Enhancing Sequential Recommendation via LLM-based Semantic Embedding Learning

Jun Hu*
zhaoda.hj@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Wenwen Xia*
wenzhen.xww@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Xiaolu Zhang
yueyin.zxl@antfin.com
Ant Group
Hangzhou, Zhejiang, China

Chilin Fu
chilin.fcl@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Weichang Wu
jiuyue.wwc@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Zhaoxin Huan
zhaoxin.hzx@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Ang Li
liang268038@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Zuoli Tang
tangzuoli@whu.edu.cn
Wuhan University
Wuhan, Hubei, China

Jun Zhou†
jun.zhoujun@antfin.com
Ant Group
Hangzhou, Zhejiang, China

ABSTRACT

Sequential recommendation systems (SRS) are crucial in various applications as they enable users to discover relevant items based on their past interactions. Recent advancements involving large language models (LLMs) have shown significant promise in addressing intricate recommendation challenges. However, these efforts exhibit certain limitations. Specifically, directly extracting representations from an LLM based on items' textual features and feeding them into a sequential model hold no guarantee that the semantic information of texts could be preserved in these representations. Additionally, concatenating textual descriptions of all items in an item sequence into a long text and feeding it into an LLM for recommendation results in lengthy token sequences, which largely diminishes the practical efficiency.

In this paper, we introduce SAID, a framework that utilizes LLMs to explicitly learn Semantically Aligned item ID embeddings based on texts. For each item, SAID employs a projector module to transform an item ID into an embedding vector, which will be fed into an LLM to elicit the exact descriptive text tokens accompanied by the item. The item embeddings are *forced* to preserve fine-grained semantic information of textual descriptions. Further, the learned embeddings can be integrated with lightweight downstream sequential models for practical recommendations. In this way, SAID circumvents lengthy token sequences in previous works, reducing resources required in industrial scenarios and also achieving superior recommendation performance. Experiments on six public datasets demonstrate that SAID outperforms baselines by about 5% to 15% in terms of NDCG@10. Moreover, SAID has been deployed in Alipay's online advertising platform, achieving a 3.07% relative improvement of cost per mille (CPM) over baselines, with an online response time of under 20 milliseconds.

CCS CONCEPTS

• **Information systems** → **Social recommendation**; **Content ranking**.

KEYWORDS

Sequential recommendation, Large language models

ACM Reference Format:
Jun Hu, Wenwen Xia, Xiaolu Zhang, Chilin Fu, Weichang Wu, Zhaoxin Huan, Ang Li, Zuoli Tang, and Jun Zhou. 2024. Enhancing Sequential Recommendation via LLM-based Semantic Embedding Learning. In *Companion Proceedings of the ACM Web Conference 2024 (WWW'24 Companion), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3589335.3648307

1 INTRODUCTION

Sequential recommendation systems (SRS) are widely utilized in various applications, enabling users to efficiently discover pertinent and tailored items by leveraging their historical interacted item sequences [29, 30]. Numerous techniques have been proposed to enhance the efficacy of SRS, including early matrix factorization-based approaches [21] as well as more recent advancements involving RNN-based and Transformer-based models, which have demonstrated substantial improvements in SRS performance [8, 12, 23, 36].

In light of the impressive capabilities exhibited by large language models (LLMs) [19, 26], it becomes reasonable and practical to enhance the performance of conventional sequential recommendation models [8, 12] and tackle challenging recommendation issues by leveraging LLMs, due to the generalization ability and common knowledge within these large models [22]. The utilization of LLMs in SRS can be broadly categorized into two paradigms:

*Both authors contributed equally to the paper.
†Corresponding author.

WWW '24 Companion, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0172-6/24/05.
https://doi.org/10.1145/3589335.3648307

LLM-augmented methods and LLM-centric methods. In the LLM-augmented paradigm, *as depicted in the upper left of Figure 2*, embeddings of items' textual descriptions are extracted from LLMs and considered as features for the items. These features are subsequently integrated with other recommendation models, such as GRU or Transformer [4, 34]. LLM-centric methods, *as sketched in the lower left of Figure 2*, transform items into textual representations and concatenate them into a long text sequence to feed into an LLM. Afterward, an LLM can *either* directly generate an item description as prediction *or* extract sequence features to discover similar items [15, 22]. Considering that text can serve as a versatile modality connecting knowledge from distinct domains, the adoption of such text-based sequential modeling paradigms holds unprecedented promise in addressing previously intricate challenges, such as the cold-start and cross-domain transfer problems in sequential recommendation systems [15, 30].

Despite the promise of LLMs in the realm of sequential recommendations, current works integrating LLMs with SRS exhibit certain limitations. *Firstly*, for the LLM-augmented methods, the textual embeddings obtained through LLMs are typically coarse-grained, which are challenging to capture an item's subtle word-level attributes to represent user preferences [22]. For instance, there may be a small difference between the representations of 'Apple iPhone 15 White 256GB' and 'Apple iPhone 5 White 256GB', making it difficult to distinguish users who prefer 'iPhone 15' over 'iPhone 5'. In other words, there is no guarantee that the extracted textual embeddings from an LLM preserve the fine-grained item text information. *Secondly*, LLM-centric methods encounter difficulties in handling lengthy token sequences and efficiency bottlenecks caused by the notorious computational complexity associated with LLMs [19]. In recent SRS literature, there is a tendency to employ significantly long item sequences, such as thousands of items, to improve the modeling of user preferences [20, 28]. However, each item typically consists of dozens or even hundreds of tokens. Consequently, the total number of tokens within an item sequence could become excessively large, leading to low efficiency and high expenses. In practical industrial applications, there is a high requirement on the response time for user queries, e.g., within ten or dozens of milliseconds [3, 6, 7, 35]. Due to the heavy inference cost of LLMs, existing LLM-centric methods can hardly meet the efficiency standards. Although several works have been dedicated to enhancing the efficiency of LLMs through prompt compression [10, 11], acceleration of attention layers [2, 17, 31], and so on. Despite the efforts, these approaches exhibit compromised performance or limitations in leveraging on-the-shelf LLMs.

