Scaling New Frontiers: Insights into Large Recommendation Models

Wei Guo^{1†}, Hao Wang^{2†}, Luankang Zhang^{2†}, Jin Yao Chin^{1†}, Zhongzhou Liu¹, Kai Cheng², Qiushi Pan², Yi Quan Lee¹, Wanqi Xue¹, Tingjia Shen², Kenan Song¹, Kefan Wang², Wenjia Xie², Yuyang Ye², Huifeng Guo¹, Yong Liu^{1*}, Defu Lian^{2*}, Ruiming Tang^{1*}, Enhong Chen^{2*}

¹Huawei Noah's Ark Lab

{guowei67, chin.jin.yao, liu.yong6, tangruiming}@huawei.com

²University of Science and Technology of China
{wanghao3, liandefu, cheneh}@ustc.edu.cn
{zhanglk5}@mail.ustc.edu.cn

ABSTRACT

Recommendation systems are essential for filtering data and retrieving relevant information across various applications. Recent advancements have seen these systems incorporate increasingly large embedding tables, scaling up to tens of terabytes for industrial use. However, the expansion of network parameters in traditional recommendation models has plateaued at tens of millions, limiting further benefits from increased embedding parameters. Inspired by the success of large language models (LLMs), a new approach has emerged that scales network parameters using innovative structures, enabling continued performance improvements. A significant development in this area is Meta's generative recommendation model HSTU, which illustrates the scaling laws of recommendation systems by expanding parameters to thousands of billions. This new paradigm has achieved substantial performance gains in online experiments. In this paper, we aim to enhance the understanding of scaling laws by conducting comprehensive evaluations of large recommendation models. Firstly, we investigate the scaling laws across different backbone architectures of the large recommendation models. Secondly, we conduct comprehensive ablation studies to explore the origins of these scaling laws. We then further assess the performance of HSTU, as the representative of large recommendation models, on complex user behavior modeling tasks to evaluate its applicability. Notably, we also analyze its effectiveness in ranking tasks for the first time. Finally, we offer insights into future directions for large recommendation models. Supplementary materials for our research are available on GitHub at https: //github.com/USTC-StarTeam/Large-Recommendation-Models.

KEYWORDS

Scaling Law, Large Recommendation Model, Generative Recommendation

1 INTRODUCTION

In the current era characterized by an overwhelming influx of information, web services such as TikTok, Taobao, and YouTube are inundated with vast amounts of data. To effectively navigate this deluge and recommend relevant items to users, real-world recommendation systems must efficiently identify items from a pool of millions to billions of candidates. This is typically achieved through a multi-stage framework, which involves sequential processes of recall, ranking, and re-ranking.

In the ongoing quest for improved user experiences and increased platform revenues, the scalability of models within industrial recommendation systems has become a focal point of research and development. Early approaches sought to enhance scalability by expanding the sparse parameters, such as embedding tables, through the integration of additional categorical and cross features. This expansion can lead to models with billions or even trillions [6, 132] of features, resulting in embedding parameters that require hundreds of gigabytes to terabytes of storage in large-scale applications. An alternative method involves increasing the embedding dimension, as demonstrated by multi-embedding models [24], which aim to address embedding collapse and enhance scalability. However, recent research [126] indicates that simply enlarging embedding tables does not effectively improve model capacity and is computationally inefficient. Consequently, it is imperative to explore new perspectives for addressing scalability challenges.

Drawing inspiration from the remarkable success of large language models (LLMs), recent research has increasingly focused on scaling up dense parameters in recommendation systems by developing innovative structures that enable sustained performance growth through layer stacking. From the perspective of feature interaction modeling, Wukong [128] investigates the scaling law by refining the feature interaction module. This is achieved through an effective network architecture that combines stacked factorization machines with linear blocks, aiming to facilitate loss scaling as dense parameters increase. However, while the authors assert the presence of a scaling law, the results show only modest reductions in loss curves, and the improvements on some public datasets are relatively minor, limiting the work's impact and attention. Conversely, in another perspective of generative recommendation (GR) using user behavior sequences as input, HSTU [126] introduces an innovative transformer-based structure. This model replaces the Softmax function with the SiLU activation function and incorporates additional multiplicative terms within the self-attention module. It demonstrates empirical scalability in line with a power-law relationship concerning training compute, spanning three orders of magnitude and achieving a scale comparable to large language models like GPT-3 and LLaMa-2.

While recent research has extensively explored the use of large language models (LLMs), such as ChatGPT, to enhance recommendations (referred to as *LLMs enhanced recommendation*) through their world knowledge and advanced logical reasoning capabilities, the challenge of scaling up the dense parameters of recommendation models (referred to as *large recommendation models*) remains underexplored.

 $[\]dagger$ Equal contribution.

^{*} Corresponding authors.

Definition 1.1 (Large Recommendation Model). A large recommendation model is a scalable system designed to process and analyze multi-modal and heterogeneous data. It supports a wide range of recommendation tasks and enhances performance by leveraging increased model parameters and larger datasets.

As we stand at the intersection of current and next-generation recommendation technologies, our focus shifts towards the relatively unexplored domain of large recommendation models across various tasks and the underlying principles of their scaling laws. Given the significant business improvements and powerful scaling effects demonstrated by the GR paradigm, this paper presents a comprehensive analysis of the scalability factors in current large recommendation models. Specifically, we aim to deepen the understanding of the scaling law of large recommendation models by evaluating their capacities in complex behavior modeling and ranking tasks, thereby uncovering their potential in a wider range of downstream applications.

In summary, our contributions are fourfold, specifically emphasizing the most critical and emerging characteristics of scalable large recommendation models:

- We analyze the scalability of various popular transformer-based architectures for large recommendation models, including HSTU, Llama, GPT, and SASRec, by evaluating their performance with an increasing number of attention blocks.
- We conduct comprehensive ablation studies and parameter analysis on HSTU, as the representative of large recommendation models, to explore the origins of its scaling law. Additionally, we enhance the scalability of SASRec, a legacy transformer-based sequential recommendation model, by integrating effective modules from scalable large recommendation models.
- We further investigate the performance of HSTU on complex user behavioral sequence data, identifying areas for improvement in modeling intricate user behaviors, particularly with data involving side information and multiple behaviors.
- To the best of our knowledge, we are the first to thoroughly evaluate HSTU on ranking tasks, demonstrating their scalability in this context. Our evaluations also provide insights into designing effective large recommendation models for ranking, considering datasets and hyper-parameters.

This paper is organized as follows: Section 2 provides a comprehensive review of the shift in user behavior modeling paradigms, particularly within large recommendation models, highlighting key changes and trends, as illustrated in Figure 1. Section 3 presents the latest advancements in the related field. In Section 4, we define the problem under investigation. Preliminary experimental results related to the research problems are discussed in Section 5. Section 6 explores prospects for future research by proposing potential directions. Finally, Section 7 summarizes our key findings and contributions. The overall framework is shown in Figure 2.

2 SHIFT OF MODELING PARADIGM

With the emergence of *large recommendation models*, the focus of recommendation systems is undergoing significant transformations. As shown in the Figure 1, a pivotal shift is the reduced emphasis on feature engineering and model architecture design.

Initially, the development of recommendation systems is heavily influenced by constraints in computational resources, prompting researchers to concentrate on crafting effective features and utilizing simple predictive models [23, 131], as depicted in Figure 1(A).

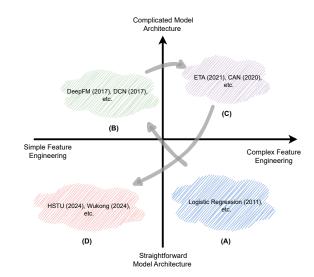


Figure 1: Illustration of the shift in modeling paradigms.

A prime example of this approach is Logistic Regression (LR) [63], which assigns adaptive weights to various features to improve prediction accuracy. These features typically include user attributes (such as country, gender, and age), item characteristics (such as brand and category), and contextual elements (including weather, hour of the day, and day of the week).

Later, with the rise of deep learning, there has been a significant shift towards developing more complex models that fully leverage the parallel computing capabilities of GPUs, as depicted in Figure 1(B). One notable example is DeepFM [20], which introduces a hybrid architecture combining a shallow FM model and a deep DNN model to simultaneously learn low-order and high-order feature interactions. Similarly, the Deep & Cross Network (DCN) [101] explicitly applies feature crossing at each layer, allowing feature interaction orders to increase in a layer-wise manner.

While the performance improvements from simply designing more complex deep models have reached a plateau, a recent trend in recommendation systems is to revisit feature engineering through learnable methods, as illustrated in Figure 1(C). For instance, the ETA model [8] leverages locality-sensitive hashing and Hamming distance to compute item similarity, allowing for the selection of the most significant features from long sequences to improve recommendation accuracy. Similarly, CAN [135] employs a meta-network to approximate the Cartesian product, effectively modeling the cross-relations between two features.

In the current era, the extraordinary success of large recommendation models, combined with the widespread acceptance of scaling laws, indicates that computational power will be crucial for future improvements in model performance, as depicted in Figure 1(D). The scaling law, which demonstrates a power-law relationship between model loss and key variables such as model size, dataset size, and computational resources, shapes our vision for the evolution of large-scale recommendation model development. Looking ahead, we anticipate two important directions: firstly, expanding the dataset by more effectively mining and leveraging user behavior sequences across various domains for user life-cycle modeling; and secondly, scaling up the model size while ensuring training and inference under certain cost constraints with efficiency optimization.

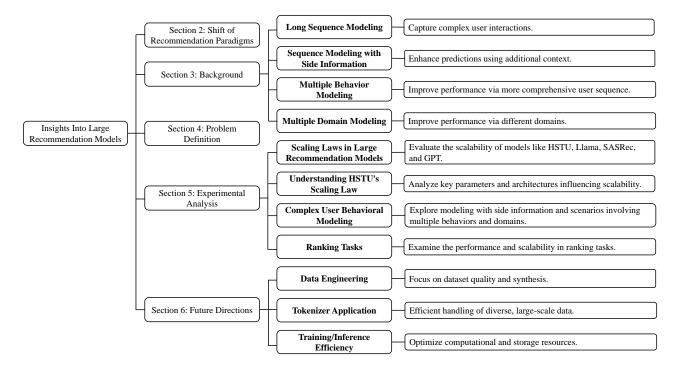


Figure 2: The figure outlines the structure of the paper, starting with the shift in user behavior modeling paradigms discussed in Section 2, followed by advancements in the field in Section 3. It then covers the problem definition in Section 4, presents preliminary experimental results in Section 5, and concludes with future research directions in Section 6.

3 BACKGROUND

Large recommendation models based on the GR paradigm can be viewed as sophisticated user behavior models. These models process user behavior sequences as input and learn behavior dependencies through transductive learning. In this section, we review key research directions in user behavior modeling to provide a comprehensive understanding of this field [30, 58, 107].

Long Sequence Modeling. The rapid development of online services has led to the accumulation of extensive user behavioral data. Consequently, modeling long sequences of user behavior is crucial for industrial recommendation systems. This focus has driven significant efforts in the long sequence modeling to effectively capture the multifaceted and evolving interests of users. Research in this area has evolved through three phases: the initial use of memory networks [10, 72, 81], followed by advancements in user behavior recall techniques [73, 74, 111–113], and most recently, the emergence of efficient transformer models [69, 108, 123].

Sequence Modeling with Side Information. Beyond the primary user behavior sequence, which includes the user's historical interactions, side information such as temporal data and user/item attributes can enhance sequence modeling. For temporal information, the simplest and most frequently used approach is to arrange the user's historical interactions in chronological order [26, 41, 87, 96, 120]. Beyond ordering, the time intervals between interactions provide insights into user preferences [13, 53, 95, 98, 117, 126]. Regarding user/item attributes, these can offer additional context to the user behavior modeling process. User features, such as age, gender, and occupation, and item features, such as price, category, and brand, can serve as complementary auxiliary information. However,

privacy concerns and regulations like GDPR may limit access to user features, leading models to primarily focus on item attributes.

