# A Bi-Step Grounding Paradigm for Large Language Models in Recommendation Systems

KEQIN BAO*, University of Science and Technology of China, China

JIZHI ZHANG*, University of Science and Technology of China, China

WENJIE WANG, National University of Singapore, Singapore

YANG ZHANG, University of Science and Technology of China, China

ZHENGYI YANG, University of Science and Technology of China, China

YANCHENG LUO, University of Science and Technology of China, China

FULI FENG, University of Science and Technology of China, China

XIANGNAN HE, University of Science and Technology of China, China

QI TIAN, Huawei Inc., China

As the focus on Large Language Models (LLMs) in the field of recommendation intensifies, the optimization of LLMs for recommendation purposes (referred to as LLM4Rec) assumes a crucial role in augmenting their effectiveness in providing recommendations. However, existing approaches for LLM4Rec often assess performance using restricted sets of candidates, which may not accurately reflect the models' overall ranking capabilities. In this paper, our objective is to investigate the comprehensive ranking capacity of LLMs and propose a two-step grounding framework known as BIGRec (Bi-step Grounding Paradigm for Recommendation). It initially grounds LLMs to the recommendation space by fine-tuning them to generate meaningful tokens for items and subsequently identifies appropriate actual items that correspond to the generated tokens. By conducting extensive experiments on two datasets, we substantiate the superior performance, capacity for handling few-shot scenarios, and versatility across multiple domains exhibited by BIGRec. Furthermore, we observe that the marginal benefits derived from increasing the quantity of training samples are modest for BIGRec, implying that LLMs possess the limited capability to assimilate statistical information, such as popularity and collaborative filtering, due to their robust semantic priors. These findings also underline the efficacy of integrating diverse statistical information into the LLM4Rec framework, thereby pointing towards a potential avenue for future research. Our code and data are available at https://github.com/SAI990323/Grounding4Rec.

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

Additional Key Words and Phrases: Large Language Models, Grounding, Sequential Recommendation

Authors' addresses: Keqin Bao*, baokq@mail.ustc.edu.cn, University of Science and Technology of China, China; Jizhi Zhang*, cdzhangjizhi@mail.ustc.edu.cn, University of Science and Technology of China, China; Wenjie Wang, wenjiewang96@gmail.com, National University of Singapore, Singapore; Yang Zhang, zy2015@mail.ustc.edu.cn, University of Science and Technology of China, China; Zhengyi Yang, yangzhy@mail.ustc.edu.cn, University of Science and Technology of China, China; Yancheng Luo, luoyanchen@mail.ustc.edu.cn, University of Science and Technology of China, China; Fuli Feng, fulifeng93@gmail.com, University of Science and Technology of China, China; Xiangnan He, xiangnanhe@gmail.com, University of Science and Technology of China, China; Qi Tian, tian.qi1@huawei.com, Huawei Inc., China.
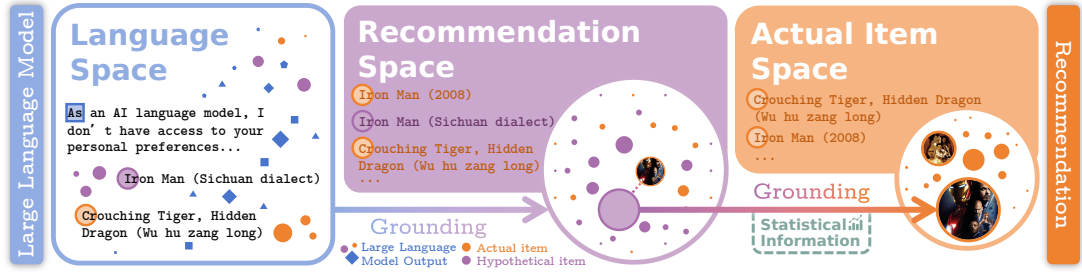
Fig. 1. Illustration of the BIGRec paradigm. During the first step, we ground the language space to recommendation space, which enables the model to generate token sequences of potential items including both actual and hypothetical items. During the second step, we ground the recommendation space to actual item space to provide users with suggestions for real-world items. In the second step, we can easily incorporate statistical information (*e.g.,* popularity and collaborative information) to obtain better recommendations.

## 1 INTRODUCTION

Large Language Models (LLMs) have garnered significant success across various domains (such as Computer Vision [47] and Robotics [13]) due to their enormous context comprehension and generation capabilities [8, 59]. Surpassing traditional language models such as BERT [10] and GPT2 [35], LLMs encode more knowledge, possess mightier reasoning abilities, and can be smoothly adapted to a new task through in-context learning with a few examples [3, 26, 39]. In light of this, exploring utilizing LLMs for recommendation (LLM4Rec) is evolving vigorously [2, 14, 46, 54]. Due to the lack of recommendation training during the pre-training stage of LLM [3], tuning LLMs plays a crucial role in helping the LLM achieve better recommendation performance.

Numerous studies currently enhance the recommendation performance of LLMs through instruction-based fine-tuning techniques and utilizing recommendation data, achieving promising results [3, 55]. When evaluating the effectiveness of these fine-tuning methods, they often only conduct recommendation predictions on a limited candidate set (*e.g.,* click-through rate prediction [3] or negative sampling setting [55]), which do not take into account the model's overall ability to rank globally. Interestingly, there is a study that proposes the avoidance of evaluating recommendation models through sampling, which may lead to a poor indicator of the true performance of recommender systems [25]. Therefore, we aspire to investigate and explore the capabilities of LLMs in the realm of all-rank recommendation sorting.

To fully leverage the recommendation capabilities of LLMs in an all-rank scenario, the LLM-based recommendation model must satisfy three requirements. 1) It must be efficient to rank a considerable number of items for a user. 2) It should provide enough flexibility to generate a meaningful item to utilize the generation and comprehension abilities of LLMs. 3) It is important to consider recommending an actual item that exists in the real world while also incorporating relevant statistical information, such as popularity and collaborative information.