In view of the aforementioned limitations, this paper aims to utilize the capability of LLMs in SRS *in an efficient and effective manner*. The main idea of SAID is to learn item embeddings that are accurately aligned with the textual descriptions of items within the embedding space of an LLM, and able to be effectively utilized with readily available lightweight sequential models. To achieve this, SAID evolves a two-stage training scheme. Note that in recommendation scenarios, an item is typically represented by a numerical ID, accompanied by several textual descriptions such as band, category, and so on. *In the first stage*, inspired by LLM-oriented alignment learning [14, 18, 37], SAID employs a projector module to transform an item ID into an embedding and feeds it into an LLM to explicitly elicit the item's textual token sequence from the LLM. In this way, SAID explicitly preserves the fine-grained semantic meaning of an item's textual description into the embedding, i.e., *semantically aligned embedding*. Only the projector undergoes training while the LLM remains fixed, with gradients propagating through it. *In the second stage*, the learned item embeddings will be exploited by a downstream sequential model such as GRU or Transformer to extract the entire sequence's representation for recommendation. In this stage, the sequential model will be trained from scratch and the embeddings learned in the first stage will be fine-tuned. After training, the downstream sequence model and the fine-tuned item embeddings will be adopted in practical inference. Since the LLM is not engaged in the second stage and the downstream sequence model can be lightweight, SAID achieves superior inference efficiency, e.g., less than 5 milliseconds for a single inference and less than 20 milliseconds for an overall online response. Moreover, thanks to the LLM-based alignment learning, the learned item embeddings significantly improve SRS performance over randomly initialized embeddings utilized in previous models [8, 13].

We perform extensive experiments to evaluate the SAID framework from various perspectives. Results on public datasets suggest that SAID outperforms baselines by about 5% to 15% on NDCG@10, and about 3% to 14% on Recall@10. In Alipay's online advertising deployment, SAID achieves 2.98% and 3.07% improvement on click-through rate (CTR) and cost per mille (CPM) respectively over baselines. We summarize our contributions as follows:

- We propose a sequential recommendation framework that learns semantically meaningful item embeddings based on LLMs. Different from randomly initialized item embedding or directly extracting representations from LLMs, the proposed framework preserves fine-grained item textual information in learned embeddings, facilitating the performance of SRS.
- We propose an alignment learning scheme that employs a projector module to learn item embeddings within the embedding space of an LLM. The fixed LLM, alongside lightweight downstream sequential models, simplifies the training and inference process and enhances its practicality in industrial scenarios.
- We conduct experiments on various datasets to verify the effectiveness and efficiency of SAID. We also conduct comprehensive ablation studies and in-depth comparisons to investigate the effect of semantic item embedding learning and other components utilized in the framework.

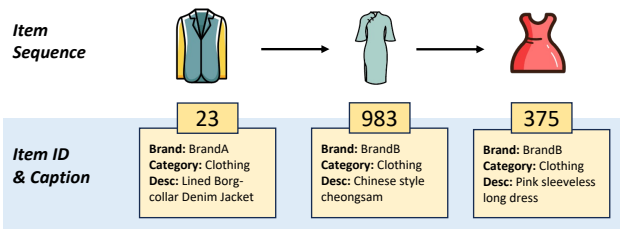

Figure 1: An item sequence with IDs and textual captions, which is fed as inputs into an SRS.

2 METHODOLOGY

In this section, we illustrate the problem formulation of SRS and elaborate on the proposed SAID framework.

2.1 Problem Formulation

In sequential recommendation modeling, a predefined item set $\mathcal{I}$ is considered, and a sequence of items $s = \{i_1, i_2, \cdots, i_n\}$ with length n is observed for a user, in which items are ordered in chronological order. A user will be represented by his/her interacted item sequence. The goal of an SRS is to predict the next item the user will interact with. In this paper, we assume that each item is associated with a unique ID and several textual attributes, i.e., for item i_k, an attribute dictionary $D_k = \{id : v_0, k_1 : v_1, k_2 : v_2, \cdots, k_m : v_m\}$ is attached. Note that in D_k, the key *id* is necessary, while other textual attributes are optional. Moreover, for ease of presentation, we denote τ_i as the concatenated (or flattened) text sequence from a user's all textual attributes, i.e., $\tau_i = [k_1, v_1, k_2, v_2, \cdots, k_m, v_m]$. For example, in Figure 1, a user has interacted with three items. For item 23, it has a textual dictionary indicating its brand, category, and detailed description. To predict the correct next item, we aim to learn a specific item embedding that contains the semantic meaning of its textual representation τ_i and extract the entire sequence's representation based on item embeddings and a sequential model.

2.2 The SAID Framework

2.2.1 Overall framework. The right part of Figure 2 depicts the architecture of SAID. SAID adopts a two-stage training process, namely (1) semantically aligned embedding learning and (2) model-agnostic sequential recommender training. In the first stage, SAID learns to generate an embedding for each item by leveraging the projector module and an on-the-shelf LLM. The size of the learned embedding for each attribute equates to the embedding size of a single token for the specific LLM. In the second stage, the embeddings acquired during the first stage are utilized as initial features of the items, which are then inputted into a downstream model (such as RNN or Transformer) for sequential recommendation. It is important to note that SAID is *model-agnostic* to the specific choices of downstream models employed in the recommendation process, thereby imparting the framework with significant adaptability and flexibility. In the subsequent sections, we will provide detailed elaborations of the two aforementioned stages respectively.