Multiple Behavior Modeling. Traditional recommendation systems often focus on a single type of user behavior. However, realworld user interactions are multifaceted, including actions such as clicks, shares, and purchases. Leveraging this multi-behavioral data is essential for constructing a comprehensive user representation, thereby enabling more precise recommendations. This challenge has led to the emergence of the Multi-Behavior Sequential Recommendation (MBSR) problem [22, 25, 64, 65, 94]. Current MBSR methodologies can be broadly categorized into two main approaches: (1) segmenting item sequences into subsequences according to behavior categories, modeling each subsequence independently, and subsequently integrating them for prediction [18, 21, 109, 110]; and (2) modeling the entire item sequence while incorporating behavior types as auxiliary inputs [15, 59, 78, 86, 124, 126].

Multiple Domain Modeling. In addition to integrating interactions from various user behaviors, several methods leverage auxiliary domain interactions to enrich user behavior profiles, thereby improving recommendation precision in the target domain. This strategy is commonly known as cross-domain or multi-domain sequential recommendation [4, 51, 60, 85, 91, 118, 130]. Recent advancements [33, 34, 52, 99, 118, 136] have incorporated auxiliary information, such as product descriptions, titles, and brands, which act as semantic bridges across domains. These approaches typically follow a two-stage framework: initially, they employ pre-training tasks to develop enhanced universal representations using the auxiliary information; subsequently, they fine-tune the model within a single domain to enable effective adaptation to new scenarios.

4 PROBLEM DEFINITION

In this study, we aim to present a comprehensive analysis of the scalability sources of current large recommendation models and reveal the capacity of current large recommendation models in complex behavior modeling and ranking tasks. We define two primary sets: the user set $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ and the item set $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$, where $|\mathcal{U}|$ and $|\mathcal{V}|$ represent the total number of users and items, respectively.

For each user $u \in \mathcal{U}$, we model their interactions as a sequence $X_u = \{x_1, x_2, \dots, x_n\}$, where each $x_i \in \mathcal{V}$ is an item the user has interacted with, listed in chronological order. The sequence length is capped at n; shorter sequences are padded, and longer sequences are truncated to retain the most recent interactions. Our goal is to train a model that predicts the next item x_{n+1} a user will interact with, based solely on the interaction sequence X_u .

For simplicity, we divide the multi-stage recommendation process into recall and ranking, omitting other possible steps like pre-ranking and re-ranking, which we will explore in future work.

Recall: In single-behavior recall, given a user's historical interaction sequence X_u , the recall model is trained to select items from the entire item set \mathcal{V} that the user might interact with at step n+1, forming a candidate set I.

Ranking: For the items in the candidate set I identified by the recall model, the ranking model evaluates them based on the user's historical interaction sequence X_u . It predicts the user's preference for each item, such as the probability of a click, and sorts the items to determine the most relevant ones to display to the user.

To further improve recommendation performance, we incorporate side information into the generative framework.

Recommendation with Side Information: Let C denote the set of side information attributes, such as time, location, and user age. For each user $u \in \mathcal{U}$, the interaction sequence is represented as $X_u = \{(x_1, c_1), (x_2, c_2), \dots, (x_n, c_n)\}$, where $x_i \in \mathcal{V}$ is the item interacted with at the i-th step, and $c_i \in C$ is the relevant side information available at that step. The goal is to train a model that leverages both the interaction sequence X_u and the associated side information to predict the next item x_{n+1} that the user will interact with in the subsequent time step.

Additionally, we address complex scenarios by exploring multibehavior and multi-domain recommendations to capture intricate user interactions and cross-domain influences.

Multi-behavior Recommendation: Let $\mathcal{B} = \{b_1, b_2, \dots, b_{|\mathcal{B}|}\}$ denote the set of different user behaviors. In multi-behavior recommendation, the interaction sequence is constructed by pairing each user interaction with its corresponding behavior, represented as $X_u = \{(x_1, b_1), (x_2, b_2), \dots, (x_n, b_n)\}$. Here, x_i denotes the item, and b_i represents the behavior associated with that item. The model's objective is to predict the next item x_{n+1} based on X_u .

Multi-domain Recommendation: In multi-domain recommendation, we consider multiple domains denoted as D^i , where $i=1,2,\ldots,d$ and d represents the total number of domains. Each domain D^i comprises a user set \mathcal{U}^i , an item set \mathcal{V}^i , and an interaction set $X^i_{\mathcal{U}}$. The objective is to utilize the combined cross-domain interaction sequences $X=X^1_{\mathcal{U}}\cup X^2_{\mathcal{U}}\cup \ldots\cup X^d_{\mathcal{U}}$ to predict the next item $x^i_{n+1}\in \mathcal{V}^i$ that a user will interact with in a specific domain i.

By defining these problems, we establish a comprehensive framework to address the entire process of recall and ranking in recommendation systems. This framework incorporates side information and extends the scope from single-behavior to complex

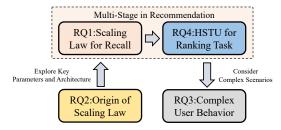


Figure 3: The research questions addressed in experiments.

multi-domain and multi-behavior scenarios, thereby enhancing the adaptability and accuracy of recommendation models.

5 EXPERIMENTS

In this section, we conduct extensive experiments on multiple datasets to address the following research questions.

- (RQ1) How does model depth influence the scaling laws of large recommendation models?
- (RQ2) Where does the scaling law of large recommendation models originate from?
- (RQ3) What is the performance of HSTU when applied to complex user behavioral sequence modeling?
- (RQ4) What is the performance of HSTU on ranking task? What are the key points to obtain the scaling law?

As depicted in Figure 3, we first introduce the experimental settings in Section 5.1. Then, we present an overview of model performances and the scalability of some popular backbones in Section 5.2 to answer RQ1. As the representative of large recommendation models, we next conduct an in-depth analysis of HSTU and address RQ2 in Section 5.3. Besides, we also conduct comprehensive evaluations of HSTU on complex user behavioral sequence modeling and ranking tasks in Sections 5.4 and 5.5, respectively, to address RQ3 and RQ4.

5.1 Experimental Settings

5.1.1 Datasets. To assess the effectiveness of the recommendation models, we conduct experiments using both single-behavior and multi-behavior datasets. For the single-behavior analysis, we utilize four sets of widely recognized public datasets, as follows.

- MovieLens-1M (ML-1M) & MovieLens-20M (ML-20M) [27]:
 These benchmark datasets in recommendation systems research contain 1 million ratings from 6,000 users on 4,000 movies (ML-1M) and 20 million ratings from 138,000 users on 27,000 movies (ML-20M). Both datasets provide user ratings from 1 to 5, along with demographic information and movie metadata.
- Amazon Books (AMZ-Books) [62]: As a subset of the Amazon review dataset, AMZ-Books contains user reviews and ratings for books on Amazon. It offers different types of explicit feedback (e.g., ratings from 1 to 5 as well as textual reviews), making it valuable in recommendation system research.
- KuaiRand-27k [16]: A dataset from a popular short video platform that provides extensive user and content interaction data, including watching, liking, and commenting behaviors. It includes detailed user and content features such as demographics,

N 11	" D1 1		ML-1M			ML-20M			AMZ-Books	
Model	# Blocks	HR@10	NDCG@10	MRR	HR@10	NDCG@10	MRR	HR@10	NDCG@10	MRR
	2	0.2923	0.1628	0.1395	0.2915	0.1642	0.1409	0.0564	0.0308	0.0279
	4	0.3115	0.1778	0.1529	0.3218	0.1849	0.1582	0.0617	0.0338	0.0305
HSTU	8	0.3299	0.1857	0.1572	0.3403	0.1990	0.1709	0.0649	0.0357	0.0322
	16	0.3322	0.1887	0.1601	0.3520	0.2079	0.1787	0.0680	0.0377	0.0340
	32	0.3298	0.1863	0.1580	0.3569	0.2113	0.1814	0.0584	0.0325	0.0295
	2	0.3029	0.1697	0.1450	0.3044	0.1724	0.1475	0.0510	0.0275	0.0252
Llama	4	0.3153	0.1796	0.1539	0.3277	0.1887	0.1615	0.0537	0.0292	0.0266
	8	0.3232	0.1848	0.1583	0.3449	0.2008	0.1718	0.0547	0.0296	0.0269
	16	0.3298	0.1872	0.1594	0.3495	0.2055	0.1764	0.0227	0.0117	0.0112
	32	0.3299	0.1896	0.1626	0.3551	0.2090	0.1791	0.0210	0.0110	0.0107
	2	0.2798	0.1564	0.1343	0.2419	0.1333	0.1155	0.0568	0.0307	0.0279
	4	0.2803	0.1543	0.1319	0.0284	0.0148	0.0162	0.0356	0.0191	0.0180
GPT	8	0.0353	0.0162	0.0178	0.0302	0.0147	0.0151	0.0049	0.0026	0.0032
	16	0.0270	0.0133	0.0162	0.0264	0.0127	0.0138	0.0050	0.0026	0.0032
	32	0.0247	0.0115	0.0140	0.0312	0.0145	0.0150	0.0058	0.0029	0.0033
	2	0.2824	0.1594	0.1375	0.2781	0.1553	0.1330	0.0561	0.0305	0.0276
	4	0.2744	0.1543	0.1335	0.0599	0.0294	0.0284	0.0544	0.0300	0.0272
SASRec	8	0.2183	0.1186	0.1030	0.0326	0.0156	0.0169	0.0084	0.0042	0.0043
	16	0.0431	0.0184	0.0176	0.0349	0.0167	0.0177	0.0095	0.0044	0.0042
	32	0.0366	0.0181	0.0195	0.0301	0.0159	0.0169	0.0084	0.0044	0.0045

Table 1: Performance comparison of different backbones.

preferences, metadata, and engagement metrics, crucial for developing and evaluating recommendation algorithms.

To evaluate the model's performance in a multi-behavior context, we also employ two prominent multi-behavior datasets:

- CIKM 2019 EComm AI Dataset (CIKM): This dataset includes user behavior logs, product information, and interactions like clicks, add-to-cart actions, purchases, and likes. Product attributes such as category, brand, and price are also provided, along with user demographics like age, gender, and location.
- IJCAI-15 Repeat Buyers Prediction Dataset (IJCAI) [90]: Sourced from real-world scenarios, this dataset includes user behavior records (browsing, clicking, purchasing, favoriting), product details (ID, category, brand), and basic user profiles (ID, registration info). It offers multi-dimensional information for exploring user behavior patterns.

To evaluate the performance in multi-domain recommendation, we conduct further experiments on a multi-domain dataset:

- Amazon Multiple Domains (AMZ-MD) [62]: Similar to AMZ-Books dataset, this dataset is selected from Amazon review dataset. For multi-domain recommendation, the interactions include users and items from four different domains: Digital Music, Movies & TV, Toys, and Video Games.
- 5.1.2 Dataset Preprocessing. For single behavior evaluations, we use the MovieLens and Amazon Books datasets to align with evaluation in [126]. For multi-behavior evaluations, we use the CIKM and IJCAI datasets from real industrial scenarios. The datasets are divided into training and test sets. In the training set, we use the whole sequence except the last two items to train and predict the second-to-last item. In the test set, we predict the last item. We

follow the data preprocessing steps in [126]. For the multi-domain evaluations, we utilize the AMZ-MD dataset, applying a 5-core filter to exclude less popular users and items.

5.1.3 Evaluation Metrics. For the recall task, we adopt three widely used evaluation protocols: Hit Ratio (HR), Normalized Discounted Cumulative Gain (NDCG), and Mean Reciprocal Rank (MRR). HR@K is applied to measure whether the test item is under the top-K list of the recommendation results. NDCG@K evaluates the top-K recommendation quality by giving higher scores to the top-ranked relevant items. MRR calculates the rank of the first relevant item presented in the recommendation results.

