance of sequential recommendations. Among all the sequential baselines, SASRec and FDSA perform the best. In four datasets (three cross-domain datasets with relatively low sparsity and the cross-platform Online Retail dataset), FDSA's NDCG is significantly higher than that of SASRec, since textual information is used as auxiliary features to improve the model performance. The comparison between GRU4Rec and SASRec indicates the superiority of self-attention in modeling sequential information, a finding consistent with related literature. Furthermore, it is noteworthy that, due to the inconsistency of the cloze task with the next item recommendation task, BERT4Rec does not achieve favorable results across all datasets.

The performance gain achieved by exploiting temporal information varies a lot across different models. Considering the TiSASRec and TiCoSeRec, both of which incorporate temporal information as baselines. Specifically, TiSASRec shows significant improvements compared to SASRec, especially on the cross-platform dataset Online Retail, indicating the effectiveness of building the relation matrix with encoded temporal information, as well as the flexible adjustment based on different dataset time spans. TiCoSeRec particularly shows significant improvement in NDCG, suggesting that converting non-uniformly distributed sequences into more uniformly distributed sequences with smaller variances has an impact on ranking tasks.

Pre-training is an effective strategy for transferring semantics and knowledge across different domains. The pre-training based methods, UniSRec [8] and Miracle, significantly outperform other baselines, demonstrating the effectiveness of integrating textual information from different domains into a universal item representation through pre-training. Moreover, compared to other baseline methods, Miracle achieves the best performance across all metrics in all datasets, and it is obvious that Miracle delivers a substantial improvement over the second-best baseline, UniSRec. These underscore the superiority of incorporating multi-interest learning in pre-training schemes.

Finally, by comparing our proposed Horae with all baseline methods, Horae consistently achieves the best performance in all cases. More specifically, at $K = 10$ and $K = 50$, our model shows an improvement in Recall over the best baseline model, Miracle, with increases ranging from 2.50% to 9.23% across all datasets. Moreover, the improvement in NDCG ranges from 6.71% to 14.01%. These comparisons highlight the superiority of our extensions and optimizations to the original scheme, including temporal information modeling strategies, sequential process of interest refinement, and interest-level contrastive learning works.

5.2 Ablation Study (RQ2)

In this subsection, we first conduct ablation studies on each of the newly added design choices in Horae to demonstrate their effectiveness. Subsequently, we evaluate the inference efficiency of Horae compared to two baseline models. Specifically, the ablations can be categorized into three major variants as follows:

- *Pre-training Unsupervised Tasks*: including Interest-Intra Learning ($\mathcal{L}_{intra}$), Interest-Inter Learning ($\mathcal{L}_{inter}$) and Anchor-based Contrastive Learning ($\mathcal{L}_{pre-post}$).
- *Sequential Interest Refinement*: including the Interest Refine Encoder (Refiner) and the Fusion Gate (Gate).
- *Temporal Modeling*: including the RoPE, the adjacent time difference (d^a_*) and the target time difference (d^t_*).

Table 8 reports the performance of these ablation variants and the full Horae model on each dataset. Here, our model Horae outperforms the variants with individual components removed, achieving the best results in 7 out of 12 metrics across all datasets. Specifically: (1) in terms of the *Temporal Modeling*, removing relative position information and two kinds of relative time interval information led to a significant performance decrease across all datasets, demonstrating the effectiveness of relative-time encoding. (2) As for the *Sequential Interest Refinement*, eliminating the Fusion Gate significantly reduces the model's performance in various metrics, proving the module's superiority in the feature-wise integration between items and interests. Furthermore, removing the Interest Refine Encoder led to a notable performance drop in four datasets, but this trend is not consistent in another two datasets (Arts and Office) with relative dense interactions. We consider this may be due to overfitting caused by interest sequence-level correction. (3) Regarding the *Pre-training Unsupervised Tasks*, the exclusion of interest-inter learning and anchor-based contrastive learning significantly decreases model performance across all datasets, underscoring the effectiveness of base