2.2.2 Semantically aligned embedding learning. Let f_θ denote the projector module with parameter set θ, then item i's embedding x_i can be represented as follows:

$$x_i = f_\theta(D_i[id]) \tag{1}$$

As stated above, the objective of training the projector in SAID is to preserve the textual information τ_i within the projected representation x_i, thereby producing *semantically-aligned embeddings* within the embedding space of an LLM. Specifically, as shown in the *Stage 1* of Figure 2, we formulate the projector learning as a conditional text generation task by an LLM, *where the goal is to elicit the text sequence τ_i from the LLM, given the projected embedding x_i as input to it.* For instance, for the item 23 in the *Stage 1* of Figure 2, its projected semantic embedding x_{23} will be fed into an LLM, the LLM is expected to output the first token 'Brand' of its textual description. Subsequently, the x_{23} and word embedding of 'Brand' are taken as inputs and expected to elicit the 'BrandA' from the LLM. The errors from all output tokens of the LLM will be back-propagated to adjust the projector's parameters.

In the following, we mathematically formulate the training process in this stage. We omit the subscript i for ease of presentation and use v_0 to denote an item ID. Let P_ϕ denote an LLM with parameter set ϕ, the optimization objective for the projector is as follows:

$$\arg\max_\theta \log \mathrm{P}_\phi(\tau | f_\theta(v_0)) \tag{2}$$

We follow a regressive approach to generate the text sequence τ, based on which the $\log \mathrm{P}_\phi(\tau | f_\theta(v_0))$ is defined as:

$$\mathcal{L}_{\text{proj}}(P_\phi, f_\theta(v_0), \tau) \triangleq \log \mathrm{P}_\phi(\tau | f_\theta(v_0)) = \sum_{l=1}^{|\tau|} \log \mathrm{P}_\phi(\tau^l | f_\theta(v_0), \tau^{0:l-1}) \tag{3}$$

where $|\tau|$ is the number of tokens contained in τ. The training of f_θ follows a gradient decent strategy, while gradients pass through the fixed LLM P_ϕ, i.e.,

$$\theta \leftarrow \theta - \alpha \nabla_{f(v_0)} \mathcal{L}_{\text{proj}}(P_\phi, f_\theta(v_0), \tau) \cdot \nabla_\theta f(v_0) \tag{4}$$

where α is the learning rate.

In the practical implementation of SAID, the output dimension of $f_\theta(v_0)$ should match the token embedding size of P_ϕ, i.e., $f_\theta(v_0)$ resembles one token for the LLM. We instantiate f_θ as an embedding lookup table, as we find the lookup table instantiation possesses extreme efficiency and achieves considerable performance in our exploratory experiments.

2.2.3 Model-agnostic sequential recommender training. After the completion of projector training in the first stage, we can obtain each item's semantically-aligned embedding x_i. As depicted in the *Stage 2* of Figure 2, these embeddings from the projector can be seamlessly integrated with a downstream sequential model for recommendation. This characteristic of SAID renders it *agnostic* to the specific downstream recommender models employed.

Depending on downstream sequential models, the embedding x_i may also be added with a position embedding p_i to explicitly identify item i's position in the sequence. This practice is particularly relevant when employing Transformer-like models for sequential recommendation. RNN-like models do not require positional embeddings as they inherently incorporate sequential information within model architectures. We denote the overall representation of an item sequence fed into a sequence model as X_s, i.e.,

$$X_s = [x_1, \cdots, x_{n-1}, x_n] \tag{5}$$

Let g_Φ denote a downstream model parameterized by Φ. The sequence representation obtained from g_Φ is denoted as $\mathbf{h}_s$, which signifies the transformed sequence:

$$\mathbf{h}_s = g_\Phi(X_s) \tag{6}$$

It is important to note that we employ x_i as an individual item's representation instead of passing it through the sequence model g_Φ, in order to improve the training and inference efficiency further. We expect the correlation between $\mathbf{h}_s$ and the groundtruth item's representation x_i to be learned automatically.