For the ranking task, we adopt the two most widely-used evaluation metrics: AUC and Logloss for evaluation. AUC measures the probability of predicting higher scores of positive interactions than negative interactions. A higher value indicates a better performance. Logloss calculates the distance between the true labels and the predicted scores. A lower value indicates a better performance.

5.1.4 Parameter Settings. All experiments are implemented using PyTorch [70] on servers equipped with 8× Huawei D910B NPUs, each with 32GB of memory. We utilize the Accelerate framework to facilitate large-scale distributed model training. To ensure a fair comparison, we maintain the original model implementations' hyper-parameters, except for those specifically being explored.

5.2 Comparison of Model Performances (RQ1)

We aim to investigate the impact of model depth on scaling laws by evaluating several popular transformer-based large recommendation model architectures for recall tasks in recommendation systems, including HSTU [126], Llama [92], GPT [1], and SASRec [41]. Our objective is to determine whether increasing the number of parameters by varying the number of attention blocks results in

Model Variant	# Blocks	HR@10	NDCG@10
	2	0.2752	0.1544
	4	0.2944	0.1674
HSTU	8	0.3083	0.1756
(w/o r.a.b.)	16	0.3135	0.1801
	32	0.3149	0.1805
	2	0.2807	0.1581
	4	0.3098	0.1761
HSTU	8	0.3298	0.1897
(w/o SiLU)	16	0.3422	0.1992
	32	0.3476	0.2043
	2	0.2710	0.1507
	4	0.2900	0.1623
HSTU	8	0.3154	0.1800
(w/o feature interaction)	16	0.3300	0.1905
	32	0.3339	0.1947

Table 2: Impact of various HSTU components on scaling law.

performance improvements and to explore further how the architecture of these backbones influences the scaling laws.

The results are detailed in Table 1. In our experiments, we observe that when the number of transformer blocks is low, i.e., the model has fewer parameters, the performance of the four backbones is similar across different datasets, with the best-performing architecture varying by dataset. For instance, Llama performs best on the ML-1M dataset, while GPT outperforms Llama on the AMZ-Books dataset with two blocks. When we increase the number of blocks to expand model parameters, HSTU and Llama demonstrate better scalability, while GPT and SASRec show no scalability. Though GPT architecture generally performs well and adheres to scaling laws in broader applications, it exhibits limited scalability in recommendation tasks. This could be due to the lack of architectural adaptations specific to recommendation features. Additionally, as shown in Table 1, model performance varies with both dataset size and model parameter size, even when the architecture remains constant. To investigate this phenomenon further, we will conduct additional studies in Section 5.3.

5.3 Understanding the Scaling Law of HSTU (RQ2)

Next, we investigate the origins of the scaling law in HSTU, as the representative of large recommendation models, by performing an in-depth analysis of its components. We begin with ablation studies to evaluate the impact of each key component on recommendation performance and the model's scalability. Following this, we conduct parameter analyses to assess HSTU's scalability across various hyperparameter settings. We then explore the potential to introduce a scaling law to the SASRec model by implementing specific modules informed by our ablation study findings. Finally, we examine the characteristics of HSTU that contribute to its scaling law through visualization analysis. All experiments in this section utilize the ML-20M dataset, the largest dataset in our research, with similar trends observed across other datasets.

5.3.1 Ablation Studies. We investigate three key components of HSTU: the selection of relative attention bias, the use of SiLU for attention score weighting, and the method of feature interaction.

Relative Attention Bias Type	HR@10	NDCG@10
Rel. Position and Time Diff. Bucket	0.3376	0.1967
Rel. Time Diff. Bucket Only	0.3356	0.1952
Rel. Position Only	0.3122	0.1787
RoPE	0.3149	0.1801
No Attention Bias	0.3083	0.1756

Table 3: Ablation study on the impact of different choices of relative attention bias.

Attention Score Function	HR@10	NDCG@10
SiLU	0.3376	0.1967
Softmax	0.3298	0.1897

Table 4: Ablation study on the impact of different attention score functions.

Feature Interaction	HR@10	NDCG@10
w/ feature interaction	0.3376	0.1967
w/o feature interaction	0.3154	0.1800

Table 5: Ablation study on the role of feature interaction.

Impact of Components on Scaling Law. To analyze the impact of various components of HSTU on the scaling law, we conduct an ablation study by systematically removing one key component at a time: relative attention bias (r.a.b.), the SiLU activation function, or feature interaction. We then evaluate how the model's performance varied with an increasing number of HSTU blocks. The results, presented in Table 2, show that most models maintain scalability even with the removal of a key component. However, the improvement in performance metrics, such as NDCG and HR, plateau most significantly when the relative attention bias is removed, particularly as the number of HSTU blocks increased from 8 to 32. This suggests that HSTU is robust, as removing a single component does not significantly impact its scalability with model depth. The qualitative results of further analyses for each individual component will be presented in the remainder of this ablation study.

Relative Attention Bias. Early transformer models utilize positional embeddings to incorporate positional information between tokens effectively. However, instead of using absolute positional information, HSTU employs relative position and time difference buckets to modify attention scores. Specifically, the attention score between two items in a sequence is adjusted by a bias that depends on (1) their relative positions and (2) their time difference buckets.

We conduct a series of experiments to evaluate the impact of different attention bias mechanisms on the performance of the HSTU model. Specifically, we replace the relative attention bias module in HSTU with three alternative mechanisms: (1) Relative Attention Bias using only Bucketed Relative Time Difference (Rel. Time Diff. Bucket Only), (2) Relative Attention Bias using only Relative Position (Rel. Position Only), and (3) Rotary Positional Encoding (RoPE). The performance results of these experiments are presented in Table 3.

Following our intuition, temporal information emerges as the most critical factor in the sequential recommendation setting, surpassing the reliance on positional information typical in language models. Our results show that models using relative attention bias with relative time information significantly outperform those using only positional information, as measured by HR and NDCG metrics.

		NDCG@10	NDCG@50	HR@10	HR@50	MRR	NDCG@10	NDCG@50	HR@10	HR@50	MRR
# Dim	# Blocks			S = 100					S = 200		
	12	0.1585	0.2153	0.2830	0.5407	0.1356	0.1741	0.2325	0.3083	0.5729	0.1485
	24	0.1663	0.2234	0.2937	0.5523	0.1425	0.1824	0.2408	0.3184	0.5826	0.1561
50	32	0.1682	0.2253	0.2951	0.5539	0.1445	0.1867	0.2445	0.3238	0.5856	0.1599
	64	0.1684	0.2244	0.2950	0.5489	0.1445	0.1891	0.2470	0.3262	0.5881	0.1622
	96	0.1680	0.2243	0.2953	0.5473	0.1447	0.1901	0.2471	0.3277	0.5854	0.1629
	12	0.1770	0.2337	0.3078	0.5651	0.1519	0.1952	0.2536	0.3357	0.6000	0.1674
	24	0.1805	0.2368	0.3132	0.5684	0.1547	0.1974	0.2604	0.3423	0.6022	0.1704
100	32	0.1815	0.2375	0.3133	0.5669	0.1559	0.2047	0.2619	0.3493	0.6078	0.1753
	64	0.1801	0.2353	0.3118	0.5617	0.1543	0.2053	0.2619	0.3488	0.6048	0.1760
	96	0.1746	0.2296	0.3030	0.5525	0.1498	0.2032	0.2602	0.3450	0.6027	0.1746
	12	0.1799	0.2362	0.3102	0.5649	0.1548	0.2019	0.2600	0.3441	0.6070	0.1734
	24	0.1849	0.2405	0.3171	0.5686	0.1591	0.2102	0.2673	0.3560	0.6139	0.1803
200	32	0.1812	0.2365	0.3138	0.5641	0.1551	0.2120	0.2689	0.3583	0.6154	0.1819
	64	0.1752	0.2308	0.3053	0.5575	0.1500	0.2100	0.2667	0.3537	0.6098	0.1807
	96	0.1789	0.2335	0.3082	0.5555	0.1538	0.2089	0.2654	0.3532	0.6081	0.1795
	12	0.1768	0.2327	0.3059	0.5588	0.1521	0.2037	0.2610	0.3467	0.6053	0.1748
	24	0.1767	0.2317	0.3045	0.5535	0.1521	0.2119	0.2685	0.3580	0.6132	0.1819
400	32	0.1814	0.2359	0.3112	0.5577	0.1560	0.2105	0.2672	0.3555	0.6113	0.1808
	64	0.1697	0.2229	0.2935	0.5350	0.1459	0.2102	0.2670	0.3541	0.6105	0.1808
	96	0.1627	0.2205	0.2883	0.5369	0.1387	0.2033	0.2593	0.3445	0.5972	0.1747

Table 6: Parameter analyses on the ML-20M dataset with varying sequence length (|S|), embedding dimension (# Dim), and number of HSTU blocks (# Blocks).

# Dim	# Blocks	# Heads	HR@10	NDCG@10
32			0.2186	0.1156
64	8	o	0.2736	0.1514
128	٥	8	0.2357	0.1770
256			0.3376	0.1967
	2		0.2928	0.1653
256	4	8	0.3194	0.1840
236	8	٥	0.3376	0.1967
	16		0.3469	0.2037
		1	0.3385	0.1978
		2	0.3380	0.1968
256	0	4	0.3381	0.1978
230	8	8	0.3376	0.1967
		16	0.3380	0.1978
		32	0.3384	0.1975

Table 7: Parameter analyses of HSTU on the ML-20M dataset.

Furthermore, removing positional information from the relative attention bias does not significantly affect recall metrics. Interestingly, when using only relative positional information, RoPE enhances attention scores more effectively than relative attention bias.

SiLU for Attention Score Weighting. A key innovation of HSTU, compared to other transformer-based language models, is its use of the SiLU activation function instead of softmax for calculating attention weights. We analyze how different activation functions affect attention score weighting. Specifically, we modify the spatial aggregation in the standard implementation of HSTU blocks using the following equation:

$$Attn(X)V(X) = Softmax\left(\frac{Q(X)K(X)^{T}}{\sqrt{n}} + rab^{p,t}\right)V(X), \quad (1)$$

where X represents the input hidden states to the network. In our ablation study, we replace the original SiLU activation function with

the Softmax function. The denominator \sqrt{n} serves as the standard normalization factor in scaled dot-product attention. As shown in Table 4, we observe that using the Softmax function decreases the performance of HSTU. This decline is likely because Softmax aggregation bounds the maximum attention score value, making it less expressive and therefore less suitable for recommendation tasks compared to SiLU when aggregating attention scores.

Feature Interaction. As shown in Table 5, we investigate the impact of feature-interaction mechanisms. These mechanisms are a common characteristic of deep-learning recommendation models, as demonstrated in works like DeepFM [20] and DCN [101]. These mechanisms typically involve feature-crossing, achieved through operations such as the dot product or Hadamard product between dense representations of feature pairs or dimensions within the dense representation itself. In HSTU, feature interaction is implemented via a point-wise transformation layer. To evaluate the impact of this layer, we assess HSTU's performance by removing the Hadamard product and layer normalization, modifying the pointwise transformation layer according to the following equation:

$$Y(X) = f_2(Attn(X)V(X)), \tag{2}$$

where f_2 represents an MLP. We remove the layer normalization applied to Attn(X)V(X) to enhance model stability, as another layer normalization is subsequently applied to Y(X). This adjustment prevents the application of two consecutive layer normalizations without an intervening non-linear transformation. The results in Table 5 indicate that removing the Pointwise Transformation Layer significantly reduces the performance of HSTU.