Table 8. The Ablation Study of Horae on Six Datasets

Model	Dataset											
	Scientific		Pantry		Instruments		Arts		Office		OnlineRetail	
	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
Miracle	0.1343	0.0722	0.0822	0.0395	0.1335	0.0776	0.1362	0.0791	0.1417	0.0918	0.1729	0.0795
w/o $\mathcal{L}_{intra}$	0.1361	0.0770	0.0861	0.0407	0.1409	0.0878	0.1450	0.0859	0.1487	0.1011	**0.1886**	**0.0936**
w/o $\mathcal{L}_{inter}$	0.1417	0.0785	0.0831	0.0401	0.1426	0.0861	0.1447	0.0849	0.1456	0.0967	0.1685	0.0777
w/o $\mathcal{L}_{pre-post}$	0.1434	0.0807	0.0852	0.0410	0.1447	0.0898	0.1454	0.0852	0.1501	0.1040	0.1770	0.0837
w/o Refiner	0.1417	0.0787	0.0858	0.0412	0.1427	0.0895	**0.1472**	**0.0884**	0.1509	**0.1055**	0.1780	0.0838
w/o Gate	0.1417	0.0787	0.0851	0.0404	0.1432	0.0885	0.1433	0.0848	0.1485	0.1026	0.1701	0.0780
w/o RoPE	0.1444	0.0809	0.0869	0.0417	0.1446	**0.0911**	0.1467	0.0866	0.1495	0.1042	0.1818	0.0878
w/o d_*^a	0.1445	0.0806	0.0889	0.0423	0.1448	0.0908	0.1458	0.0870	0.1496	0.1045	0.1756	0.0840
w/o d_*^t	0.1406	0.0779	0.0834	0.0406	0.1398	0.0871	0.1405	0.0820	0.1451	0.1005	0.1811	0.0862
Horae	**0.1467**	**0.0816**	**0.0893**	**0.0428**	**0.1450**	**0.0911**	0.1465	0.0863	**0.1515**	0.1043	0.1857	0.0911

The best and second-best results are highlighted in boldface and underlined, respectively.

Table 9. The Average Inference Time and Relative Increase of Different Methods on the Instruments Dataset

Model	Average Inference Time (s)	Relative Increase (%)
UniSRec	8.67	-
Miracle	9.36	7.96%
Horae	9.92	14.42%

capsule domain regularization and anchor-based refinement. Besides, removing the interest-intra learning function led to a remarkable decrease in metrics in cross-domain datasets, but metrics on cross-platform dataset tend to improve. This might be due to the over-narrowing of the gap between item representations and their activated interest representations, which reduces the learning of the universal interest capsule.

Inference Efficiency. We evaluate the average inference time over ten runs for Horae, Miracle, and UniSRec on the Instruments dataset. Each model is run on the same workstation, which includes a 12-core CPU, 120 GB of memory, a 250 GB disk, and an A100 GPU. The batch size is fixed as 2,048. The software versions are Python 3.8, Pytorch 2.1.0, and CUDA 12.1, respectively. As shown in Table 9, we observe that UniSRec has the fastest inference speed, while our model Horae has the slowest. This is reasonable because Miracle introduces a sparse interest capsule network compared to UniSRec, and Horae further includes a Temporal Sparese Interest Capsule Network and a Sequential Interest Refinement structure compared to Miracle. These additional structures slightly reduce the inference speed but enable more accurate recommendations.

5.3 Hyperparameter Sensitivity (RQ3)

In this subsection, we analyze the sensitivity of Horae to different parameters. Following this, we experiment with the hyperparameter weights assigned to the multi-interest learning tasks during the pre-training stage. Furthermore, we examine the impact of substituting certain components within the model architecture.

The Impact of Transformer Layer. As illustrated in Figure 4, regarding the number of transformer layers L_t, we can observe that as L_t increases from 1 to 2, Horae shows a slight improvement in

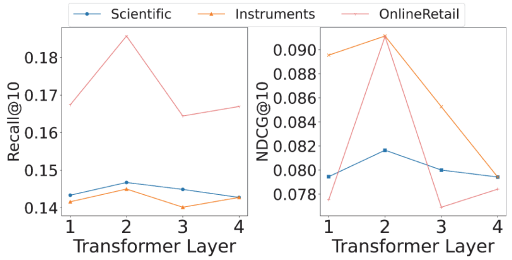

Fig. 4. Model Performance with different transformer layer L_t.