In light of this, we should first make the LLM consistent between the instruction tuning phase and the pre-training phase of LLMs in a generative manner. Simultaneously, we should make the recommendation compatible with the

creation of meaningful items that do not actually exist in the real world (*e.g.,* Iron Man (Sichuan dialect)[1]), as LLMs are capable of generating creative content. Lastly, it is imperative to consider statistical information from past user behaviors when making recommendations to enhance their usefulness.

To achieve this goal, we formulate a <u>bi</u>-step <u>g</u>rounding paradigm for <u>rec</u>ommendation (BIGRec) with "language space" → "recommendation space" → "actual item space" as shown in Figure 1. The language space refers to the set of all possible sequences that can be generated by the LLM; the recommendation space is a subset of the language space that includes descriptions of various items satisfying user preferences, which encompasses both actual and hypothetical items. In the first step, we ground LLMs to the recommendation space by fine-tuning them to generate meaningful tokens for item descriptions (see Table 1). The second step involves identifying the most suitable actual items that match the generated tokens, utilizing their latent representations obtained from the LLM. This step also offers flexibility for incorporating diverse statistical information required for recommendations, such as weighting the distance of representations according to item popularity.

We conduct extensive experiments on two real-world datasets to investigate the extraordinary ability of BIGRec and the influence of popularity and collaborative information. The results first demonstrate the strong few-shot and multi-domain abilities of BIGRec, which largely outperforms traditional recommendation models and existing LLM4Rec methods. Remarkably, BIGRec can outperform most of the traditional models trained with 100 or even 1,000 times more samples. Furthermore, we observe that the benefits obtained from scaling up training samples are relatively modest for BIGRec when compared to traditional models. Considering that more training samples benefit traditional models with richer statistical information, we postulate that LLMs may take limited statistical information (*e.g.,* popularity and collaborative information) from training samples due to their powerful semantic priors. We further validate the effectiveness of incorporating popularity and collaborative information, showing the potential generalization of the paradigm for future works.

In conclusion, our contributions are as follows:

- We study LLM4Rec in an all-rank setting and introduce a bi-step grounding paradigm, which leverages the comprehension and generation ability of LLMs in an efficient and effective manner with the support of seamlessly integrating statistical information.
- We validate the effectiveness of BIGRec, which shows extraordinary ability for few-shot and cross-domain recommendation; and reveal the influence of scaling up the training data.
- We integrate popularity and collaborative information into BIGRec and reveal the benefit of that statistical information in LLM4Rec, showing potential future directions.

## 2 RELATED WORK

***LLM-based Recommendation.*** Researchers in the field primarily investigate LLM-based recommendations from two perspectives. Firstly, they aim to leverage the well-established in-context learning (ICL) approach within the natural language processing (NLP) community [6, 16, 30]. LLMs like ChatGPT can learn the recommendation task through autoregression by providing appropriate instructions and a few examples. However, previous studies have explored the effectiveness of ICL-based recommendations and have observed that the performance of these methods is often limited, which is attributed to the constraints imposed on the size of input for LLMs and the lack of recommendation knowledge [3, 30]. To overcome this problem, the other perspective believes that we should apply the LLM to the

---

[1]The Sichuan dialect is the prevalent vernacular in a region of China. Although the movie may not be included in the recommended list or might not even exist in reality, it could still be considered appropriate for individuals hailing from Sichuan.

recommendation task by fine-tuning the LLM due to the lack of relevant recommendation data in the pre-training phase of the LLM [3, 28]. Among them, [28] uses the LLM to get the item embedding which severs as an item representation and is fed into the traditional recommendation models (e.g. SASRec [23]). Nevertheless, this method still relies on traditional models making it difficult to take advantage of the generative ability of LLMs. In addition, two existing studies, TALLRec [3] and InstructRec [55], employ recommendation data in an instruction-tuning phase to improve the recommendation capability of the LLM. Yet, albeit they do consider the use of LLM from a generative standpoint, their evaluation is limited to the CTR scenario or conducted through negative sampling and does not explore the LLM's ability to rank all items.

*Sequential Recommendation*. The recommendation paradigm that is closest to our experimental setup is the sequential recommendation, which requires the recommender system to predict the next item a user is likely to like based on the user's historical sequence of interactions [15, 42]. Previous work has separately endeavored RNN-based models [9, 12, 18], CNN-based models [38, 49, 52], and attention-based models [23, 48, 56] to model user behavior, garnering respectable outcomes. Based on these methods, an increasingly large number of works are focusing on enhancing model performance through pretraining [32, 51], data augmentation [33, 34], debiasing techniques [11, 43], distributionally robust optimization [45, 50] and other analogous methodologies. However, constrained by conventional ID-based recommender systems, such methods lack generalization ability as well as the rapidity to adapt to novel scenarios. Over recent years, there has been an increasing awareness of the role semantics plays in the recommendation, with efforts now being made to leverage language models to aid the recommendation process [19, 27]. However, these methods, either persist with based language models employing encoders to extract features, which requires enormous data for pre-training, or utilize word overlapping to retrieve items, which lacks utilizing semantic information and is easy to be influenced by noise.

*Grounding in LLM*. Currently, in the research of LLMs, we can mainly consider the concept of grounding from two perspectives: modality grounding and affordance grouding [44]. The former is in order to ground linguistic modalities to the knowledge of other modalities, such as images, audio, or other modalities, through this approach, one can enable LLMs to capture and process richer information from the real world [4]. In the line of the modality grounding [22, 29, 53], individuals endeavor at introducing information of other modalities during both the training and inference phases of LLMs, leading to the capability of processing multimodal inputs, as exemplified by Vicuna [7, 60], MiniGPT4 [61]. However due to the requirements of a large amount of data and resources [44], within this paper, we have not accorded precedence to this grounding strategy but leave it as a future work. The latter aims at grounding the LLM to a specific contextual scenario, ensuring that the results generated by the model are associated with the task rather than detached from the scenario [1]. In practice, people achieve this goal via instruction-tuning on domain-specific data or utilizing customized prompts to steer the LLM [44]. Where in the recommender domain, owing to its gap with the generative task itself – we necessitate recommending an item that truly exists, hence we apply a step grounded in the generation output to the real world to fulfill the recommendation.