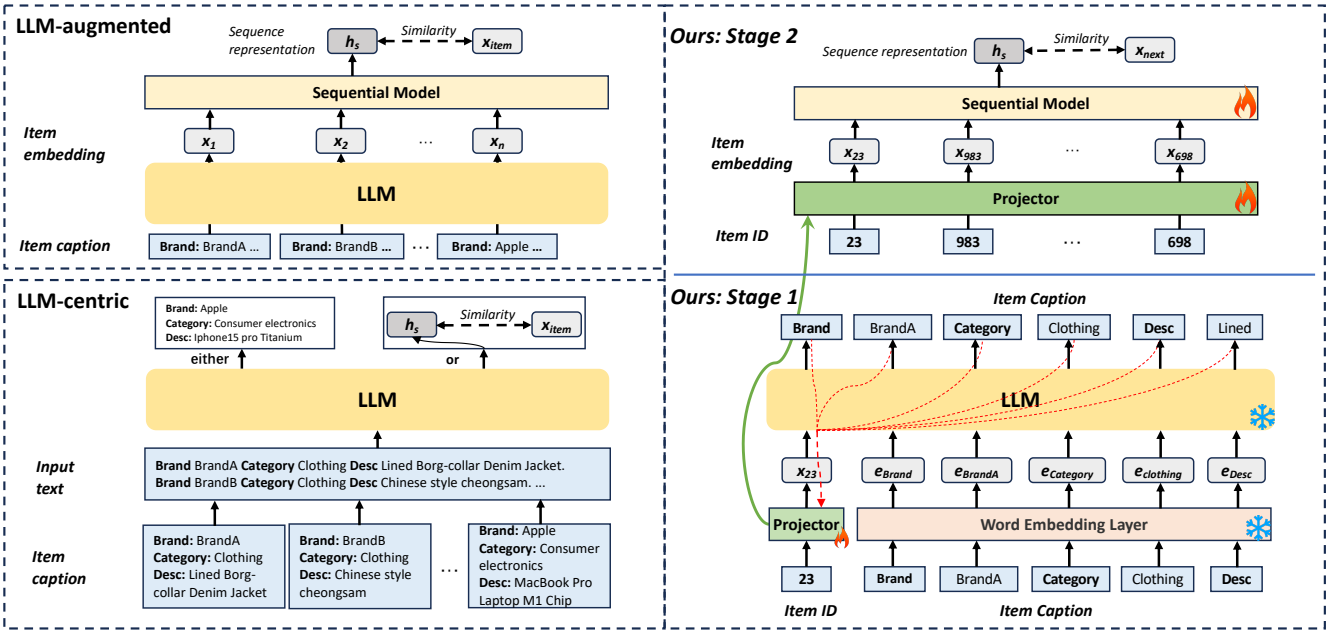

Figure 2: *Upper left*: architecture of LLM-augmented SRS systems. *Lower left*: architecture of LLM-centric SRS systems. *Right*: the proposed SAID framework, which consists of two stages, i.e., (1) semantically-aligned embedding learning and (2) model-agnostic sequential recommender training. Note that in stage 1 of SAID, all item embeddings are learned *in parallel*.

After obtaining the sequence representation $\mathbf{h}_s$, we predict the next item using the cosine similarity between $\mathbf{h}_s$ and representations of all items in $\mathcal{I}$. The similarity score is calculated as follows:

$$r_{s,i} = \frac{\mathbf{h}_s^T x_i}{\|\mathbf{h}_s\| \cdot \|x_i\|} \tag{7}$$

To predict the next item, we select the one with the highest similarity with the sequence as the prediction $\hat{i}_s$:

$$\hat{i}_s = \arg\max_{i \in \mathcal{I}} (r_{s,i}) \tag{8}$$

In the training process of this stage, we employ the item-item contrastive learning objective, where negative items come from the entire item set $\mathcal{I}$. Note that **only the sequential model g_Φ and the projectors f_θ** will be trained, while the LLM $\mathbf{P}_\phi$ does not participate in this stage. The optimization objective adopted for item-item contrastive learning is formulated as follows:

$$\mathcal{L}_{\text{rec}} = -\log \frac{\exp\{\text{sim}(\mathbf{h}_s, x_i^+)\}}{\exp\{\text{sim}(\mathbf{h}_s, x_i^+)\} + \sum_{j \in \mathcal{I}} \exp\{\text{sim}(\mathbf{h}_s, x_j)\}} \tag{9}$$

where x_i^+ is the representation of the groundtruth next item, x_j is the representation of a negative item in $\mathcal{I}$, and sim denotes a specific similarity metric.

3 EXPERIMENTS

3.1 Experimental setup

3.1.1 Datasets. To evaluate the performance of our method, we select six sub-category datasets from the Amazon review[1] datasets, i.e., "Industrial and Scientific", "Musical Instruments", "Arts, Crafts and Sewing", "Office Products", "Video Games", and "Pet Supplies". The statistics of datasets after preprocessing are shown in Table 1.

Table 1: Statistics of the datasets after preprocessing. Avg. n denotes the average length of item sequences.

Datasets	#Users	#Items	Avg. n
Scientific	11,041	5,327	6.96
Instruments	27,530	10,611	8.40
Arts	56,210	22,855	8.76
Office	101,501	27,932	7.87
Games	11,036	15,402	9.08
Pet	47,569	37,970	8.84

For training and testing, we remove the items without title information in meta-data as title will be used for item identification, symbolization and modeling. Then we group each user's interacted items in chronological order for sequence construction. Following previous work [15], we select three item attributes, i.e., title, category, and brand, to construct a caption.

3.1.2 Baselines. Considering that the primary contribution of SAID is the semantically-aligned item embedding learning, we compare it with several commonly employed item embedding initialization schemes in previous works.

i) *Random*, which is the most widely adopted item embedding initialization scheme in the SRS literature.

[1] https://nijianmo.github.io/amazon/index.html

ii) *Last hidden state (LH)*, which extracts an LLM's last hidden state by inputting an item's textual caption into it. We choose LH for comparison since it is an representative LLM-augmented method for SRS. To ensure a fair comparison, we utilize the same LLM model for LH when comparing it with our method.

As for the downstream sequential model, we adopt the two most representative methods, i.e, the **GRU4Rec**[8] with an RNN-based architecture and the **SASRec**[13] with a Transformer-based architecture.

3.1.3 Evaluation Settings. To evaluate the performance of sequential recommendation, we adopt three widely adopted metrics, i.e., NDCG@N, Recall@N, and MRR, where N is set to 10. Moreover, we evaluate the capability of different LLMs with the generation accuracy, which calculates a word-level accuracy when generating an item's caption. To perform train-validate-test splitting, we employ the leave-one-out strategy. The freshest item in an item sequence is reserved for testing, the penultimate item is used for validation, and the remaining items are for training. We rank the ground-truth item of each sequence among all items and report the averaged results over all item sequences.

3.1.4 Implementation Details. By default, we utilize the LLama2-7B[2] in the first stage of SAID for performance comparison. Moreover, we also compare the capability of distinct LLMs, including Bloomz-560M[3], Bloomz-1B7[4], Bloomz-7B[5] and LLama2-13B[6]. All baselines utilize the same sequential models, batch size, and optimizer settings, differing only in the initial item embeddings fed into sequential models. The embedding size and hidden dimension of sequential models are both set to 256. The batch size is set to 256. A single layer is employed for all sequential models. For SASRec, one attention head is adopted. The maximum item sequence length is set to 50. We adopt the AdamW[16] optimizer and a learning rate of 5e-4 along with early stopping, with a patience of 5 epochs.