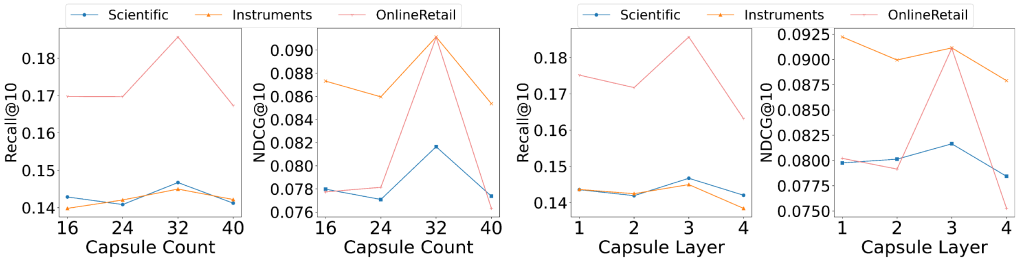

Fig. 5. Model Performance with different Capsule Count C and Capsule Layer L_c.

Recall but a significant rise in NDCG. However, both performance declines are observed at $L_t = 3$. This contradicts the general belief that larger pre-trained models yield better results. Possible reasons include: (1) the limited amount of pre-training data, where an overly large model could overfit; (2) the freezing of the sequence encoder's parameters during the fine-tuning stage, making it difficult for a larger model to adapt to downstream datasets.

The Impact of Capsule Count and Layer. Observing Figure 5, we could find that HORAE achieves the best overall results when the interest capsule number $C = 32$. As C increases, performance declines, possibly due to the overly granular interests created by too many capsules, leading to reduced effectiveness. For the interest capsule layers L_c, there is no significant difference in performance between 1 and 2. At $L_c = 3$, there is an improvement observed in two cross-domain datasets, with a significant enhancement in the Online Retail dataset. However, at $L_c = 4$, a performance decline is observed. Possible reasons include: (1) Insufficient iterations for dynamic routing at lower layers to calculate scores adequately; (2) Overfitting with an excessive number of layers, particularly resulting in poor performance on cross-platform datasets with significant differences in data distributions.

The Impact of Multi-Interest Learning Tasks. In the pre-training stage, the four multi-interest learning tasks are integral to our model design. We separately examine the impact of each component on the overall effect, as shown in Figure 6. Specifically, when the weight becomes large, the model's performance on the three datasets generally shows an initial increase followed by a decrease. This is reasonable since extremely small weight values are almost equivalent to the ablation settings in Section 5.2, while excessively large weights may deteriorate user preference learning. Furthermore, we can observe that HORAE shows more fluctuations in terms of NDCG compared to the HR metric. Considering that our multi-interest learning tasks mainly focus on enhancing fine-grained adjustments for activating interests in items, this indicates the effectiveness of our interest-level unsupervised learning for ranking tasks.

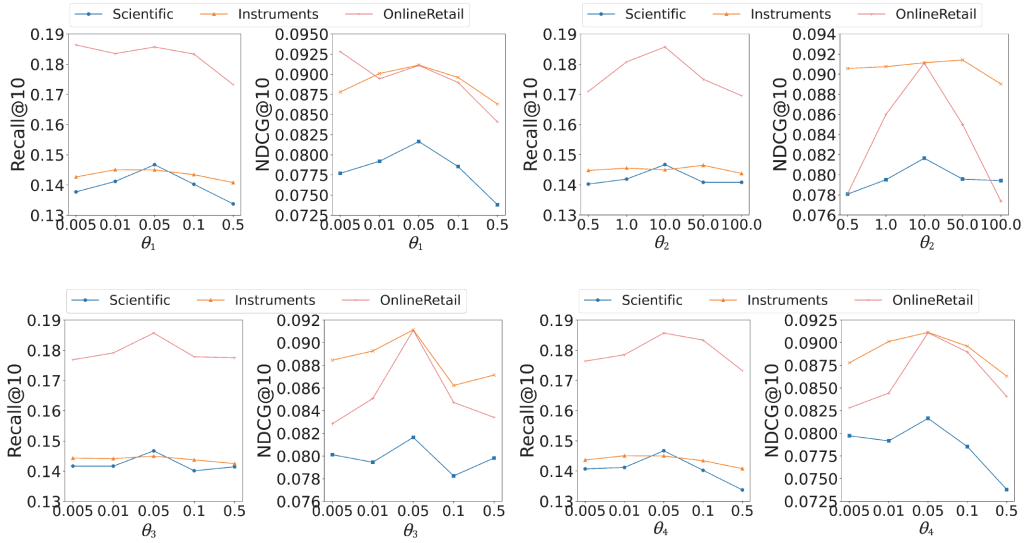

Fig. 6. Model Performance with different hyperparameters of four multi-interest learning tasks: θ_1 for Interest-Intra Learning, θ_2 for Interest-Inter Learning, θ_3 for Anchor-based Contrastive Learning, and θ_4 for Interest Self-Augmentation.