## 3 BIGREC

In this section, we introduce an elementary implementation of BIGRec with two grounding steps.

### 3.1 Preliminary

*Defination*. To better understand the grounding paradigm, we first give definitions of the following key phrases:

Table 1. Example of the instruction-tuning data for the step of grounding to the space.

| Instruction Input | |
| --- | --- |
| **Instruction:** | Given ten movies that the user watched recently, please recommend a new movie that the user likes to the user. |
| **Input:** | The user has watched the following movies before: "Traffic (2000)", "Ocean's Eleven (2001)", ... "Fargo (1996)" |
| Instruction Output | |
| **Output:** | "Crouching Tiger, Hidden Dragon (Wu hu zang long) (2000)" |

- **Language Space.** This space prior to the grounding paradigm encompasses all conceivable language sequences that an LLM could generate, such as the statement, "As an AI language model, I don't have access to your personal preferences …". It is not feasible to utilize this space directly for generating recommendations due to its vast and varied nature.

- **Recommendation Space.** This is a sub-space within the language space that includes a wide range of entities that fulfill the user's preferences. These entities can represent both actual and imaginary items in a particular domain. However, it is important to note that recommending purely imaginary entities may not be appropriate. For instance, suggesting "Iron Man (Sichuan dialect)" as a recommendation would be infeasible.

- **Actual Item Space.** The actual item space contains only the actual items in the recommendation space. Recommending items from this actual item space is necessary. For example, in the context of movie recommendations, the recommended items must be selected from the available movies on the platform.

To fine-tune LLMs for recommendation, we propose the BIGRec paradigm with two grounding steps. Firstly, we ground the output of LLMs from the language space to the recommendation space for a specific recommendation task. Secondly, we ground it from the recommendation space to the actual item space, enabling the recommendations of actual items to users. To demonstrate the capabilities of our paradigm, we present a simple implementation, which illustrates the potential and possibilities of this BIGRec paradigm.

### 3.2 Implementation

In this subsection, we describe how we implement the BIGRec paradigm for recommendations.

*3.2.1 Step 1: Grounding Language Space to Recommendation Space.* In accordance with the methodology proposed in [3], we perform an instruction-tuning phase on the alpaca self-instruct data [39] using LLaMA [40]. Afterward, we conduct a recommendation-specific instruction-tuning to restrict the output of LLMs from the language space to the recommendation space. As demonstrated in Table 1, we fine-tune LLMs in a generative manner: given a user's past interactions with items, we ask LLMs to generate a new item as the recommendation to the user. By fine-tuning with such data, we limit the LLMs' output to the designated recommendation space as instructed. However, due to the creativity of the LLM, it is hard to ensure that the output of the LLMs will correspond to an actual item that exists in the real world. Therefore, it is essential to ground the output of LLMs to the actual item space.

*3.2.2 Step 2: Grounding Recommendation Space to Actual ItemsSpace.* In this subsection, we elaborate on how to anchor the recommender space to the actual item space. Firstly, we align the output of LLMs with real-world items based on the representations of LLMs to implement a vanilla version of BIGRec. Then, we introduce statistical information (*e.g.,* popularity and collaborative information) to accurately locate the actual items for recommendations. Specifically, we extract the latent representation of the generated tokens and the embedding of the actual items. Thereafter, we rank these actual items by calculating the L2 distance between their embeddings. The L2 distance is obtained as follows:

$$D_i = ||\mathbf{emb}_i - \mathbf{oracle}||_2, \tag{1}$$

where $\mathbf{emb}_i$ denotes the embedding of the $i$-th item and $\mathbf{oracle}$ denotes the embedding of the outputs generated by the LLM.

***Injection of Statistical Information.*** We then introduce how we incorporate popularity information and collaborative information into the grounding step. To inject popularity, we follow the idea in PDA [57] and reweight the L2 distance in Eq. (1) by popularity. In detail, we first calculate the popularity factor of each item from the following equation:

$$\begin{cases} C_i = \dfrac{\mathcal{N}^i}{\sum_{j \in I} \mathcal{N}^j}, \\ P_i = \dfrac{C_i - \min_{j \in I}\{C_j\}}{\max_{j \in I}\{C_j\} - \min_{j \in I}\{C_j\}}, \end{cases} \tag{2}$$

where $\mathcal{N}$ denotes the set of user-item interactions in the training data, $\mathcal{N}^j$ denotes the number of observed interactions for item $j$ in $\mathcal{N}$ and $I$ denotes all items, $C_i$ represents the popularity of the $i$-th item, and $P_i$ is the normalized value of $C_i$.

We then adjust the L2 distance in Eq. (1) by popularity:

$$\begin{cases} \hat{D}_i = \dfrac{D_i - \min_{j \in I}\{D_j\}}{\max_{j \in I}\{D_j\} - \min_{j \in I}\{D_j\}}, \\ \widetilde{D}_i = \dfrac{\hat{D}_i}{(1 + P_i)^\gamma}, \end{cases} \tag{3}$$

where $D_i$ denotes the L2 distance between the embedding of $i$-th item and the embedding of the generated outputs by LLMs, $\hat{D}_i$ is the normalized $D_i$, and $\widetilde{D}_i$ reweights $\hat{D}_i$ by using popularity. To reweight $\hat{D}_i$, we utilize the inverse popularity and introduce a hyperparameter $\gamma$ to regulate the influence of popularity. By putting popularity in the denominator, a popular item will be assigned a smaller L2 distance and a higher rank. This slightly differs from the original implementation in PDA, which directly reweights prediction scores instead of L2 distance[2].