3.2 Overall Performance

In this section, we illustrate performance comparisons between SAID and baselines on six public datasets in an offline manner. Results are shown in Table 2. All improvements are computed from the averages. We can find that the random baseline performs the worst in most cases, while the LH baseline generally obtains the second best, and SAID achieves the best results in most settings. Specifically, for the GRU4Rec (SASRec) model, SAID improves about 17.2% (14.9%), 27.1% (21.5%), and 15.4% (14.3%) over the random baseline w.r.t the NDCG@10, Recall@10 and MRR, respectively. Compared with the second-best method, for the GRU4Rec (SASRec) model, SAID improves about 7.35% (7.7%), 2.85% (4.6%), and 8.4% (8.2%) w.r.t the NDCG@10, Recall@10 and MRR, respectively. The results verify that the learned semantically aligned item embeddings leveraging LLMs could significantly enhance the SRS performance despite practical downstream models employed. Moreover, by inspecting the comparison with the LH baseline, we can find that SAID has superior improvement on the NDCG@10 metric than the Recall@10. Since the NDCG@10 is more *fine-grained* than Recall@10 (NDCG considers subtle item ranking information while Recall ignores it), the superior improvement over NDCG@10 suggests that SAID indeed learns more subtle item representations than directly extracting the hidden states from an LLM.

[2]https://huggingface.co/meta-llama/Llama-2-7b
[3]https://huggingface.co/bigscience/bloomz-560m
[4]https://huggingface.co/bigscience/bloomz-1b7
[5]https://huggingface.co/bigscience/bloomz-7b1
[6]https://huggingface.co/meta-llama/Llama-2-13b

3.3 Further Analysis

Effect of different LLMs. We first investigate the capability of different LLMs on semantically-aligned embedding learning. The results are shown in Figure3 with the Scientific dataset. The y-axis indicates LLMs' token-level generation accuracy. It is evident that as the scale of LLMs increases, the generation accuracy also improves. While both Llama2-7B and Llama2-13B demonstrate near-perfect generation performance, the larger Llama2-13B exhibits faster convergence speed in the initial stage of training.

Based on the findings in Figure 3, we choose the item embeddings trained from the top three LLMs for further comparison of SRS performance, as depicted in Figure 4. The results indicate that Llama2-7B consistently outperforms Bloomz-7B in terms of both metrics and downstream models. Moreover, Llama2-13B demonstrates slightly better performance than Llama2-7B. Considering these outcomes, we select Llama2-7B as the default LLM in SAID, maintaining a balance between efficiency and efficacy.

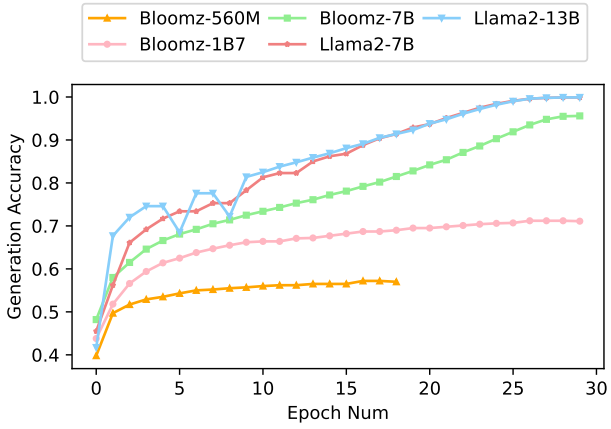

Figure 3: Generation accuracy of LLMs in the semantically aligned embedding learning stage on the Scientific dataset.

Effects of embedding alignment quality. In this part, we investigate the correlation between the quality of embedding alignment and the performance of downstream recommendation. We employ the GRU4Rec backbone for illustration. The results are illustrated in Figure 5. The x-axis indicates the generation accuracy of the adopted LLM during stage 1 of SAID. The y-axis signals the performance of downstream models. Note that the distribution of points along the x-axis is not uniform due to the practical generation accuracy of alignment, which cannot be guaranteed uniformly distributed in advance. The grey line represents the performance of GRU4Rec with randomly initialized item embeddings, and the red line suggests the performance of GRU4Rec with item embeddings learned in stage 1 of our SAID. From Figure 5, we can conclude

Table 2: Performance comparison of different methods. The best results of all methods are highlighted in bold, and the best performance of baselines is underlined. The improvement is calculated over the second-best result.

Dataset	Metrics	GRU4Rec				SASRec			
		Random	LH	SAID	Improv.	Random	LH	SAID	Improv.
Scientific	NDCG@10	0.0896	0.0958	**0.1021**	6.6%	0.0869	0.0992	**0.1050**	5.8%
	Recall@10	0.1037	0.1317	**0.1321**	0.3%	0.1088	0.1313	**0.1353**	3.0%
	MRR	0.0877	0.0913	**0.0977**	7.0%	0.0825	0.0945	**0.1005**	6.3%
Instruments	NDCG@10	0.0760	0.0774	**0.0862**	11.4%	0.0768	0.0837	**0.0928**	10.9%
	Recall@10	0.0910	0.1085	**0.1122**	3.4%	0.0920	0.1158	**0.1211**	4.6%
	MRR	0.0741	0.0741	**0.0832**	12.3%	0.0750	0.0802	**0.0896**	11.7%
Arts	NDCG@10	0.0878	0.1057	**0.1180**	11.6%	0.1017	0.0995	**0.1090**	7.2%
	Recall@10	0.0978	**0.1441**	0.1413	-	0.1302	0.1433	**0.1487**	3.8%
	MRR	0.0866	0.0993	**0.1145**	15.3%	0.0951	0.0910	**0.1017**	6.9%
Office	NDCG@10	0.1127	0.1136	**0.1202**	5.8%	0.1084	0.1134	**0.1208**	6.5%
	Recall@10	0.1285	0.1374	**0.1440**	4.8%	0.1265	0.1377	**0.1450**	5.3%
	MRR	0.1097	0.1098	**0.1160**	5.6%	0.1047	0.1092	**0.1165**	6.7%
Games	NDCG@10	0.0641	0.0748	**0.0785**	4.9%	0.0673	0.0752	**0.0812**	8.0%
	Recall@10	0.0877	**0.1221**	0.1173	-	0.0936	**0.1221**	0.1204	-
	MRR	0.0617	0.0694	**0.0741**	6.8%	0.0647	0.0696	**0.0770**	10.6%
Pet	NDCG@10	0.0854	0.0925	**0.0960**	3.8%	0.0878	0.0881	**0.0951**	7.9%
	Recall@10	0.0945	0.1120	**0.1152**	2.9%	0.0978	0.1062	**0.1129**	6.3%
	MRR	0.0843	0.0902	**0.0934**	3.5%	0.0866	0.0859	**0.0929**	7.3%