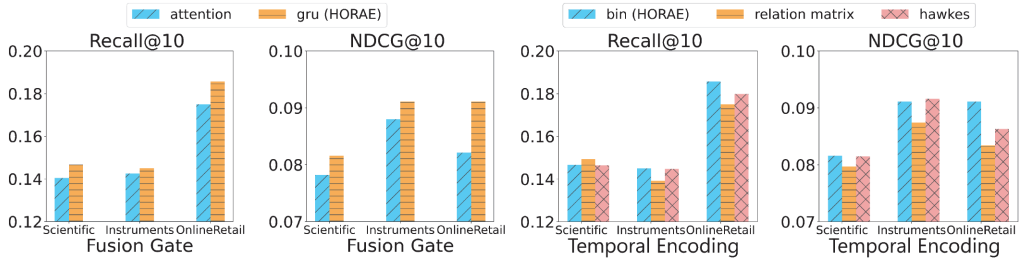

Fig. 7. Model Performance with different Fusion Gates and Temporal Encoding.

The Impact of Fusion Gates. We modify some modules of HORAE to other variant forms. In the Fusion Gate, we replace the GRU with self-attention mechanism for testing, as illustrated in Figure 7. It can be observed that using GRU as the backbone network results in better performance when the Fusion Gate encodes the item sequence and adaptively fuses it with the interest sequence. This is reasonable since GRU is more effective at encoding temporal sequence compared to self-attention mechanisms, suggesting that temporal patterns are important for sequential interest refinement.

The Impact of Temporal Encoding. For temporal encoding, we replace our time-binning function $f_t(\cdot)$ with the following approaches: (1) Relation Matrix: we adopt the method from TiSASRec [13], establishing a personalized item-time interval relation matrix for each user. This involves creating a learnable embedding matrix for relative time intervals in the self-attention network, similar to position embeddings, and performing element-wise addition with the input embedding. (2) Hawkes Encoding: following the scheme in [32], we employ Hawkes encoding for each item in the sequence based on timestamp information.

Figure 7 reports the performance when using different methods for temporal encoding. We observe that the time-binning and the hawkes encoding outperform the relation matrix approach,

Fig. 8. The index of each item is labeled below each item. The first row illustrates the user's historical purchasing behavior and corresponding purchase dates. The second row shows the top-3 item recommended by our model and the variant model that removes three kinds of temporal information respectively. Items framed in different colors represent varying interests, and an item completely filled with color is the ground truth.

possibly because the former two methods not only fuse relative order information at the item sequence-level but also adaptively adjust the routing weights based on fine-grained temporal information for multi-interest learning. In cross-domain datasets, the performance of these two methods is comparable, but in cross-platform dataset, the time-binning based approach performs better, possibly due to the large gaps in time intervals across platforms, making the time-binning method more stable for transferring the temporal patterns.

In summary, through hyperparameter experiments, we find that the optimal combination of hyperparameters is $L_t = 2, C = 32, L_c = 3, \theta_1 = 0.05, \theta_2 = 10, \theta_3 = 0.05, \theta_4 = 0.05$. Furthermore, we demonstrate the superiority of our approach by replacing the proposed *Temporal Encoding* and *Fusion Gate* components with alternative ones, showing that our method outperforms these alternatives.

5.4 Case Study (RQ4)

Temporal Modeling. Utilizing HORAE and several variants, we examine the top-3 items recommended to underscore the crucial role of our temporal modeling mechanisms. We delve into two selected examples from Pantry dataset as follows: (1) Figure 8 sheds light on the impact of temporal modeling in distinguishing both long-term and short-term interests. The user purchased a substantial quantity of potato chips on 26 April 2016, followed by an almost 2-year purchasing hiatus, before acquiring biscuits on 06 March 2018. The variant model, hampered by its lack of temporal insight, focuses on both the recent biscuit purchase and the 2-year-old potato chip purchases equally, neglecting the considerable time lapse. Our model, taking relative time information into account, effectively identifies the user's present interests and accurately recommends other kinds of biscuits instead. (2) The second example, depicted in Figure 9, features a user with recent yet intensive interactions, showing varied tastes for different flavors of the same product. More specifically, this user purchased item 2738 on 22 August 2015, followed by item 500 on 29 August 2015, revealing a periodic preference for varying mashed potato flavors. Our model, in capturing this trend of cyclical purchases, delivers correct prediction. In contrast, the temporal-unaware variant model fails to capture the cyclical purchasing preferences of the user. The above two examples effectively demonstrate the efficacy of our approach in jointly modeling various kinds of fine-grained temporal information.