Different from popularity, it is non-trivial to use statistical methods to quantify collaborative information. Considering that traditional recommendation models rely on collaborative filtering for recommendations, we regard the prediction score by these collaborative filtering models as collaborative information. Akin to the injection of popularity, we can substitute the variable $P_i$ in Eq. (3) with the prediction score to inject collaborative information into the grounding process. For more details, please refer to Section §4.4.

## 4 EXPERIMENTS

In this section, we conduct experiments to answer the following research questions:

- RQ1: How does the performance of BIGRec compare to existing methods, when trained on a limited sample of 1024 data points?

---

[2]In Section §4.4, we still reweight prediction scores using PDA [57] to incorporate popularity into two traditional models.

- RQ2: How does the performance of our model trained with limited data compare to that of traditional recommendation models trained with 100× or even 1,000× more samples?
- RQ3: Can BIGRec achieve significant performance gains with increasing training data like traditional models?
- RQ4: To what extent can the performance of BIGRec be enhanced by integrating popularity/collaborative information during model grounding?

### 4.1 Experimental Settings

*4.1.1 Datasets.* We conduct experiments on two datasets:

- **Movie.** This refers to the well-known benchmark dataset for movie recommendation — MovieLens10M[3]. It contains 10,682 items, 10,000,054 interactions, and 9,301,274 interaction sequences.
- **Game.** This is a video-games recommendation dataset from Amazon[4] and we use its 5-core subset. The dataset contains 17,408 items, 496,315 interactions, and 149,796 interaction sequences.

We chose the two datasets intentionally, as they exhibit different characteristics with regard to popularity bias. The Movie dataset reveals a significant bias toward popular items, while the Game dataset demonstrates a lesser degree of bias. Figure 2(a) illustrates the frequency of interactions across items with different levels of popularity, confirming that the Game dataset presents a more equitable distribution of interactions between popular and unpopular items than the Movie dataset. This suggests a lower degree of popularity bias in the Game dataset.

To simulate real-world sequential recommendation scenarios, we divided each dataset into 10 periods based on the timestamp of the interactions. Subsequently, we split the periods of each dataset into training, validation, and testing sets[5] using a ratio of 8:1:1. This approach ensures that the predicted interactions during testing come after all interactions observed during training, preventing the leakage of the testing phase information [21] during training model. This is more akin to real-world situations [21, 58].

*4.1.2 Evaluation Protocols.* Following the previous work [23, 50], for each testing interaction, we set the input historical interaction sequence as the set of interactions that happened immediately prior to it. This approach allows for the possibility of including interactions that occur during the testing phase in the input sequence. To evaluate the model's performance, we use two commonly used evaluation metrics: Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG), which we compute using the all-ranking protocol [50]. In this protocol, all items that a user has not interacted with are considered potential candidates.

*4.1.3 Compared Method.* To demonstrate the superiority of our method in the sequential recommendation, we compare it against the following conventional methods and LLM-based methods:

- **GRU4Rec [18].** This is an RNN-based model that uses Gated Recurrent Units (GRU) to encode user's past interactions with items and generate recommendations based on the learned patterns.
- **Caser [38].** It is a CNN-based model that represents the sequence of recent items as an image and uses convolutional filters in both horizontal and vertical directions to capture sequential patterns. In our implementation, we use one vertical filter and 16 horizontal filters with searching the height in {2, 3, 4}.

---

[3]https://grouplens.org/datasets/movielens/10m/
[4]https://jmcauley.ucsd.edu/data/amazon/
[5]Due to limitations in the inference speed of LLM models, we randomly sample 5,000 validation and testing interactions as the final validation and testing sets, respectively.
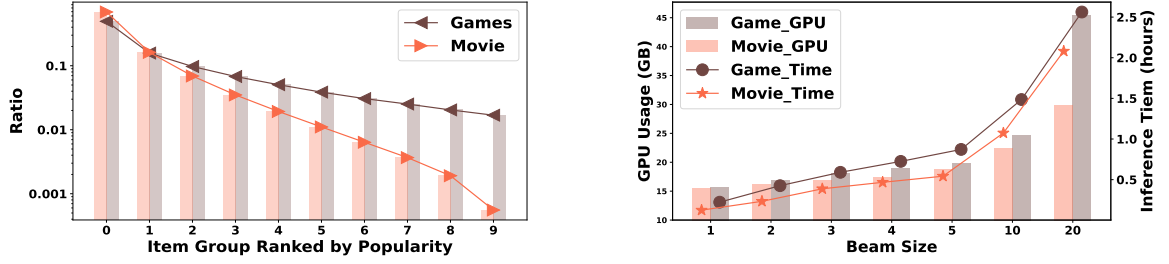
(a) The distribution of items with varying popularity. It is evident that the left item groups have higher levels of popularity in this figure.



(b) The GPU usage and inference time for different beam sizes with fp16 model and batch size equals to 1 for fair comparision.

Fig. 2. The two figure show the distribution of items with different popularity and the GPU usage and inference time for different beam sizes on two dataset, respectively.

- **SASRec [23].** This is a self-attention-based model that uses a causal (left-to-right) attention mechanism to learn sequential patterns and predict the next item.
- **P5 [17].** This approach utilizes the T5 [36] model as the backbone model and leverages a combination of item IDs and natural language templates to undergo continued pre-training for various recommendation tasks[6].
- **DROS [50].** This is the state-of-the-art sequential recommendation method that employs distributionally robust optimization techniques to enhance recommendation robustness against distributional changes.
- **GPT4Rec-LLaMA [27]** is a modified version of the language model-based recommendation approach, GPT4Rec. The original method uses GPT-2 to generate hypothetical "search queries" based on a user's historical data, which are then searched using the BM25 search engine[7] to retrieve recommended items . To ensure a fair comparison, we have replaced the GPT-2 model with the LLaMA-7B model and performed model tuning using the LoRA [20] technique.