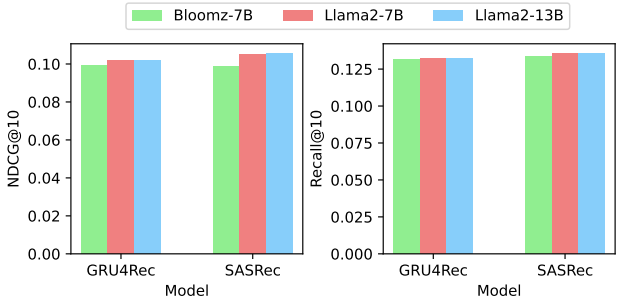

Figure 4: Performance of downstream models using items' semantic embeddings learned from the top three LLMs (ranked by their generation capability) on the Scientific dataset.

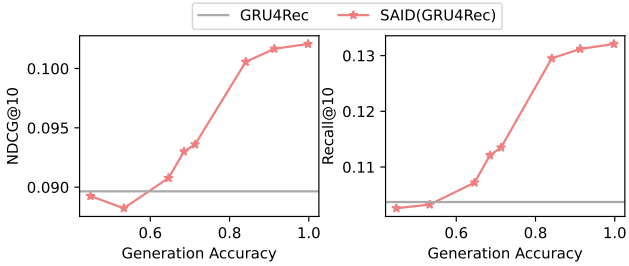

Figure 5: Performance of downstream models under different generation accuracy of the adopted LLM on the Scientific dataset.

that as the generation accuracy increases, the performance of the downstream model also improves. This consistent improvement is observed in both the NGCG@10 and Recall@10 metrics, confirming that item embedding alignment plays a significant role in enhancing the SRS performance.

Effects of w/o freezing item embeddings. In this part, we investigate the impact of freezing item embeddings during the training of downstream models. The results, depicted in Figure 6 using the Scientific dataset and GRU4Rec as the downstream model, demonstrate that the without-freezing scheme outperforms the with-freezing counterpart in all cases. This outcome is expected since the without-freezing scheme can be considered as a further fine-tuning over the item embeddings. However, even though it is not as effective as the without-freezing scheme, the with-freezing scheme still achieves better performance than the random baseline. This finding confirms the effectiveness of the semantically-aligned embeddings in initializing item embeddings.

Efficiency of SAID. To assess the efficiency of different sequence recommendation paradigms, we conduct comparisons using a machine equipped with an A100 80G GPU. For our SAID with both sequence model backbones, we adopt a sequence length of 50. Further, we test a text-based sequence recommendation method with

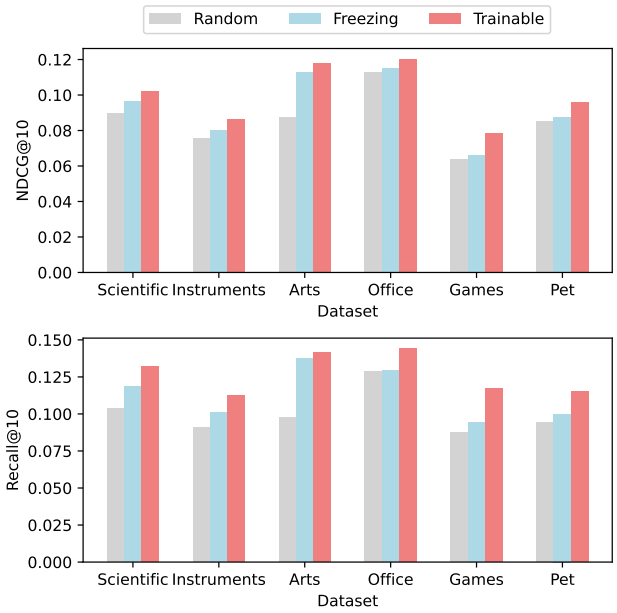

Figure 6: Performance comparison among the without freezing scheme, with freezing scheme, and the random baseline. The experiment is conducted on the Scientific dataset.

Table 3: Inference efficiency between our SAID with both sequential models and the LLama2-7B-based SRS model.

Method	Inference Time Per Sample
SAID (GRU4Rec)	2.37ms
SAID (SASRec)	2.34ms
LLM (LLama2-7B)	103.21ms

LLama2-7B, setting a maximum token length of 1024. We utilize Pytorch and conduct inference with half-precision. Table 3 displays the experimental outcomes. We can find that SAID demonstrates significant efficiency superiority, requiring 2.2% of the time taken by the LLM method. Additionally, in Alipay's online advertising deployment, SAID could achieve an overall query response time of less than 20 milliseconds. In industrial scenarios, it is often necessary for the response time to be below 20 milliseconds, a standard that LLM-centric methods typically struggle to meet.

3.4 Visualization

Visualization of learned item embeddings. We visualize the learned item embeddings on six datasets using t-SNE [27] in Figure 7. The embeddings clearly exhibit clustering within each dataset and distinguishable ability between different datasets. Given the distinction of item textual descriptions from different datasets, the results in Figure 7 provide evidence that the item embeddings successfully captured the semantic meaning of their textual descriptions.