5.3.2 Parameter Analyses. In this study, we examine scaling laws by varying embedding dimensions, the number of transformer heads, model depth, and sequence length using the ML-20M dataset. We begin by analyzing two crucial parameters: embedding dimension and the number of model layers. Table 6 illustrates the effects of adjusting the maximum sequence length for the HSTU model on

Model	# Blocks	HR@10	NDCG@10	MRR
	2	0.2781	0.1553	0.1330
	4	0.0599	0.0294	0.0284
SASRec	8	0.0326	0.0156	0.0169
	16	0.0349	0.0167	0.0177
	32	0.0301	0.0159	0.0169
	2	0.0478	0.0269	0.0258
SASRec	4	0.0447	0.0227	0.0222
(w/ r.a.b.)	8	0.0398	0.0213	0.0221
(w/ 1.a.b.)	16	0.0454	0.0228	0.0222
	32	0.0460	0.0232	0.0224
	2	0.2788	0.1553	0.1330
SASRec	4	0.2551	0.1398	0.1199
(residual in HSTU)	8	0.2059	0.1085	0.0939
(residual in 11010)	16	0.0628	0.0297	0.0296
	32	0.0589	0.0282	0.0282
	2	0.2850	0.1592	0.1359
SASRec	4	0.2575	0.1422	0.1223
(residual in Llama)	8	0.1380	0.0699	0.0633
(Tesiddai iii Elailia)	16	0.1242	0.0617	0.0562
	32	0.1234	0.0609	0.0552
	2	0.2990	0.1689	0.1445
SASRec	4	0.3116	0.1799	0.1513
(residual in Llama,	8	0.3152	0.1809	0.1571
w/ r.a.b.)	16	0.3140	0.1796	0.1538
	32	0.3182	0.1835	0.1575

Table 8: Analyses on the scaling law of SASRec on the ML-20M dataset.

the ML-20M dataset. From this table, we derive four principal conclusions: (1) Increasing sequence length does not necessarily benefit the performance, as longer sequences may introduce more noise. (2) Performance does not consistently improve with increasing model size. Upon reaching a peak, performance begins to fluctuate, highlighting the need for guidelines to align model size with dataset size for optimal performance. (3) The optimal number of layers decreases as the embedding size increases, while the product of the optimal number of layers (L) and embedding dimension (D) remains constant, supporting our theoretical model that size is proportional to O(LD). (4) As the maximum sequence length increases, the optimal model size O(LD) also increases, indicating that larger datasets should be matched with larger models.

We also conduct an investigation into the interplay among embedding dimensions, model layers, and attention heads. The findings related to attention heads are detailed in Table 7. Our analysis indicates that increasing the number of HSTU blocks and the embedding dimension significantly enhances recall performance. In contrast, variations in the number of attention heads generally do not substantially affect recall performance.

5.3.3 Scaling Law in SASRec. To further investigate the source of the scaling law, we examine whether the success of HSTU and Llama can be replicated by modifying the self-attention layers in traditional backbone models like SASRec. Specifically, we introduce the relative attention bias (r.a.b.) from HSTU into the standard self-attention layers of SASRec by incorporating it into the $Q(X)K(X)^T$ computation during attention processing. Additionally, we adapt

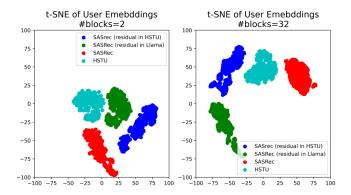


Figure 4: Visualization of user embeddings on the ML-20M dataset.

the residual connection using the patterns implemented in HSTU and Llama, as described below:

HSTU
$$\begin{cases} g_1(X) = SA(LN(X)), \\ g_2(X) = FFN(LN(X)), \\ B_l(X) = X + g_2(g_1(X)), \end{cases}$$
(3)

Llama
$$\begin{cases} g_1(X) = X + SA(LN(X)), \\ g_2(X) = X + FFN(LN(X)), \\ B_l(X) = g_2(g_1(X)), \end{cases}$$
 (4)

where $B_l(X)$ denotes the output of the l-th attention block, given the hidden state X from the (l-1)-th block. The components of the attention blocks include $LN(\cdot)$ for layer normalization, $SA(\cdot)$ for self-attention layers, and $FFN(\cdot)$ for feed-forward networks. A key distinction in our approach is the implementation of the residual connection *before* the layer normalization operation, in contrast to the SASRec framework, which applies it *after* layer normalization. The results of our experiments using the ML-20M dataset are presented in Table 8.

Our experiments reveal that increasing the number of transformer blocks degrades the performance of SASRec on the ML-20M dataset, thereby highlighting the poor scalability of the original SASRec model. This negative trend persists regardless of whether we incorporate the relative attention bias (r.a.b.) module or modify the residual connection as described in Equations 3 and 4. Notably, the residual connection pattern used in Llama exhibits greater robustness to an increasing number of blocks. Consequently, we apply the r.a.b. module to the residual connections in Llama, which demonstrates improved scalability on the ML-20M dataset. These findings suggest that both the residual connection pattern and the relative attention bias contribute to enhancing the scalability of traditional recommendation models. These preliminary evaluations provide valuable insights for future research on the scaling laws of recommendation systems.

5.3.4 Visualization Analysis. Previous analyses have indicated that components such as residual connections and relative attention bias can influence the scalability of recommendation models. Additionally, factors like dimension size and sequence length affect recall performance. However, the specific model characteristics that enhance scalability remain unclear. To investigate the intrinsic factors influencing model scalability, we visualize user embeddings

Dataset	Side Info.	# Blocks	HR@10	HR@50	NDCG@10	NDCG@50	MRR
		2	0.2962	0.5757	0.1662	0.2283	0.1427
		4	0.3206	0.5856	0.1791	0.2379	0.1513
	w/o	8	0.3312	0.5991	0.1871	0.2468	0.1588
		16	0.3333	0.5973	0.1881	0.2468	0.1590
ML-1M		32	0.3289	0.5946	0.1842	0.2431	0.1554
1112 1111		2	0.3011	0.5768	0.1693	0.2305	0.1450
		4	0.3177	0.5908	0.1795	0.2402	0.1531
	$\mathbf{w}/$	8	0.3236	0.5998	0.1870	0.2485	0.1613
		16	0.3242	0.5975	0.1841	0.2449	0.1569
		32	0.3345	0.6003	0.1881	0.2472	0.1587
		2	0.2907	0.5560	0.1639	0.2225	0.1407
		4	0.3176	0.5818	0.1817	0.2402	0.1556
	w/o	8	0.3407	0.6009	0.1991	0.2567	0.1708
		16	0.3517	0.6116	0.2076	0.2651	0.1783
ML-20M		32	0.3597	0.6148	0.2126	0.2691	0.1821
TVILL LOTVI		2	0.2877	0.4428	0.1622	0.2207	0.1394
		4	0.3162	0.5797	0.1812	0.2394	0.1552
	$\mathbf{w}/$	8	0.3355	0.5971	0.1952	0.2531	0.1674
		16	0.3493	0.6094	0.2048	0.2262	0.1755
		32	0.3575	0.6136	0.2110	0.2677	0.1808

Table 9: Comparison of the HSTU model's performance with and without side information. "w/" indicates the inclusion of side information, while "w/o" denotes its absence.

generated by four model variants: HSTU, SASRec, SASRec (w/ residual in HSTU, cf. Equation 3), and SASRec (w/ residual in Llama, cf. Equation 4). We randomly sample 1,000 users from the ML-20M dataset and evaluate the embeddings with 2 and 32 transformer blocks, resulting in a total of 8,000 samples. We apply t-SNE to visualize these embeddings, with the results presented in Figure 4.

The visualization in Figure 4 reveals that embeddings in the shallower model (with 2 blocks) are more clustered compared to those in the deeper model (with 32 blocks). Notably, the HSTU model's embeddings are closest to the coordinate (0,0) in both settings. Given that HSTU demonstrates superior performance in accuracy and scalability, as shown in Tables 1 and 8, we infer that a well-normalized model may enhance both scalability and recall performance. This suggests that normalization is crucial in enhancing the effectiveness of both shallower and deeper models.

5.4 Evaluating HSTU in Complex User Behavioral Sequence Modeling (RQ3)

In this subsection, we further analyze the capacity of large recommendation models in complex user behavioral sequences. We use HSTU as the representative model for the evaluation. Specifically, we concentrate on three primary scenarios: behavior modeling with side information, multi-behavior sequence modeling, and multi-domain sequence modeling.

5.4.1 Behavior Modeling with Side Information. In this section, we investigate the effect of incorporating side information on the user behavior modeling process. We evaluate the performance of HSTU using two datasets, ML-1M and ML-20M, comparing scenarios with and without item attributes. The ML-1M and ML-20M datasets represent the smallest and largest data sizes in our study, respectively, providing a comprehensive range for analysis. These attributes, including movie metadata, are converted into dense vectors and

combined with item ID-based embeddings using mean pooling. Additionally, we vary the model depth to investigate how the inclusion of side information influences the scaling behavior of large recommendation models.

The results are detailed in Table 9. First, we observe that incorporating side information does not necessarily enhance the model's performance. In most instances, the inclusion of side information leads to slightly inferior results compared to the model without it. We hypothesize that this unexpected outcome may be attributed to several factors: (1) the side information employed is relatively simplistic and does not provide significant benefits to the user behavior modeling process, and (2) the method of integrating side information with item ID-based embeddings through mean pooling may be insufficient for extracting meaningful insights from the movie metadata. Future work could explore datasets with more comprehensive side information and employ more sophisticated techniques for integrating side information with user behavior sequences.

Next, we observe that the model maintains its scalability even with the incorporation of side information. This is a desirable attribute, as deeper networks are typically more challenging to train. As shown in Table 9, the model's performance consistently improves with increased depth, using up to 32 blocks. These results indicate that the integration of side information in large recommendation models remains a promising area for further research.

5.4.2 Multi-behavior Modeling. We evaluate the HSTU model for multi-behavior modeling through three experiments: (1) the impact of using multiple behavior training data, (2) the impact of explicit behavior modeling, and (3) the exploration of scaling law in multibehavior scenarios. For this evaluation, we use two multi-behavior datasets, CIKM and IJCAI, to construct user behavior sequences.

Impact of Multiple Behavior Training Data. Users' multi-behavior sequences are constructed by combining different types of user

Dataset	Model	Training	Test	HR@50	NDCG@50	Dataset	Model	Training	Test	HR@50	NDCG@50
	SASRec	all	buy	0.1428	0.0491		SASRec	all	buy	0.1207	0.0488
	SASKet	all	pv	0.1653	0.0642	_	SASKet	an	pv	0.1729	0.0687
	MBSTR	all	buy	0.0922	0.0317		MBSTR all	all	buy	0.0960	0.0339
	MBSTR	an	pv	0.2208	0.0819	_	MIDSTR	an	pv	0.2307	0.0863
01171.4		buy	buy	0.0422	0.0177	IJCAI -		buy	buy	0.0596	0.0222
CIKM			pv	0.0264	0.0148				pv	0.0449	0.0172
		TU pv	buy	0.1104	0.0452			nv	buy	0.1108	0.0386
	HSTU		pv	0.1089	0.0433		HSTU	pv	pv	0.2133	0.0823
		buy & pv	buy	0.1455	0.0575			1 0	buy	0.1060	0.0365
		buy & pv	pv	0.1510	0.0609	-		buy & pv	pv	0.2255	0.0848
		all	buy	0.1431	0.0558			all	buy	0.1049	0.0363
			pv	0.1761	0.0696			all	pv	0.2301	0.0875

Table 10: Performance comparison on multi-behavior sequences. The "all" in the training set represents user sequences that contain all four behaviors.