We primarily utilize decoder-only LLMs as the backbone LLM model in our **BIGRec** method, given their prominent roles in the LLMs field. Specifically, we have chosen LLaMA-7B as our default choice for this study. In addition to GPT4Rec, there are other existing LLM-based recommendation methods, such as TALLRec [3]. However, since these methods are not applicable to our all-ranking setting, we do not compare them in our study.

*4.1.4   Implementation Details.* We implement all of our methods using PyTorch. To preprocess the data, we first pad user interaction history with lengths less than 11 to a fixed length of 11. Then, we use a sliding window of length 11 to extract sequences, where the last item in each sequence is used as the prediction target, and the preceding items from the past interaction sequence for it. For all conventional models, we optimize them using the binary cross-entropy loss and uniformly sample negative samples. We use the Adam [24] as the optimizer with a tuned learning rate of $1e-3$, a batch size of 1024, and tune the weight decay in the range of $[1e\text{-}2, 1e\text{-}3, 1e\text{-}4, 1e\text{-}5, 1e\text{-}6, 1e\text{-}7]$. Regarding the hyperparameters of the conventional model architectures, we set their embedding size to 64 and the dropout ratio to 0.1. As suggested by the original papers, we only use one GRU layer for GRU4Rec; for SASRec, we set the number of self-attention heads and blocks to 1. We implemented DROS based on SASRec, as it performs the best in most scenarios among the three conventional sequential recommendation methods. For LLM-based methods, we follow the Alpaca setting [39], directly set the learning rate to $1e-4$, and use the AdamW [31] optimizer. During the generation, due to

---

[6]For a fair comparison, we exclusively retain the task of sequence recommendation and its corresponding multiple templates during the training process.
[7]We use the Rank-BM25 package at https://github.com/dorianbrown/rank_bm25.

Table 2. Comparison of model performances under the few-shot training setting (1024 samples) with NDCG@K (NG@K) and HR@K (HR@K) as the metrics. The best results are highlighted in bold, and the sub-optimal results are underlined. 'Improve' refers to the relative improvement of BIGRec over the best baseline.

| Dataset | Model | NG@1 | NG@3 | NG@5 | NG@10 | NG@20 | HR@1 | HR@3 | HR@5 | HR@10 | HR@20 |
|---------|-------|------|------|------|-------|-------|------|------|------|-------|-------|
| Movie | GRU4Rec | 0.0015 | 0.0034 | 0.0047 | 0.0070 | 0.0104 | 0.0015 | 0.0047 | 0.0079 | 0.0147 | 0.0281 |
| | Caser | 0.0020 | 0.0035 | 0.0052 | 0.0078 | 0.0109 | 0.0020 | 0.0046 | 0.0088 | 0.0171 | 0.0293 |
| | SASRec | 0.0023 | 0.0051 | 0.0062 | 0.0082 | 0.0117 | 0.0023 | 0.0070 | 0.0097 | 0.0161 | 0.0301 |
| | P5 | 0.0014 | 0.0026 | 0.0036 | 0.0051 | 0.0069 | 0.0014 | 0.0035 | 0.0059 | 0.0107 | 0.0176 |
| | DROS | 0.0022 | 0.0040 | 0.0052 | 0.0081 | 0.0112 | 0.0022 | 0.0051 | 0.0081 | 0.0173 | 0.0297 |
| | GPT4Rec-LLaMA | 0.0016 | 0.0022 | 0.0024 | 0.0028 | 0.0035 | 0.0016 | 0.0026 | 0.0030 | 0.0044 | 0.0074 |
| | **BIGRec (1024)** | **0.0176** | **0.0214** | **0.0230** | **0.0257** | **0.0283** | **0.0176** | **0.0241** | **0.0281** | **0.0366** | **0.0471** |
| | **Improve** | **654.29%** | **323.31%** | **273.70%** | **213.71%** | **142.55%** | **654.29%** | **244.71%** | **188.39%** | **111.97%** | **56.55%** |
| Game | GRU4Rec | 0.0013 | 0.0016 | 0.0018 | 0.0024 | 0.0030 | 0.0013 | 0.0018 | 0.0024 | 0.0041 | 0.0069 |
| | Caser | 0.0007 | 0.0012 | 0.0019 | 0.0024 | 0.0035 | 0.0007 | 0.0016 | 0.0032 | 0.0048 | 0.0092 |
| | SASRec | 0.0009 | 0.0012 | 0.0015 | 0.0020 | 0.0025 | 0.0009 | 0.0015 | 0.0021 | 0.0037 | 0.0057 |
| | P5 | 0.0002 | 0.0005 | 0.0007 | 0.0010 | 0.0017 | 0.0002 | 0.0007 | 0.0012 | 0.0023 | 0.0049 |
| | DROS | 0.0006 | 0.0011 | 0.0013 | 0.0016 | 0.0022 | 0.0006 | 0.0015 | 0.0019 | 0.0027 | 0.0052 |
| | GPT4Rec-LLaMA | 0.0000 | 0.0000 | 0.0000 | 0.0001 | 0.0001 | 0.0000 | 0.0000 | 0.0000 | 0.0002 | 0.0002 |
| | **BIGRec (1024)** | **0.0133** | **0.0169** | **0.0189** | **0.0216** | **0.0248** | **0.0133** | **0.0195** | **0.0243** | **0.0329** | **0.0457** |
| | **Improve** | **952.63%** | **976.26%** | **888.19%** | **799.64%** | **613.76%** | **952.63%** | **985.19%** | **660.42%** | **586.11%** | **397.10%** |

the substantial computational and inferential costs associated with the extensive use of GPU shown in Figure 2(b), we follow the setting of the previous work [5, 8, 41] and employ a beam size of 4 to generate meaningful outputs.