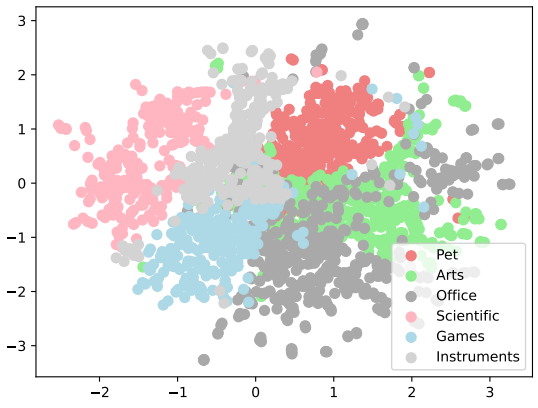

Figure 7: Visualization of learned item embedding from six datasets using t-SNE with the scikit-learn's default settings.

Visualization of items with similar embeddings. To examine the caption of items that share similar learned embeddings, we perform clustering on item embeddings extracted from the Scientific dataset. The clustering is conducted using the K-means algorithm with 100 clusters and the cosine similarity metric. Afterward, we select two clusters and randomly select two items within each cluster. The ground-truth text descriptions of these items are listed in Table 4. Table 4 reveals the presence of intra-cluster similarity and inter-cluster distinction, e.g., cluster A is about scientific measurement and cluster B relates to industrial adhesives. This result indicates that the learned item embeddings are capable of capturing the semantic meaning of their original textual descriptions, which is consistent with results in Figure 7.

Visualization of generated item descriptions. Although a capable LLM could achieve virtually perfect (100%) generation accuracy as indicated in Figure 3, there still exist erroneous generations in some cases. To investigate the generation results of concrete LLMs vividly, we illustrate LLM-generated texts of two items that acquire imperfect generation accuracy. The results are listed in Table 5. We highlight the different words in green in ground truth texts and red in generated texts. We can find that erroneous words are rare and generally have a minor influence on the overall semantic meaning. This finding confirms the effectiveness of the alignment learning with a proper LLM in our SAID.

4 ONLINE DEPLOYMENT

We deployed the proposed SAID in Alipay's advertising system[7] and conducted an A/B test for one week. The online experiment demonstrates that the SAID framework significantly improved the click-through rate (CTR) by 2.98% and the cost per mille (CPM) by 3.07% compared with the LH baseline. The online response time is less than 20 milliseconds. This outcome verifies the efficacy of SAID in read-world recommendation scenarios. In addition, it has

[7]Alipay is one of the largest mobile payment platforms in the world.

Table 4: Item descriptions from same clusters based on their learned embeddings.

Cluster	Item Caption
A	title: Grizzly G9650 9-Inch by 12-Inch by 3-Inch Granite Surface Plate, 2 Ledges, brand: Grizzly, category: Industrial & Scientific Test, Measure & Inspect Calibration Dimensional Calibration Surface Plates
	title: Anytime Tools DIAL TEST INDICATOR 2-way Horizontal Extended Range +/- 0.060"/0.0005", brand: Anytime Tools, category: Industrial & Scientific Test, Measure & Inspect Dimensional Measurement Calipers Dial Calipers
B	title: Loctite Clear Silicone Waterproof Sealant 2.7-Ounce Tube (908570), brand: Loctite, category: Industrial & Scientific Tapes, Adhesives & Sealants Industrial Sealants
	title: Rectorseal 97606 4-Ounce Ep-400 Epoxy Putty, brand: Rectorseal, category: Industrial & Scientific Tapes, Adhesives & Sealants Epoxy Adhesives Tile Epoxy Adhesives

Table 5: Comparison between groundtruth captions and generated captions for items with imperfect generation accuracy. The discrepancies are highlighted.

Case	Ground-truth Caption	Generated Caption
1	title: Pet Media Feathered Phonics The Easy Way To Teach Your Bird To Speak Volume 1: 96 Words and Phrases, brand: Pet Media, category: Pet Supplies Top Selection from Amazon-Pets	title: Petron Theather Ph Phonics The Easy Way To Teach Your Bird To Speak Volume 1: 96 Words And Phrases, brand: Pet Media, category: Pet Supplies Top Selection from Amazon-Pets
2	title: Knock Knock All Out Of Pad (Red), brand: Knock Knock, category: Office Products Office & School Supplies Paper Notebooks & Writing Pads	title: Knock Knock All Out Of Office (Red), brand: Knock Knock, category: Office Products Office & School Supplies Paper Notebooks & Writing Pads

gone into production in 2023 to boost the CPM and conversion rate (CVR) in 10+ advertising display channels.

5 RELATED WORK

5.1 Sequential Recommendation

Early sequential recommendation models mainly focus on modeling users' sequential behaviors as a Markov chain, in which an item-to-item transition matrix is learned for next item prediction [21]. Attribute to the extraordinary capability of deep learning for complicated item sequence pattern modeling, a surge of deep neural networks-based methods are proposed to enhance the performance of sequential recommendation models, e.g., RNN-based methods [8] and CNN-based methods [24] are elaborated effective for sequential recommendation. However, these solutions often fail to capture long-term dependency between arbitrary items in a sequence. In light of this, Transformer-based solutions, e.g., SASRec [12], BERT4Rec [23], are widely applied and achieved promising results in sequential recommendation. Moreover, some sophisticated methods, e.g., contrastive learning [1, 32] are also adopted to relieve the data sparsity in sequential recommendation scenarios.