Activated Capsules Per User. It is interesting to further check how our model utilizes users' historical interactions to activate personalized multiple interests. In Figure 10, we randomly sample

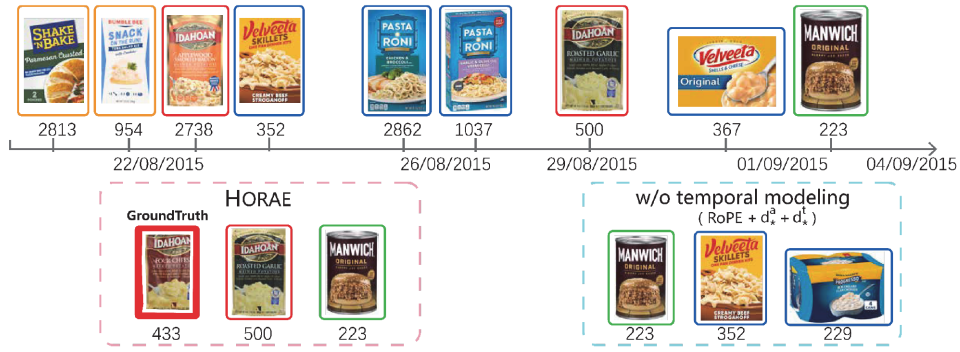

Fig. 9. Another example from Pantry dataset, following the same format as Figure 8, demonstrates the importance of temporal modeling in capturing cyclical interests.

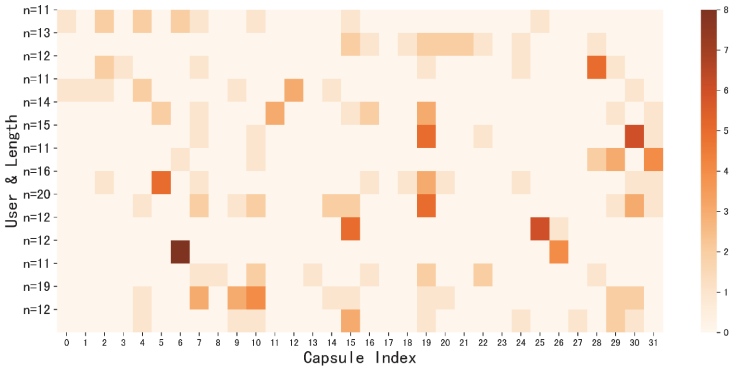

Fig. 10. The x-axis represents the capsule index, while the y-axis shows the sequence length of each user. In the heatmap, the darker the color, the more items activate that capsule.

14 users from Scientific dataset to visualize the distribution of the activated interests by the *Temporal Sparse Interest Capsule Network*. Here, we can observe that the activation of interest capsules is dynamic. The sequences of different lengths may have the same number of interests, while the sequences of the same length can also have varying numbers of activated interests.

Activated Capsules Per Item. Then, we also investigate whether similar items tend to activate the same capsule. In Figure 11, we select five groups of semantically similar items (similarity can be measured based on the textual content of the items through dot product) from Scientific dataset and check the distribution of capsules activated for each item accordingly. We observe that items within the same group have very similar global capsule activation patterns, while items from different groups show obvious differences instead. This indicates that similar items activate the same interests, while dissimilar items activate different interests, aligning with our design purpose for the multi-interest learning.