For the hyperparameters of BM25, we follow the tuning range provided in GPT4Rec [27] paper. For model selection, we employ an early stop strategy with a patience of 20 epochs for baseline methods and 5 epochs for LLM-based methods. Additionally, we report the average results for three random seeds. For the hyperparameters of $\gamma$ in Eq. (3), due to the disparate calculation methods employed by traditional models and LLM-based recommendation models, when incorporating information, we conducted a detailed search for a balanced integration of both approaches[8].

## 4.2 Performance Comparison with Limited Training Data (RQ1)

We first follow existing work [3] to study the effectiveness of our method with limited training data (1024 training points) per dataset. We compare it with baselines trained with equivalent training samples. The comparison results are summarized in Table 2, where we draw the following main observations:

- When training data is limited, the conventional sequential baselines (GRU4Rec, Caser, SASRec) exhibit significantly worse performance than BIGRec implemented with LLM. These results are not surprising as these baselines rely on ID embeddings to build a recommender model, which is difficult to learn well with limited data. In contrast, BIGRec rapidly leverages the semantic knowledge of LLM obtained during the pre-training phase to achieve effective recommendations. These findings demonstrate the superiority of BIGRec over traditional recommendation models when dealing with limited training data. Furthermore, the improvement of BIGRec over these baselines in the top-ranked positions is much higher, indicating that BIGRec may tend to rank items of interest to users in higher positions.
- While GPT4Rec-LLaMA is also LLM-based, it exhibits poor performance compared to BIGRec. We attribute this to the fact that BM25, which can also be thought of as a method to ground the LLM outputs into actual items, is not suitable for the recommendation task. BM25 is a retrieval tool designed for document-level text. However, for the

---

[8]Range from 0 to 1 with an increment of 0.01, and from 1 to 100 with an increment of 1.
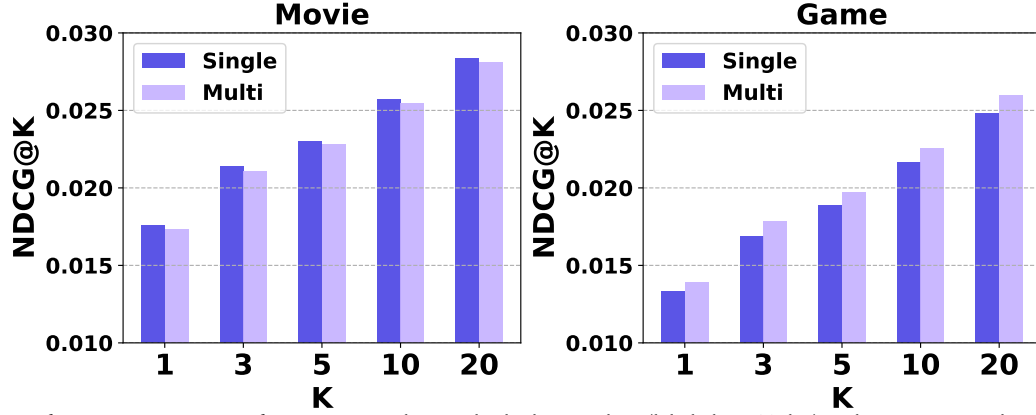
Fig. 3. Performance comparison of BIGRec trained on multiple domain data (labeled as "Multi") and BIGRec trained on single target-domain data (labeled as "Single"), shown for NDCG@K on Movie and Game domains.

two datasets used in our study, the queries in GPT4Rec (item titles) are very short. As a result, BM25 is susceptible to the impact of low-frequency word noise, making it difficult to accurately retrieve relevant items.

- The improvement of BIGRec over conventional models is significantly higher for the Game dataset compared to the Movie dataset. This difference is possibly due to the varying properties of popularity bias between the two datasets. Conventional methods tend to capture popular bias, while BIGRec is less affected by popularity bias. Therefore, conventional methods may perform better on the Movie dataset, in which popular items play more dominant roles as evidenced by Figure 2(a).

Furthermore, we also testify to the potential of BIGRec for grounding to different actual item spaces simultaneously. To explore this, we randomly selected 1024 samples from two domains (Movie and Game) to train our model and tested it on the testing sets of two domains, respectively. As shown in Figure 3, the results demonstrate that BIGRec trained on two-domain data performs comparably on each individual domain as compared to when it is trained on the data of the single domain. These findings suggest that BIGRec can simultaneously ground LLMs to different actual item spaces, at least at the level of different domains.

### 4.3 Performance Comparison to Baselines Trained with More Data (RQ2)

In light of the remarkable performance improvements demonstrated by BIGRec over baselines trained on limited data, we further comprehend the disparity between BIGRec trained on limited data and baselines trained with significantly larger amounts of data (100 or even 1,000 times more) to explore the extent of this gap.

Table 3 contains the results of the baselines trained with more data, which consisted of over 7 million sequences for Movie and 120 thousand for Game, alongside the results of BIGRec trained with only 1024 samples[9]. From the table, we find that, compared to conventional baselines (excluding DROS) on the Movie data, BIGRec exhibits better performance at the top position of the recommendation list, but performs worse with regard to the relative bottom positions of the recommendation list. Contrary to the results obtained from the Movie dataset, our method consistently demonstrates superior performance compared to traditional models (excluding DROS) when applied to the Game data. Compared

---

[9]Due to the constraints of computational resources, we only present the results of baselines that consume significant computational resources on the Game dataset.