5.2 LLM based recommendation

Motivated by the notable achievements of LLMs, several works try to employ these large models in recommendations [5, 9, 15, 25, 33]. MoRec [33] investigates the viability of utilizing textual representations, derived from language models, as an alternative to ID embedding for recommendations. ZESRec [5] explores the zero-short capability of a sequential recommendation model, focusing on a specific domain. They employ a pre-trained language model to generate textual-based representations of items based on their associated information. Subsequently, the derived representations are utilized for the training of sequential models, which will be deployed for personalized recommendations in diverse domains. Following ZESRec, UniSRec [9] leverages multi-domain data to pretrain the sequential model, and subsequently employs domain-specific data for fine-tuning. Furthermore, Recformer [15] pre-trains and fine-tunes a language model in a holistic approach for item text encoding and sequential recommendation. However, the integration of LLMs and textual representations in these studies results in prolonged token sequences that significantly impede efficiency. Besides, these works overlook the explicit alignment between item embeddings and the semantic meaning of corresponding textual descriptions, thereby offering no assurance that the textual information of the items could be accurately preserved in the generated embeddings.

6 CONCLUSION

In this paper, we propose SAID, a framework that utilizes LLMs to project item IDs into semantically meaningful embeddings within the embedding space of these LLMs. The learned semantic embeddings are leveraged by downstream sequential models for concrete recommendation tasks. In this way, SAID efficiently diminishes the lengthy token sequence challenge when directly utilizing LLMs in recommendation and also preserves superior performance by leveraging the capability of LLMs. Experiments conducted on six public datasets and Alipay's online advertising deployment justify the efficiency and efficacy of SAID. Additionally, we observe that the choice of LLM has a substantial impact on the quality of learned item embeddings, consequently influencing the performance of downstream models. In practical industrial applications, items are always combined with multi-modal features. In the future, we hope to unify the integration of these multi-modal data in our framework for further improvement.

REFERENCES

[1] Yongjun Chen, Zhiwei Liu, Jia Li, Julian J. McAuley, and Caiming Xiong. 2022. Intent Contrastive Learning for Sequential Recommendation. In *Proceedings of the ACM Web Conference 2022*. 2172–2182.

[2] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023. LongLoRA: Efficient Fine-tuning of Long-Context Large Language Models. *arXiv preprint arXiv:2309.12307* (2023).

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

[4] Hao Ding, Yifei Ma, Anoop Deoras, Yuyang Wang, and Hao Wang. 2021. Zero-Shot Recommender Systems. arXiv:2105.08318 [cs.LG]

[5] Hao Ding, Yifei Ma, Anoop Deoras, Yuyang Wang, and Hao Wang. 2021. Zero-shot recommender systems. *arXiv preprint arXiv:2105.08318* (2021).

[6] Udit Gupta, Samuel Hsia, Vikram Saraph, Xiaodong Wang, Brandon Reagen, Gu-Yeon Wei, Hsien-Hsin S Lee, David Brooks, and Carole-Jean Wu. 2020. Deeprecsys: A system for optimizing end-to-end at-scale neural recommendation inference. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 982–995.

[7] Udit Gupta, Carole-Jean Wu, Xiaodong Wang, Maxim Naumov, Brandon Reagen, David Brooks, Bradford Cottel, Kim Hazelwood, Mark Hempstead, Bill Jia, et al. 2020. The architectural implications of facebook's dnn-based personalized recommendation. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 488–501.

[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[9] 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 *KDD*. arXiv:2206.05941 [cs.IR]

[10] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LLMLingua: Compressing Prompts for Accelerated Inference of Large Language Models. *arXiv preprint arXiv:2310.05736* (2023).

[11] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression. *arXiv preprint arXiv:2310.06839* (2023).

[12] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *Proceedings of IEEE International Conference on Data Mining*. arXiv:1808.09781 [cs.IR]

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

[14] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597* (2023).

[15] 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. *arXiv preprint arXiv:2305.13731* (2023).

[16] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[17] Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark Attention: Random-Access Infinite Context Length for Transformers. *arXiv preprint arXiv:2305.16300* (2023).

[18] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, et al. 2023. AnyMAL: An Efficient and Scalable Any-Modality Augmented Language Model. *arXiv preprint arXiv:2309.16058* (2023).

[19] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[20] 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.

[21] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. 811–820.

[22] Xiaoming Shi, Siqiao Xue, Kangrui Wang, Fan Zhou, James Y Zhang, Jun Zhou, Chenhao Tan, and Hongyuan Mei. 2023. Language Models Can Improve Event Prediction by Few-Shot Abductive Reasoning. *arXiv preprint arXiv:2305.16646* (2023).

[23] 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* (Beijing, China) *(CIKM '19)*. ACM, New York, NY, USA, 1441–1450. https://doi.org/10.1145/3357384.3357895

[24] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 565–573.

[25] Zuoli Tang, Lin Wang, Lixin Zou, Xiaolu Zhang, Jun Zhou, and Chenliang Li. 2023. Towards Multi-Interest Pre-training with Sparse Capsule Network. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 311–320.

[26] 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).

[27] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[28] Jiachun Wang, Fajie Yuan, Jian Chen, Qingyao Wu, Min Yang, Yang Sun, and Guoxiao Zhang. 2021. Stackrec: Efficient training of very deep sequential recommender models by iterative stacking. In *Proceedings of the 44th International ACM SIGIR conference on Research and Development in Information Retrieval*. 357–366.

[29] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/883

[30] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A Survey on Large Language Models for Recommendation. *arXiv preprint arXiv:2305.19860* (2023).

[31] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient Streaming Language Models with Attention Sinks. *arXiv preprint arXiv:2309.17453* (2023).

[32] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive Learning for Sequential Recommendation. In *38th IEEE International Conference on Data Engineering*. 1259–1273.

[33] Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. *arXiv preprint arXiv:2303.13835* (2023).

[34] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation.. In *IJCAI*. 4320–4326.

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

[36] 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.

[37] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592* (2023).