6 Conclusion

In this article, we propose a novel temporal multi-interest pre-training framework HORAE for sequential recommendation on the basis of our recent work, MIRACLE. We consider relative position information and two kinds of relative time interval information jointly when performing multi-interest learning. Then, we design a sequential interest refinement to capture the subtle nuances of

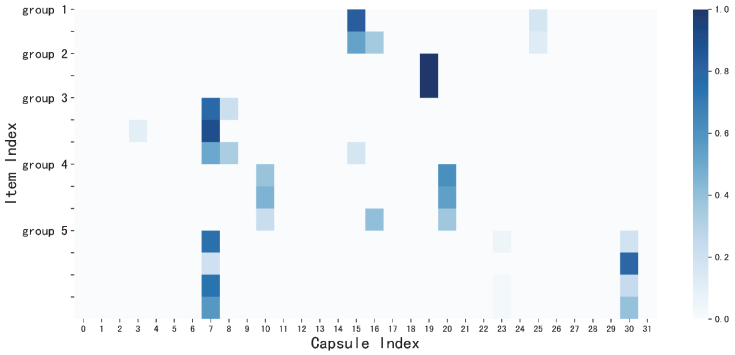

Fig. 11. The x-axis represents the capsule index, and the y-axis represents each item, with $[group_i, group_{i+1})$ indicating a set of semantically similar item representations.

how interests change and shift along the timeline. Besides, four interest-level unsupervised tasks during the pre-training stage are proposed to enhance universal multi-interest learning. Empirical results on five cross-domain datasets and one cross-platform dataset demonstrate that our model performs better than state-of-the-art baselines.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. arXiv:1607.06450. Retrieved from https://arxiv.org/abs/1607.06450

[2] Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2942–2951.

[3] Zheng Chai, Zhihong Chen, Chenliang Li, Rong Xiao, Houyi Li, Jiawei Wu, Jingxu Chen, and Haihong Tang. 2022. User-aware multi-interest learning for candidate matching in recommenders. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1326–1335.

[4] Yizhou Dang, Enneng Yang, Guibing Guo, Linying Jiang, Xingwei Wang, Xiaoxiao Xu, Qinghui Sun, and Hong Liu.2024. TiCoSeRec: Augmenting data to uniform sequences by time intervals for effective recommendation. *IEEE Transactions on Knowledge and Data Engineering* 36, 6 (2024), 2686–2700.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805. Retrieved from https://arxiv.org/abs/1810.04805

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

[7] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. arXiv:1511.06939. Retrieved from https://arxiv.org/abs/1511.06939

[8] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 585–593.

[9] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[10] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv:1412.6980. Retrieved from https://arxiv.org/abs/1412.6980

[11] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. arXiv:2011.05864. Retrieved from https://arxiv.org/abs/2011.05864

[12] 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.

[13] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 322–330.

[14] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S. Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive self-supervised sequential recommendation with robust augmentation. arXiv:2108.06479. Retrieved from https://arxiv.org/abs/2108.06479

[15] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 188–197.

[16] Yuqi Qin, Pengfei Wang, and Chenliang Li. 2021. The world is binary: Contrastive learning for denoising next basket recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 859–868.

[17] Ruihong Qiu, Zi Huang, Tong Chen, and Hongzhi Yin. 2021. Exploiting positional information for session-based recommendation. *ACM Transactions on Information Systems (TOIS)* 40, 2 (2021), 1–24.

[18] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 579–588.

[19] Ruihong Qiu, Hongzhi Yin, Zi Huang, and Tong Chen. 2020. Gag: Global attributed graph neural network for streaming session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 669–678.

[20] Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. 2021. U-BERT: Pre-training user representations for improved recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 4320–4327.

[21] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 811–820.

[22] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. In *Proceedings of the Advances in Neural Information Processing Systems* 30.

[23] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv:1701.06538. Retrieved from https://arxiv.org/abs/1701.06538

[24] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. arXiv:2104.09864. Retrieved from https://arxiv.org/abs/2104.09864

[25] 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.

[26] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 598–606.

[27] Zuoli Tang, Lin Wang, Lixin Zou, Xiaolu Zhang, Jun Zhou, and Chenliang Li. 2023. Towards multi-interest pre-training with sparse capsule network. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 311–320.

[28] Yu Tian, Jianxin Chang, Yanan Niu, Yang Song, and Chenliang Li. 2022. When multi-level meets multi-interest: A multi-grained neural model for sequential recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1632–1641.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems* 30.

[30] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level deeper self-attention network for sequential recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 4320–4326.

[31] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1893–1902.

[32] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. 2020. Transformer Hawkes process. In *Proceedings of the International Conference on Machine Learning. PMLR*, 11692–11702.

Received 11 July 2024; revised 18 February 2025; accepted 28 March 2025