Table 3. Performance comparison among baselines trained with full data, BIGRec (0) did not train the model, BIGRec (1024) trained with 1024 samples, and BIGRec (full) trained with full data. "Most-Pop" refers to the method recommending the most popular items. The best results are highlighted in Bold and sub-optimal results are underlined. BIGRec (full) on Movie is omitted due to high computation cost.

| Injected | Model | NG@1 | NG@3 | NG@5 | NG@10 | NG@20 | HR@1 | HR@3 | HR@5 | HR@10 | HR@20 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | **Movie** | | | | | |
| Single Model (+None) | Most-Pop | 0.0032 | 0.0076 | 0.0088 | 0.0121 | 0.0170 | 0.0032 | 0.0108 | 0.0138 | 0.0244 | 0.0438 |
| | GRU4Rec | 0.0047 | 0.0108 | 0.0151 | 0.0237 | 0.0351 | 0.0047 | 0.0155 | 0.0259 | 0.0527 | 0.0985 |
| | Caser | 0.0045 | 0.0113 | 0.0161 | 0.0242 | 0.0354 | 0.0045 | 0.0165 | 0.0281 | 0.0537 | 0.0986 |
| | SASRec | 0.0045 | 0.0119 | 0.0171 | 0.0268 | 0.0389 | 0.0045 | 0.0175 | 0.0302 | 0.0606 | 0.1088 |
| | DROS | 0.0087 | 0.0186 | 0.0245 | 0.0359 | 0.0493 | 0.0087 | 0.0261 | 0.0406 | 0.0761 | 0.1292 |
| | **BIGRec (0)** | 0.0020 | 0.0034 | 0.0046 | 0.0060 | 0.0095 | 0.0020 | 0.0044 | 0.0074 | 0.0120 | 0.0258 |
| | **BIGRec (1024)** | **0.0176** | 0.0214 | 0.0230 | 0.0257 | 0.0283 | **0.0176** | 0.0241 | 0.0281 | 0.0366 | 0.0471 |
| + DROS | Caser | 0.0087 | 0.0183 | 0.0247 | 0.0354 | 0.0494 | 0.0087 | 0.0258 | 0.0404 | 0.0756 | 0.1296 |
| | SASRec | 0.0089 | 0.0184 | 0.0245 | 0.0357 | 0.0493 | 0.0089 | 0.0256 | 0.0409 | 0.0754 | 0.1307 |
| | **BIGRec (1024)** | **0.0176** | **0.0250** | **0.0315** | **0.0427** | **0.0562** | **0.0176** | **0.0308** | **0.0464** | **0.0813** | **0.1353** |
| | | | | | | **Game** | | | | | |
| Single Model (+None) | Most-Pop | 0.0000 | 0.0000 | 0.0000 | 0.0004 | 0.0018 | 0.0000 | 0.0000 | 0.0000 | 0.0014 | 0.0068 |
| | GRU4Rec | 0.0051 | 0.0080 | 0.0094 | 0.0109 | 0.0129 | 0.0051 | 0.0101 | 0.0135 | 0.0184 | 0.0263 |
| | Caser | 0.0059 | 0.0094 | 0.0111 | 0.0141 | 0.0177 | 0.0059 | 0.0119 | 0.0161 | 0.0256 | 0.0401 |
| | SASRec | 0.0113 | 0.0151 | 0.0164 | 0.0185 | 0.0204 | 0.0113 | 0.0179 | 0.0209 | 0.0275 | 0.0353 |
| | P5 | 0.0094 | 0.0116 | 0.0131 | 0.0145 | 0.0167 | 0.0094 | 0.0134 | 0.0172 | 0.0216 | 0.0300 |
| | DROS | 0.0156 | 0.0194 | 0.0213 | 0.0244 | 0.0278 | 0.0156 | 0.0221 | 0.0269 | 0.0365 | 0.0500 |
| | **BIGRec (0)** | 0.0020 | 0.0032 | 0.0036 | 0.0054 | 0.0073 | 0.002 | 0.004 | 0.005 | 0.0106 | 0.0184 |
| | **BIGRec (1024)** | 0.0133 | 0.0169 | 0.0189 | 0.0216 | 0.0248 | 0.0133 | 0.0195 | 0.0243 | 0.0329 | 0.0457 |
| | **BIGRec (full)** | **0.0221** | 0.0250 | 0.0272 | 0.0297 | 0.0319 | **0.0221** | 0.0270 | 0.0326 | 0.0401 | 0.0490 |
| + DROS | Caser | 0.0159 | 0.0199 | 0.0217 | 0.0249 | 0.0288 | 0.0159 | 0.0227 | 0.0279 | 0.0375 | 0.0529 |
| | SASRec | 0.0151 | 0.0197 | 0.0216 | 0.0247 | 0.0279 | 0.0151 | 0.0226 | 0.0279 | 0.0373 | 0.0488 |
| | **BIGRec (1024)** | 0.0133 | 0.0243 | 0.0268 | 0.0302 | 0.0338 | 0.0133 | 0.0320 | 0.0377 | 0.0474 | **0.0619** |
| | **BIGRec (full)** | **0.0221** | **0.0292** | **0.0312** | **0.0338** | **0.0375** | **0.0221** | **0.0340** | **0.0387** | **0.0478** | 0.0611 |

to the meticulously optimized and state-of-the-art traditional method DROS, BIGRec trained with only 1024 samples could still show comparable performance in most cases. Based on the results, we draw two conclusions:

- Our method, BIGRec, which was trained on a relatively small subset of the dataset with only 1,024 samples, has shown comparable performance to conventional models that were trained with a significantly larger number of data, with 100 or even 1,000 times bigger, particularly when recommendation (exposure) resources are limited. These findings highlight the practicality of utilizing LLMs for recommendation systems. Meanwhile, the results also suggest that there is potential to further enhance BIGRec's performance by increasing the size of the training set and expanding the range of items it has encountered during grounding (see BIGRec (full) in Tabel 3).
- When compared to the Game dataset, the Movie dataset exhibits higher density and is more susceptible to popularity bias (as evidenced by the performance of the Most-Pop methods on the two datasets and Figure 2(a)). The difference in the performance gain of BIGRec between the two datasets suggests that BIGRec performs exceptionally well in scenarios where data is scarce and works effectively with less reliance on popularity bias.