Model	Training	Test	HR@50	NDCG@50
HSTU	all	buy	0.1431	0.0558
пото	an	pv	0.1761	0.0696
HSTU (w/b)	all	buy	0.1463	0.0566
пэто (w/b)	all	pv	0.1780	0.0712

Table 11: Performance of the HSTU trained on the CIKM dataset. The "w/b" variant denotes the inclusion of explicit behavior modeling. "all" in the training set column refers to user sequences encompassing all four specified behaviors.

behaviors in chronological order, which we refer to as "all". To thoroughly evaluate the impact of multi-behavior data, we create distinct subsets by selectively choosing specific behaviors and combining them in chronological order. Specifically, the training set "buy" includes only purchase behaviors, "pv" comprises solely page view behaviors, and "buy & pv" is a chronological concatenation of purchase and page view behaviors. The model's performance is assessed with page views and purchases as target behaviors. To comprehensively evaluate the effectiveness of the HSTU model, we compare it against several state-of-the-art multi-behavior recommendation models, including SASRec [41] and MBSTR [124]. For a fair comparison, all models are configured with identical parameter settings: four transformer blocks, four attention heads, a batch size of 512, and a learning rate of 1e-3.

The experimental results are shown in Table 10. Overall, the HSTU model outperforms the baseline models (SASRec and MB-STR), though it shows slightly worse performance on certain partitioned datasets. This suggests that while generative models like HSTU are generally superior to traditional models, they may be sensitive to specific data distributions. Incorporating a greater variety of behavioral data into the training set tends to improve recall performance on the target behavior. For instance, in the CIKM dataset, training with page view data ("pv") yields an HR@50 value of 0.1089 for page views. Adding purchase data ("buy & pv") increases the HR@50 to 0.1510, and training with all behavior types further increases it to 0.1761. This demonstrates that exposure to a broader range of behavioral data during training significantly enhances model performance. This finding suggests that recommendation models can benefit from enriched training data, similar to large

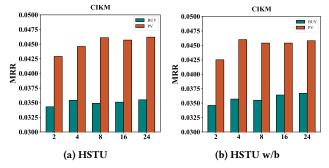


Figure 5: Comparison of the scalability performance between the HSTU model and the HSTU w/b variant on the CIKM dataset. The figure illustrates how each model scales with an increasing transformer layer.

language models (LLMs). Future work should explore the integration of additional behavior types and multi-modal recommendation data to further improve model efficacy.

The results for purchase behavior presented in Table 10 further reinforce this observation. Training on purchase behavior alone yields an HR@50 of 0.0422, which is significantly lower than training on page view behavior, which achieves an HR@50 of 0.1104. This occurs even though the purchase data aligns with the test set distribution, suggesting that larger training sets, even with slightly lower data quality, can still substantially improve performance. However, increasing dataset size does not always lead to better performance. For instance, when comparing training on the full CIKM dataset ("all") versus just purchase and page view behaviors ("buy & pv"), the results for purchase behavior are similar, with some metrics declining slightly. This may be due to the limited correlation and smaller volume of additional behaviors (e.g., add to cart, like) in relation to purchase behavior. Consequently, when the increase in data size is marginal and the quality of the additional data is lower, it can have an adverse impact on overall performance. Future work will focus on thoroughly evaluating the correlation and dependencies between different types of user behaviors and constructing high-quality multi-behavior datasets that effectively strike a balance between data quantity and quality to further enhance model performance and robustness.

Model	Digital Music		Movies & TV		Toys		Video Games	
Model	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
SASRec	0.1332	0.0764	0.0977	0.0577	0.0758	0.0441	0.1228	0.0677
C2DSR	0.1363	0.0772	0.0974	0.0578	0.0745	0.0442	0.1194	0.0658
HSTU	0.1451	0.0860	0.1020	0.0598	0.0704	0.0425	0.1330	0.0738
HSTU-single	0.1004	0.0577	0.1022	0.0597	0.0588	0.0337	0.1055	0.0576

Table 12: Performance comparison for cross-domain recommendation on different target datasets. The "HSTU-single" model variant is trained with a single domain dataset.

# Blocks	Digital Music		Movies & TV		Toys		Video Games	
# DIOCKS	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
2	0.1225	0.0704	0.0909	0.0532	0.0645	0.0383	0.1166	0.0643
4	0.1338	0.0773	0.0957	0.0564	0.0669	0.0399	0.1237	0.0686
8	0.1353	0.0805	0.0971	0.0565	0.0671	0.0404	0.1235	0.0672
16	0.1489	0.0874	0.1031	0.0596	0.0685	0.0404	0.1365	0.0764
32	0.1556	0.0937	0.1090	0.0625	0.0716	0.0416	0.1422	0.0781

Table 13: Performance comparison for cross-domain recommendation on different target datasets with varying number of HSTU blocks.

Impact of Explicit Behavior Modeling. In previous research, models often treated behaviors within sequences as indistinguishable. For instance, when purchase and page view behaviors were concatenated chronologically, the model could not distinguish between them in the input sequence. In this section, we explore whether explicitly modeling behavior types can enhance performance. We achieve this by representing each behavior explicitly as a token in the input sequence, formatted as $X_u = \{x_1, b_1, x_2, b_2, \ldots, x_n, b_n\}$. Using the same training set, we conduct experiments across two subsets: single behavior ("buy" or "pv" only). The results in Table 11 indicate that the model incorporating explicit behavior modeling ("HSTU w/b") generally performs better than the model without this feature ("HSTU"). This demonstrates that adding explicit behavior tokens greatly enhances the model's ability to capture user interaction nuances, leading to improved performance.

Scaling Law in Multi-Behavior Scenarios. To investigate the scaling law of HSTU in multi-behavior scenarios, we systematically increase the number of model layers from 2 to 24. As illustrated in Figures 5, we observe that, with the increase in the number of layers, the overall performance generally improves, with a few exceptions. This trend holds consistently across the IJCAI dataset in our experiments, highlighting the benefits of scaling up model parameters and demonstrating the potential advantages of larger recommendation models.

5.4.3 Multi-domain Modeling. In this section, we evaluate the performance of HSTU in multi-domain modeling. We select several state-of-the-art multi-domain recommendation models as baselines, including C2DSR [4] and SASRec, to ensure a comprehensive comparison. HSTU serves as the backbone model to test its performance and scalability in multi-domain recommendation contexts. We conduct experiments on a multi-domain dataset AMZ-MD.

Performance of HSTU for Multi-domain Recommendation. In practical applications, user behavior data is often collected from multiple scenarios, enabling large recommendation models to learn across domains. We evaluate HSTU's performance on the AMZ-MD dataset and compare it with the baseline models. As shown in Table 12, HSTU outperforms other recommendation models across

most domains, though it exhibits some performance gaps in the Toys domain. This suggests large recommendation models can more effectively learn from multi-domain interaction data due to their extensive parameter capacity. However, when trained on single-domain datasets, HSTU's performance significantly declines in all domains except Movies & TV, highlighting its strength and potential in multi-domain scenarios. The results in the Movies & TV domain indicate that this domain is slightly influenced by knowledge transfer from other domains, with performance primarily determined by the HSTU's single-domain modeling capability.

Scaling Law in HSTU for Multi-domain Recommendation. To explore the scaling law in HSTU, we conduct experiments by varying the number of layers in multi-domain datasets. The results, shown in Table 13, indicate that as the model's complexity increases, performance improves across all domains, paralleling trends observed in single-domain experiments. This improvement is particularly notable in domains with fewer items and interactions, such as Digital Music and Video Games, suggesting that larger models facilitate enhanced cross-domain knowledge transfer. This finding implies that HSTU could be instrumental in addressing the cold-start problem.

5.5 Evaluating HSTU in Ranking Tasks (RQ4)

In this subsection, we aim to explore whether HSTU is still effective and scalable in ranking tasks.

5.5.1 Experimental Settings. In the ranking task, the input sequence $X_u = \{x_1, b'_1, x_2, b'_2, \dots, x_n, b'_n\}$ is processed to produce an output sequence. Formally, this is expressed as $Transformer(X_u) = \{b'_1, x_2, b'_2, x_3, \dots, b'_n, x_{n+1}\}$. At each position in the sequence, a shared small neural network is attached to predict the label b'_i of the user clicking on the item x_i .

To optimize the ranking model's parameters, we employ binary cross-entropy loss $\mathcal{L}_{ranking} = \sum_{u \in \mathcal{U}} \sum_{k=1}^n \text{BCELoss}(b_k'', b_k')$ to minimize the differences between prediction b_k'' and ground-truth label b_k' . We validate the effectiveness of this ranking model through experiments conducted on the ML-1M, ML-20M, and AMZ-Books datasets. We transform user-item interaction in these datasets into a binary feedback format. Specifically, interactions where users rate

Model	# Blocks	ML-1M		ML-20M		AMZ-Books	
		AUC	Logloss	AUC	Logloss	AUC	Logloss
DIN	-	0.7241	0.6141	0.7247	0.6135	0.7060	0.4562
	2	0.7559	0.5814	0.7813	0.5539	0.7257	0.4608
	4	0.7530	0.5821	0.7920	0.5422	0.7386	0.4682
HSTU	8	0.7591	0.5772	0.7960	0.5394	0.7283	0.5134
пото	16	0.7943	0.5318	0.7879	0.5463	0.7442	0.5089
	24	0.7943	0.5307	0.7992	0.5360	0.7450	0.4496
	32	0.7947	0.5341	0.7914	0.5416	0.7606	0.4140
	2	0.7922	0.5403	0.7568	0.5878	0.7181	0.5175
	4	0.7923	0.5592	0.7595	0.5732	0.7585	0.4183
Llama	8	0.7939	0.5454	0.7375	0.5940	0.7449	0.4868
Liailia	16	0.7915	0.5422	0.6390	0.6790	0.7469	0.4896
	24	0.7883	0.5495	0.5777	0.6738	0.7517	0.4250
	32	0.7923	0.5453	0.7107	0.6127	0.7491	0.4653

Table 14: Performance comparison for the ranking task. A higher value indicates a better performance for the AUC metric, while a lower value indicates a better performance for the Logloss metric.

items with a score of 4 or 5 are categorized as positive feedback and assigned a value of 1. Conversely, interactions with ratings below 4 are considered non-positive feedback and assigned a value of 0.

5.5.2 Ranking Performance. Table 14 highlights the superior performance of HSTU compared to Llama as well as the traditional recommendation model DIN. Within the generative framework, Llama generally demonstrates better performance than HSTU when both have the same relatively small number of blocks. However, upon expanding the transformer blocks, we observe that the performance improvement with Llama is minimal and, in some cases, slightly decreases. In contrast, HSTU exhibits better scalability, generally improving as the number of blocks increases.

Our experiments indicate that discrepancies can occur between Logloss and AUC metrics. For instance, although the HSTU model with 32 blocks achieves a higher AUC on the ML-1M dataset compared to the HSTU model with 24 blocks, its Logloss performance is inferior. This finding highlights that a reduction in the Logloss does not consistently correspond to an improvement in AUC, suggesting that a decrease in loss does not necessarily equate to enhanced performance in the recommendation domain. This observation implies that focusing solely on scaling laws may be insufficient; it is essential to investigate underlying performance laws to gain a more comprehensive understanding.

5.5.3 Impact of the Number of Negative Samples. In real-world applications, datasets often exhibit an imbalanced ratio of positive to negative samples, with significantly fewer positive samples compared to negative ones. This imbalance can skew the gradient updates during model training, hindering the model's ability to accurately learn user behavior patterns. To mitigate the challenges posed by this imbalance, traditional approaches selectively sample a subset of negative samples from those the user has interacted with, thereby controlling the proportion of negative samples.

While sampling negative samples can improve the performance of recommendation models, it is not an optimal solution, as it primarily addresses the models' limited capacities to handle complex data. Specifically, the imbalance between positive and negative samples often results in low-quality data, posing a significant challenge for traditional models. These models typically simplify the problem by randomly sampling negative instances, which can lead to the loss

Model	Sampling	AUC					
	Ratio	ML-1M	ML-20M	AMZ-Books			
	0.2	0.7468	0.7903	0.6925			
	0.4	0.7399	0.7952	0.7093			
HSTU	0.6	0.7626	0.7950	0.7324			
	0.8	0.7622	0.7899	0.7338			
	1.0	0.7794	0.7916	0.7037			
	0.2	0.7866	0.7357	0.6824			
	0.4	0.7939	0.7376	0.7159			
Llama	0.6	0.7930	0.7260	0.7313			
	0.8	0.7916	0.7394	0.7414			
	1.0	0.7927	0.7495	0.7433			

Table 15: Performance comparison for the ranking task under varying negative sampling ratios.

of critical information. In this section, we examine whether generative recommendation models can better handle such complexities. We specifically investigate how varying the ratios of negative samples affects model performance.

We randomly sample negative samples at ratios from the set {0.2, 0.4, 0.6, 0.8, 1.0}, where a ratio of 1.0 reflects the model's performance on the original dataset. The experimental results are shown in Table 15. It can be observed that increasing the sampling ratio and thereby augmenting the number of negative samples, leads to a continuous improvement in the model's performance. This improvement suggests that the generative recommendation model is adept at handling complex data and possesses superior modeling capabilities, as it benefits from the enriched information provided by the additional negative samples. Consequently, it is well-suited for complex real-world scenarios, offering significant application potential. Furthermore, the performance improvements are more pronounced in the larger dataset (ML-20M) compared to the smaller dataset (ML-1M). This observation underscores the importance of understanding data scaling laws: while expanding the dataset generally enhances model performance, the benefits appear to exhibit diminishing returns as the dataset size increases.

5.5.4 The Impact of Scoring Network Architecture. To further examine the impact of scoring network architecture, we employ a small

Model		AUC				
	Architecture	ML-1M	ML-20M	AMZ-Books		
HSTU	Dot	0.7886	0.7576	0.7468		
	MLP	0.7489	0.7913	0.7386		
	FFN	0.7765	0.7920	0.7348		
Llama	Dot	0.7945	0.7137	0.7177		
	MLP	0.7923	0.7311	0.7585		
	FFN	0.7901	0.7573	0.7402		

Table 16: Performance comparison for the ranking task with different scoring network architectures.

Model	# Blocks	AUC				
		ML-1M	ML-20M	AMZ-Books		
	2	0.7759 (†)	0.7326 (\)	0.7363 (↑)		
	4	0.7866 (↑)	0.7623 (\)	0.7495 (↑)		
HOTH	8	0.7772 (↑)	0.7714 (\)	0.7511 (↑)		
HSTU	16	0.7910 (\)	0.7697 (\)	0.7488 (↑)		
	24	0.7882 (↑)	0.7728 (\)	0.7609 (↑)		
	32	0.7872 (\)	0.7660 (\)	0.7611 (↑)		

Table 17: Performance of HSTU for the ranking task with a smaller item embedding size. The arrows indicate the direction of change in AUC compared to Table 14: "↑" indicates an increase and "↓" indicates a decrease.

neural network to generate scores for the ranking stage, utilizing the output from the models. To assess the impact of different neural network architectures, we implement three variations: (a) dot product (Dot), (b) multi-layer perceptron (MLP), and (c) Feed-forward networks (FFN). The detailed architectures of the neural networks are illustrated in Figure 6, where b_t^\prime denotes the target label. Among these, the dot product structure represents the simplest architecture, while the FFN is the most complex.

The experimental results, presented in Table 16, reveal that for smaller datasets such as ML-1M and AMZ-Books, simpler scoring network architectures result in better model performance. Conversely, for larger datasets like ML-20M, more complex architectures yield superior performance. Larger datasets are generally better suited to more complex neural network architectures. These findings underscore the principle that stronger model capabilities do not universally translate to better performance; instead, optimal results are achieved when the architecture is appropriately tailored to the dataset size and complexity.

5.5.5 Model Performance for Reduced Embedding Size. As discussed in the previous section, achieving optimal recommendation performance requires aligning the model's expressive capability with the dataset size. For smaller datasets, increasing the model's expressive capability does not always result in better performance. To further investigate this finding, we conduct experiments using a very small embedding size of 4. This contrasts with the embedding sizes used in previous experiments, which are 50 for ML-1M, 256 for ML-20M, and 64 for AMZ-Books. The results of these experiments are presented in Table 17.

Interestingly, when comparing the results with larger embedding sizes in Table 14, reducing the embedding size to a very small value improves performance for the smaller datasets, ML-1M and AMZ-Books. However, for the larger dataset, ML-20M, performance consistently declines. This observation indicates that for small datasets,

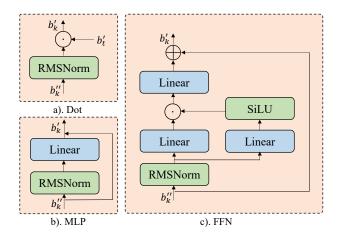


Figure 6: Illustration of scoring network architectures.

using an excessively large embedding size may not be necessary and could even negatively impact model performance.

At the same time, as shown in Table 17, a smaller embedding size results in poorer scaling laws for ML-20M, despite the model's reduced parameter size compared to those in Table 14. This highlights an often-overlooked aspect: the scaling laws for generative recommendation models in ranking tasks are influenced not only by the vertical expansion (i.e., blocks of the transformer) of the model but also by its horizontal scale (i.e., embedding size). Previous studies on model parameter scaling have largely ignored the exploration of horizontal expansion.

Overall, Table 17 supports our conclusion that model size should match dataset size, with larger datasets requiring larger models. This finding leads to two key insights for future research directions:

- In practical applications, the complexity and volume of user interaction data require more sophisticated models, highlighting the enhanced application potential and research value of generative large recommendation models that have scaling laws.
- Parameter tuning for different datasets to achieve optimal performance can be resource-intensive. Future research should explore strategies for determining the optimal horizontal and vertical scaling of models (including embedding size and the number of layers) tailored to different datasets, to streamline the process and enhance recommendation performance.

6 FUTURE DIRECTIONS

In this paper, we have systematically reviewed the recent advancements in large recommendation models. Our experiments involve both preliminary explorations and extensive evaluations using a combination of public datasets. These efforts highlight the significant potential of large recommendation models for future research and practical applications. Despite these promising findings, it is important to acknowledge that the field of large recommendation models is still in its nascent stages. Our paper represents only an initial foray into this area, and further investigation is necessary to deepen understanding and enhance these models.

To aid future research endeavors, we have also identified and summarized several key perspectives from a substantial body of prior research that were not addressed in this paper. These insightful perspectives can help researchers pinpoint critical issues and pain points, thus offering promising opportunities for future exploration. Specifically, these perspectives primarily include unresolved challenges in data engineering, the application of tokenizers, and training/inference efficiency.

6.1 Data Engineering

The advent of deep learning has spurred the development of novel model-based recommendation methods aimed at enhancing performance. Despite these advancements, such methods often neglect a critical aspect: the analysis and quality improvement of the underlying recommendation data. A previous study [11] has demonstrated that the intrinsic characteristics of different recommendation datasets significantly influence the outcomes and comparative analyses of recommendation systems.

Recognizing this challenge, researchers are increasingly aware of the pivotal role of data quality in recommendation systems, which has led to the rise of data-centric approaches. The core principle of these approaches is that the quality and characteristics of the dataset fundamentally constrain model performance [49, 121]. To address data-related limitations, solutions such as data synthesis [35, 100, 106, 119] and data denoising [57, 75, 102, 129] have been proposed.

In parallel, the rapid development of large language models (LLMs) has showcased their exceptional capabilities across various natural language tasks. Researches [31, 42] highlight that the performance of LLMs consistently improves with increases in both the scale of training data and model size. This has shifted attention towards data engineering within LLMs, focusing on aspects such as data composition, quality control, and quantity [104].

Given the critical role of data in both traditional recommendation models and LLMs, we posit that data engineering represents a vital research avenue in the realm of large recommendation models. Specifically, two key research directions emerge: quantifying the impact of data scale on model performance and defining as well as enhancing the quality of recommendation datasets. These areas hold significant potential for advancing the field.

6.2 Tokenizer Application

Tokenizers [39, 50, 52, 76, 79, 93] are fundamental components in deep learning, serving as a bridge between raw data and models. They play a crucial role in both recommendation systems and natural language processing (NLP) by enhancing the efficiency and scalability of deep learning solutions.

In recommendation systems, tokenizers convert user and item IDs into meaningful embedded representations, effectively capturing user preferences and item features. This transformation has spurred the development of various encoding techniques, including ID-based [29, 45, 55], text-based [5, 7, 12, 43, 134], graph-based [17, 28, 44], and compression encoding techniques [38, 105]. These methods provide practical advantages in processing text data efficiently and enriching semantic information.

In the development of large language models (LLMs), tokenization technology has progressed from simple rule-based methods [3, 46, 67, 71, 77, 114, 125] to sophisticated context-aware models [47, 82, 83]. These advancements enhance the accuracy of language understanding and generation, while also broadening the potential for integrating diverse data modalities.

Given the critical role of tokenizers in recommendation systems and large models, they are poised to become even more crucial in large recommendation systems. This is particularly relevant when expanding recommendation datasets, as data expansion leads to a rapid increase in vocabulary size. The unique, dynamic, and extensive vocabulary in recommendation scenarios underscores the importance of tokenizers. We anticipate significant research opportunities in this area, focusing on the development of efficient and lossless tokenizers. Such research aims to minimize context information loss, improve processing speed and efficiency, and design tokenization strategies tailored to different modality features, thereby reducing text dependency. These studies hold promise for enhancing the capability and efficiency of tokenizers.

6.3 Training/Inference Efficiency

As data and parameters continue to expand, the computational and storage demands increase, creating a significant bottleneck in scaling models due to training and inference inefficiencies.

To address these challenges, researchers have explored various methods to scale LLMs. Techniques such as data parallelism [80, 84] and model parallelism [19, 37, 54, 103] are used to efficiently handle large data volumes and model requests efficiently. Model structure compression techniques [32, 36, 56, 61, 88, 133] simplify models at the algorithmic level, computational graph reconstruction methods [9, 14, 80, 127] enhance efficiency during compilation, and system optimizations [2, 40, 48, 48, 66, 68] improve overall throughput and reduce latency.

While there has been progress in optimizing the efficiency of general LLMs, the recommendation domain has seen limited exploration [89, 97, 108, 115, 116, 122], particularly concerning emerging large recommendation models. Recommendation systems face unique challenges, such as feature sparsity and the massive scale of users and items, resulting in a daily token processing volume that can be several orders of magnitude larger than what general LLMs handle over months. This imposes a substantial training burden and necessitates stringent real-time inference requirements.

Improving the training and inference efficiency and throughput of recommendation systems is thus a crucial research direction. Specific focus areas include achieving targeted software and hardware co-optimization based on the unique characteristics of recommendation systems, efficiently managing streaming data for parameter updates, and deploying systems with low resource requirements while ensuring efficient and accurate inference.

7 CONCLUSION

The emergence of large language models heightened interest in the scalability of models within the research community. The discovery of scaling laws had significant implications for recommendation systems as well. While Meta introduced HSTU and observed scaling law, several questions about scaling laws in large recommendation models remained unresolved. In this paper, we investigated the potential of large recommendation models, especially HSTU, across various recommendation tasks and the scaling laws they exhibited. Firstly, we implemented a range of transformer-based architectures for large recommendation models and assessed their performance as model parameters were scaled up. We then conducted ablation studies to identify key modules that influenced scaling law. Furthermore, we examined the application potential of HSTU across different recommendation tasks. Additionally, we explored, for the first time, the performance and scaling laws of HSTU in ranking tasks. We believe this paper will shed light on future research regarding large recommendation models.

REFERENCES

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023).
- [2] Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav S Gulavani, Alexey Tumanov, and Ramachandran Ramjee. 2024. Taming throughput-latency tradeoff in llm inference with sarathi-serve. arXiv preprint arXiv:2403.02310 (2024).
- [3] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. 1992. Class-based n-gram models of natural language. Computational linguistics 18, 4 (1992), 467–480.
- [4] Jie Cao, Xinyu Cong, Jianxin Sheng, et al. 2022. Contrastive Cross-Domain Sequential Recommendation. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 138–147.
- [5] Rose Catherine and William Cohen. 2017. Transnets: Learning to transform for recommendation. In Proceedings of the 11th ACM Conference on Recommender Systems. 288–296.
- [6] Jianxin Chang, Chenbin Zhang, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. Pepnet: Parameter and embedding personalized network for infusing with personalized prior information. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 3795–3804.
- [7] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In Proceedings of the 2018 World Wide Web conference. 1583–1592.
- [8] Qiwei Chen, Changhua Pei, Shanshan Lv, Chao Li, Junfeng Ge, and Wenwu Ou. 2021. End-to-end user behavior retrieval in click-through rateprediction model. arXiv preprint arXiv:2108.04468 (2021).
- [9] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, et al. 2018. {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In 13th USENIX Symposium on Operating Systems Design and Implementation. 578–594.
- [10] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In Proceedings of the eleventh ACM international conference on web search and data mining. 108–116.
- [11] Jin Yao Chin, Yile Chen, and Gao Cong. 2022. The Datasets Dilemma: How Much Do We Really Know About Recommendation Datasets?. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. 141–149.
- [12] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based Neural Recommender. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management. 147–156.
- [13] Sung Min Cho, Eunhyeok Park, and Sungjoo Yoo. 2020. MEANTIME: Mixture of Attention Mechanisms with Multi-temporal Embeddings for Sequential Recommendation. In Proceedings of the 14th ACM Conference on Recommender Systems. 515–520.
- [14] Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. arXiv preprint arXiv:2307.08691 (2023).
- [15] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2019. Neural Multi-task Recommendation from Multi-behavior Data. In 2019 IEEE 35th International Conference on Data Engineering. IEEE. 1554–1557.
- [16] Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. 2022. Kuairand: an unbiased sequential recommendation dataset with randomly exposed videos. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 3953–3957.
- [17] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 855–864.
- [18] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Lixin Zou, Yiding Liu, and Dawei Yin. 2020. Deep multifaceted transformers for multi-objective ranking in largescale e-commerce recommender systems. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management.
- [19] Huifeng Guo, Wei Guo, Yong Gao, Ruiming Tang, Xiuqiang He, and Wenzhi Liu. 2021. Scalefreectr: Mixcache-based distributed training system for ctr models with huge embedding table. In Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval. 1269–1278.
- [20] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. arXiv preprint arXiv:1703.04247 (2017).
- [21] Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, and Bin Cui. 2019. Buying or browsing?: Predicting real-time purchasing intent using attention-based deep network with multiple behavior. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 1984–1992.
- [22] Wei Guo, Chang Meng, Enming Yuan, Zhicheng He, Huifeng Guo, Yingxue Zhang, Bo Chen, Yaochen Hu, Ruiming Tang, Xiu Li, et al. 2023. Compressed interaction graph based framework for multi-behavior recommendation. In Proceedings of the ACM Web Conference 2023. 960–970.

- [23] Wei Guo, Rong Su, Renhao Tan, Huifeng Guo, Yingxue Zhang, Zhirong Liu, Ruiming Tang, and Xiuqiang He. 2021. Dual graph enhanced embedding neural network for CTR prediction. In Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. 496–504.
- [24] Xingzhuo Guo, Junwei Pan, Ximei Wang, Baixu Chen, Jie Jiang, and Mingsheng Long. 2023. On the Embedding Collapse when Scaling up Recommendation Models. arXiv preprint arXiv:2310.04400 (2023).
- [25] Yongqiang Han, Hao Wang, Kefan Wang, Likang Wu, Zhi Li, Wei Guo, Yong Liu, Defu Lian, and Enhong Chen. 2024. Efficient Noise-Decoupling for Multi-Behavior Sequential Recommendation. In Proceedings of the ACM on Web Conference 2024. 3297–3306.
- [26] Yongqiang Han, Likang Wu, Hao Wang, Guifeng Wang, Mengdi Zhang, Zhi Li, Defu Lian, and Enhong Chen. 2023. Guesr: A global unsupervised dataenhancement with bucket-cluster sampling for sequential recommendation. In International Conference on Database Systems for Advanced Applications. Springer, 286–296.
- [27] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems 5, 4 (2015), 1–19
- [28] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang, 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 639–648.
- [29] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web. 173–182.
- [30] Zhicheng He, Weiwen Liu, Wei Guo, Jiarui Qin, Yingxue Zhang, Yaochen Hu, and Ruiming Tang. 2023. A survey on user behavior modeling in recommender systems. arXiv preprint arXiv:2302.11087 (2023).
- [31] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. An empirical analysis of compute-optimal large language model training. Advances in Neural Information Processing Systems 35 (2022), 30016–30030.
- [32] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2024. Kvquant: Towards 10 million context length llm inference with kv cache quantization. arXiv preprint arXiv:2401.18079 (2024).
- [33] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In Proceedings of the ACM Web Conference 2023. 1162–1171.
- [34] 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.
- [35] Tinglin Huang, Yuxiao Dong, Ming Ding, Zhen Yang, Wenzheng Feng, Xinyu Wang, and Jie Tang. 2021. Mixgcf: An improved training method for graph neural network-based recommender systems. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 665–674.
- [36] Yukun Huang, Yanda Chen, Zhou Yu, and Kathleen McKeown. 2022. In-context learning distillation: Transferring few-shot learning ability of pre-trained language models. arXiv preprint arXiv:2212.10670 (2022).
- [37] Dmytro Ivchenko, Dennis Van Der Staay, Colin Taylor, Xing Liu, Will Feng, Rahul Kindi, Anirudh Sudarshan, and Shahin Sefati. 2022. Torchrec: a pytorch domain library for recommendation systems. In Proceedings of the 16th ACM Conference on Recommender Systems. 482–483.
- [38] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence 33, 1 (2010), 117–128.
- [39] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and visionlanguage representation learning with noisy text supervision. In *International* conference on machine learning. PMLR, 4904–4916.
- [40] Yunho Jin, Chun-Feng Wu, David Brooks, and Gu-Yeon Wei. 2023. S³: Increasing GPU Utilization during Generative Inference for Higher Throughput. Advances in Neural Information Processing Systems 36 (2023), 18015–18027.
- [41] W. Kang and J. McAuley. 2018. Self-Attentive Sequential Recommendation. In 2018 IEEE International Conference on Data Mining. IEEE Computer Society, 197–206
- [42] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361 (2020).
- [43] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In Proceedings of the 10th ACM Conference on Recommender Systems. 233–240.
- [44] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016).
- [45] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. 426–434.
- [46] Garima Koushik, K Rajeswari, and Suresh Kannan Muthusamy. 2019. Automated hate speech detection on Twitter. In 2019 5th International Conference

- On Computing, Communication, Control And Automation. IEEE, 1-4.
- [47] Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. arXiv preprint arXiv:1804.10959 (2018).
- [48] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In Proceedings of the 29th Symposium on Operating Systems Principles. 611–626.
- [49] Riwei Lai, Li Chen, Rui Chen, and Chi Zhang. 2024. A Survey on Data-Centric Recommender Systems. arXiv preprint arXiv:2401.17878 (2024).
- [50] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 11523– 11532
- [51] Cheng Li, Ming Zhao, Haoyan Zhang, et al. 2022. RecGURU: Adversarial Learning of Generalized User Representations for Cross-domain Recommendation. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM). 571–581.
- [52] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text is all you need: Learning language representations for sequential recommendation. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 1258–1267.
- [53] 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.
- [54] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling distributed machine learning with the parameter server. In 11th USENIX Symposium on operating systems design and implementation. 583–598.
- [55] Xiucheng Li, Jin Yao Chin, Yile Chen, and Gao Cong. 2021. Sinkhorn Collaborative Filtering. In Proceedings of the Web Conference 2021. 582–592.
- [56] Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning*. PMLR, 20852–20867.
- [57] Weilin Lin, Xiangyu Zhao, Yejing Wang, Yuanshao Zhu, and Wanyu Wang. 2023. Autodenoise: Automatic data instance denoising for recommendations. In Proceedings of the ACM Web Conference 2023. 1003–1011.
- [58] Weiwen Liu, Wei Guo, Yong Liu, Ruiming Tang, and Hao Wang. 2023. User Behavior Modeling with Deep Learning for Recommendation: Recent Advances. In Proceedings of the 17th ACM Conference on Recommender Systems. 1286–1287.
- [59] Z Liu, Y Hou, and J McAuley. 2024. Multi-Behavior Generative Recommendation. arXiv preprint arXiv:2405.16871 (2024).
- [60] H. Ma, R. Xie, L. Meng, et al. 2024. Triple Sequence Learning for Cross-Domain Recommendation. ACM Transactions on Information Systems 42, 4 (2024), 1–29. To appear.
- [61] Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. Advances in neural information processing systems 36 (2023), 21702–21720.
- [62] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In Proceedings of the 7th ACM conference on Recommender systems. 165–172.
- [63] Brendan McMahan. 2011. Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, 525–533.
- [64] Chang Meng, Hengyu Zhang, Wei Guo, Huifeng Guo, Haotian Liu, Yingxue Zhang, Hongkun Zheng, Ruiming Tang, Xiu Li, and Rui Zhang. 2023. Hierarchical projection enhanced multi-behavior recommendation. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 4649–4660.
- [65] Chang Meng, Ziqi Zhao, Wei Guo, Yingxue Zhang, Haolun Wu, Chen Gao, Dong Li, Xiu Li, and Ruiming Tang. 2023. Coarse-to-fine knowledge-enhanced multi-interest learning framework for multi-behavior recommendation. ACM Transactions on Information Systems 42, 1 (2023), 1–27.
- [66] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. 2023. SpecInfer: Accelerating Generative Large Language Model Serving with Tree-based Speculative Inference and Verification. arXiv preprint arXiv:2305.09781 (2023).
- [67] T Mikolov. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013).
- [68] ModelTC. 2024. Lightllm. https://github.com/ModelTC/lightllm. [Online].
- [69] Nikil Pancha, Andrew Zhai, Jure Leskovec, and Charles Rosenberg. 2022. Pinner-former: Sequence modeling for user representation at pinterest. In Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining. 3702–3712.
- [70] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems 32 (2019).
 [71] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove:
- [71] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on

- empirical methods in natural language processing, 1532-1543.
- [72] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2671–2679.
- [73] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 2685–2692.
- [74] Jiarui Qin, Weinan Zhang, Rong Su, Zhirong Liu, Weiwen Liu, Guangpeng Zhao, Hao Li, Ruiming Tang, Xiuqiang He, and Yong Yu. 2023. Learning to retrieve user behaviors for click-through rate estimation. ACM Transactions on Information Systems 41, 4 (2023), 1–31.
- [75] Yuhan Quan, Jingtao Ding, Chen Gao, Lingling Yi, Depeng Jin, and Yong Li. 2023. Robust preference-guided denoising for graph based social recommendation. In Proceedings of the ACM Web Conference 2023. 1097–1108.
- [76] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In International conference on machine learning. PMLR, 8748–8763.
- [77] Nived Rajaraman, Jiantao Jiao, and Kannan Ramchandran. 2024. Toward a Theory of Tokenization in LLMs. arXiv preprint arXiv:2404.08335 (2024).
- [78] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2024. Recommender systems with generative retrieval. Advances in Neural Information Processing Systems 36 (2024).
- [79] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International conference on machine learning*. Pmlr, 8821–8831.
- [80] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 3505–3506.
- [81] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, et al. 2019. Lifelong sequential modeling with personalized memorization for user response prediction. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. 565–574.
- [82] Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE. 5149-5152.
- [83] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909 (2015).
- [84] Alexander Sergeev and Mike Del Balso. 2018. Horovod: fast and easy distributed deep learning in TensorFlow. arXiv preprint arXiv:1802.05799 (2018).
- [85] Tingjia Shen, Hao Wang, Jiaqing Zhang, Sirui Zhao, Liangyue Li, Zulong Chen, Defu Lian, and Enhong Chen. 2024. Exploring User Retrieval Integration towards Large Language Models for Cross-Domain Sequential Recommendation. arXiv preprint arXiv:2406.03085 (2024).
- [86] Jiajie Su, Chaochao Chen, Zibin Lin, Xi Li, Weiming Liu, and Xiaolin Zheng. 2023. Personalized Behavior-Aware Transformer for Multi-Behavior Sequential Recommendation. In Proceedings of the 31st ACM International Conference on Multimedia.
- [87] 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. Association for Computing Machinery, 1441–1450.
- [88] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. arXiv preprint arXiv:2306.11695 (2023).
- [89] Zhen Tian, Ting Bai, Zibin Zhang, Zhiyuan Xu, Kangyi Lin, Ji-Rong Wen, and Wayne Xin Zhao. 2023. Directed acyclic graph factorization machines for CTR prediction via knowledge distillation. In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. 715–723.
- [90] Tianchi. 2018. IJCAI-15 Repeat Buyers Prediction Dataset. https://tianchi. aliyun.com/dataset/dataDetail?dataId=42
- [91] Junxiong Tong, Mingjia Yin, Hao Wang, Qiushi Pan, Defu Lian, and Enhong Chen. 2024. MDAP: A Multi-view Disentangled and Adaptive Preference Learning Framework for Cross-Domain Recommendation. arXiv preprint arXiv:2410.05877 (2024).
- [92] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023).
- [93] Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. Advances in neural information processing systems 30 (2017).
- [94] Hao Wang, Yongqiang Han, Kefan Wang, Kai Cheng, Zhen Wang, Wei Guo, Yong Liu, Defu Lian, and Enhong Chen. 2024. Denoising Pre-Training and Customized

- Prompt Learning for Efficient Multi-Behavior Sequential Recommendation. arXiv preprint arXiv:2408.11372 (2024).
- [95] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. 2021. Decoupled representation learning for attributed networks. IEEE Transactions on Knowledge and Data Engineering 35, 3 (2021), 2430–2444.
- [96] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. 2021. Hypersorec: Exploiting hyperbolic user and item representations with multiple aspects for social-aware recommendation. ACM Transactions on Information Systems 40, 2 (2021), 1–28.
- [97] Huanyu Wang, Junjie Liu, Xin Ma, Yang Yong, Zhenhua Chai, and Jianxin Wu. 2022. Compressing models with few samples: Mimicking then replacing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 701–710.
- [98] Hao Wang, Tong Xu, Qi Liu, Defu Lian, Enhong Chen, Dongfang Du, Han Wu, and Wen Su. 2019. MCNE: An end-to-end framework for learning multiple conditional network representations of social network. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. 1064–1072.
- [99] Hao Wang, Mingjia Yin, Luankang Zhang, Sirui Zhao, and Enhong Chen. [n. d.]. MF-GSLAE: A Multi-Factor User Representation Pre-training Framework for Dual-Target Cross-Domain Recommendation. ACM Transactions on Information Systems ([n. d.]).
- [100] Qinyong Wang, Hongzhi Yin, Hao Wang, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2019. Enhancing collaborative filtering with generative augmentation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 548–556.
- [101] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In Proceedings of the ADKDD'17. 1–7.
- [102] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising implicit feedback for recommendation. In Proceedings of the 14th ACM international conference on web search and data mining. 373–381.
- [103] Zehuan Wang, Yingcan Wei, Minseok Lee, Matthias Langer, Fan Yu, Jie Liu, Shijie Liu, Daniel G Abel, Xu Guo, Jianbing Dong, et al. 2022. Merlin hugeCTR: GPU-accelerated recommender system training and inference. In Proceedings of the 16th ACM Conference on Recommender Systems. 534–537.
- [104] Zige Wang, Wanjun Zhong, Yufei Wang, Qi Zhu, Fei Mi, Baojun Wang, Lifeng Shang, Xin Jiang, and Qun Liu. 2024. Data Management For Training Large Language Models: A Survey. arXiv:cs.CL/2312.01700
- [105] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. 2009. Feature Hashing for Large Scale Multitask Learning. In Proceedings of the 26th annual international conference on machine learning. 1113–1120.
- [106] Le Wu, Yonghui Yang, Kun Zhang, Richang Hong, Yanjie Fu, and Meng Wang. 2020. Joint item recommendation and attribute inference: An adaptive graph convolutional network approach. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 679–688.
- [107] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A Survey on Large Language Models for Recommendation. World Wide Web 27, 5 (2024), 60.
- [108] Yongji Wu, Defu Lian, Neil Zhenqiang Gong, Lu Yin, Mingyang Yin, Jingren Zhou, and Hongxia Yang. 2021. Linear-time self attention with codeword histogram for efficient recommendation. In Proceedings of the Web Conference 2021. 1262–1273.
- [109] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Xiang Ao, Xin Chen, Xu Zhang, Fuzhen Zhuang, Leyu Lin, and Qing He. 2022. Multi-view multi-behavior contrastive learning in recommendation. In Database Systems for Advanced Applications: 27th International Conference. Springer, 166–182.
- [110] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Bo Zhang, and Liefeng Bo. 2020. Multiplex behavioral relation learning for recommendation via memory augmented transformer network. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2397– 2406.
- [111] Wenjia Xie, Hao Wang, Luankang Zhang, Rui Zhou, Defu Lian, and Enhong Chen. 2024. Breaking Determinism: Fuzzy Modeling of Sequential Recommendation Using Discrete State Space Diffusion Model. arXiv preprint arXiv:2410.23994 (2024)
- [112] Wenjia Xie, Rui Zhou, Hao Wang, Tingjia Shen, and Enhong Chen. 2024. Bridging User Dynamics: Transforming Sequential Recommendations with Schrödinger Bridge and Diffusion Models. In Proceedings of the 33rd ACM International Conference on Information and Knowledge Management. 2618–2628.
- [113] Xiang Xu, Hao Wang, Wei Guo, Luankang Zhang, Wanshan Yang, Runlong Yu, Yong Liu, Defu Lian, and Enhong Chen. 2024. Multi-granularity Interest Retrieval and Refinement Network for Long-Term User Behavior Modeling in CTR Prediction. arXiv preprint arXiv:2411.15005 (2024).
- [114] Fadi Yamout and Rachad Lakkis. 2018. Improved TFIDF weighting techniques in document Retrieval. In 2018 Thirteenth International Conference on Digital Information Management. 69–73.
- [115] Bencheng Yan, Pengjie Wang, Jinquan Liu, Wei Chao Lin, Kuang Chih Lee, Jian Xu, and Bo Zheng. 2021. Binary Code based Hash Embedding for Webscale Applications. Proceedings of the 30th ACM International Conference on Information & Knowledge Management (2021).

- [116] Xuanhua Yang, Xiaoyu Peng, Penghui Wei, Shaoguo Liu, Liang Wang, and Bo Zheng. 2022. Adasparse: Learning adaptively sparse structures for multi-domain click-through rate prediction. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 4635–4639.
- [117] Wenwen Ye, Shuaiqiang Wang, Xu Chen, Xuepeng Wang, Zheng Qin, and Dawei Yin. 2020. Time Matters: Sequential Recommendation with Complex Temporal Information. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 1459–1468.
- [118] Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Zhi Li, Sirui Zhao, Zhen Wang, Defu Lian, and Enhong Chen. 2024. Learning Partially Aligned Item Representation for Cross-Domain Sequential Recommendation. arXiv preprint arXiv:2405.12473 (2024)
- [119] Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Suojuan Zhang, Sirui Zhao, Defu Lian, and Enhong Chen. 2024. Dataset Regeneration for Sequential Recommendation. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 3954–3965.
- [120] Mingjia Yin, Hao Wang, Xiang Xu, Likang Wu, Sirui Zhao, Wei Guo, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. 2023. APGL4SR: A Generic Framework with Adaptive and Personalized Global Collaborative Information in Sequential Recommendation. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. 3009–3019.
- [121] Mingjia Yin, Chuhan Wu, Yufei Wang, Hao Wang, Wei Guo, Yasheng Wang, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. 2024. Entropy Law: The Story behind Data Compression and LLM Performance. arXiv preprint arXiv:2407.06645 (2024).
- [122] Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. 2022. Orca: A distributed serving system for {Transformer-Based} generative models. In 16th USENIX Symposium on Operating Systems Design and Implementation. 521-538.
- [123] Wenhui Yu, Chao Feng, Yanze Zhang, Lantao Hu, Peng Jiang, and Han Li. 2024. IFA: Interaction Fidelity Attention for Entire Lifelong Behaviour Sequence Modeling. arXiv preprint arXiv:2406.09742 (2024).
- [124] Enming Yuan, Wei Guo, Zhicheng He, Huifeng Guo, Chengkai Liu, and Ruiming Tang. 2022. Multi-behavior sequential transformer recommender. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. 1642–1652.
- [125] Huanhuan Yuan, Yongli Wang, Xia Feng, and Shurong Sun. 2018. Sentiment analysis based on weighted word2vec and att-lstm. In Proceedings of the 2018 2nd international conference on computer science and artificial intelligence. 420–424.
- [126] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. [n. d.]. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. In Forty-first International Conference on Machine Learning.
- [127] Yujia Zhai, Chengquan Jiang, Leyuan Wang, Xiaoying Jia, Shang Zhang, Zizhong Chen, Xin Liu, and Yibo Zhu. 2023. ByteTransformer: A high-performance transformer boosted for variable-length inputs. In 2023 IEEE International Parallel and Distributed Processing Symposium. 344–355.
- [128] Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Daifeng Guo, Yanli Zhao, Shen Li, Yuchen Hao, Yantao Yao, et al. 2024. Wukong: Towards a Scaling Law for Large-Scale Recommendation. arXiv preprint arXiv:2403.02545 (2024).
- [129] Chi Zhang, Yantong Du, Xiangyu Zhao, Qilong Han, Rui Chen, and Li Li. 2022. Hierarchical item inconsistency signal learning for sequence denoising in sequential recommendation. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 2508–2518.
- [130] Luankang Zhang, Hao Wang, Suojuan Zhang, Mingjia Yin, Yongqiang Han, Jiaqing Zhang, Defu Lian, and Enhong Chen. 2024. A Unified Framework for Adaptive Representation Enhancement and Inversed Learning in Cross-Domain Recommendation. arXiv preprint arXiv:2404.00268 (2024).
- [131] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep learning for click-through rate estimation. arXiv preprint arXiv:2104.10584 (2021).
- [132] Weijie Zhao, Deping Xie, Ronglai Jia, Yulei Qian, Ruiquan Ding, Mingming Sun, and Ping Li. 2020. Distributed hierarchical gpu parameter server for massive scale deep learning ads systems. Proceedings of Machine Learning and Systems 2 (2020), 412–428.
- [133] Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2024. Atom: Lowbit quantization for efficient and accurate llm serving. Proceedings of Machine Learning and Systems 6 (2024), 196–209.
- [134] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In Proceedings of the 10th ACM International Conference on Web Search and Data Mining. 425–434.
- [135] Guorui Zhou, Weijie Bian, Kailun Wu, Lejian Ren, Qi Pi, Yujing Zhang, Can Xiao, Xiang-Rong Sheng, Na Mou, Xinchen Luo, et al. 2020. CAN: revisiting feature co-action for click-through rate prediction. arXiv preprint arXiv:2011.05625 (2020)
- [136] 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.