### 4.4 Performance of BIGRec with Increasing Training Samples (RQ3)

When comparing the results of traditional ID-based methods (such as SASRec and DROS) in Table 2 and Table 3, it becomes evident that increasing the training data can enhance the model performance. This improvement can be
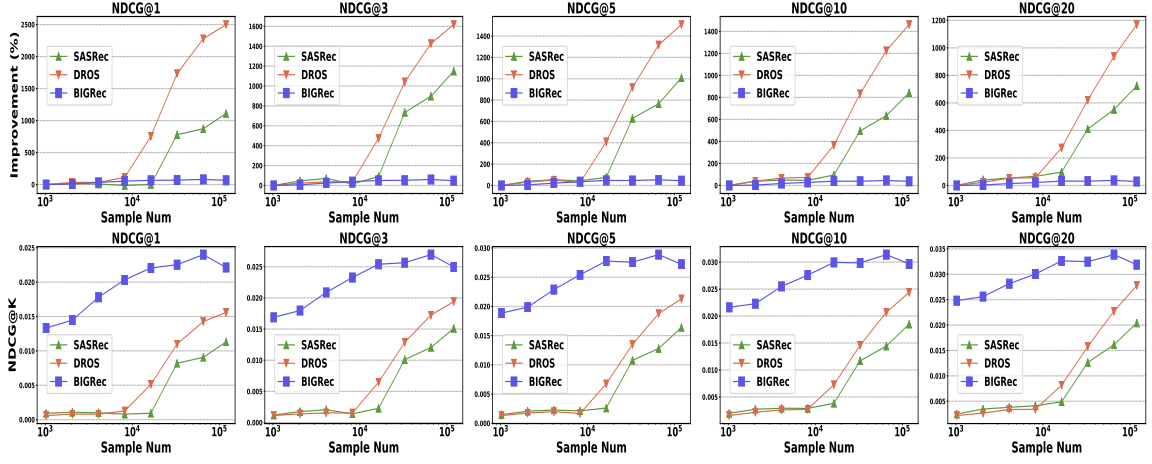
Fig. 4. Performance of SASRec, DROS, and BIGRec as training data size increases (denoted by Sample Num), along with their respective performance improvement curves relative to their initial states (1024 training samples). The bottom sub-figures showcase the recommendation performance (measured by NDCG@K), while the upper sub-figures illustrate the improvement.

attributed to the fact that increasing the training data aids in the acquisition of statistical information (*e.g.,* popularity and collaborative information) contained in the dataset, which is shown beneficial for recommendation [37, 57].

To investigate whether LLMs can also capture dataset local statistical information in the dataset by increasing the amount of training data to enhance performance, we analyze the performance of BIGRec when increasing training data on Game. Figure 4 illustrates the performance curve when increasing the training data, as well as the curve depicting the performance improvement compared to the initial state. As shown in the figure, compared to traditional recommendation models, the increase in data volume has a limited impact on the performance improvements of BIGRec. We guess that BIGRec is less aggressive in capturing the statistics present in the dataset, and instead prefers to utilize the semantic information of LLMs to accomplish tasks while disregarding valuable statistical information. Therefore, increasing training data could not significantly enhance BIGRec performance compared to traditional ID-based methods.

### 4.5 Performance of Introducing Valuable Statistical Information (RQ4)

The aforementioned speculations suggest that BIGRec heavily relies on the semantic information stored in LLM while disregarding the dataset's valuable statistical information. In light of this, we believe that appropriately incorporating valuable statistical information could enhance BIGRec's recommendation capabilities. The consideration is that the information may be ignored by BIGRec during training but indeed describes some useful characteristics of the actual items, and thus incorporating it may enhance the grounding of BIGRec to the actual item space. To verify it, we conduct experiments by introducing two types of statistical information: popularity and collaborative information, respectively. Note that the two types of information are recognized as valuable for recommendation systems [37, 57].

• **Introducing popularity information.** We first study the performance of BIGRec when introducing the popularity information of the training dataset using the method in Eq. (3). We conduct experiments on the Movie dataset, considering that the popularity information is more important on it. We compare the original version of BIGRec with the version that incorporates popularity information in Figure 5. The figure demonstrates that after incorporating popularity information, BIGRec achieves performance improvements for the NDCG@K and Recall@K metrics, particularly for
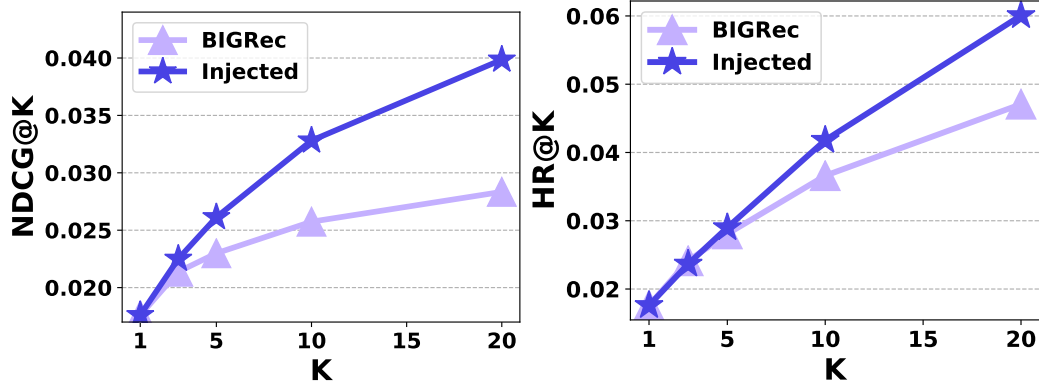
Fig. 5. Performance comparison between BIGRec with popularity injection during grounding (labeled as "Injected") and the original BIGRec. NDCG@K and HR@K metrics are displayed for different values of K.
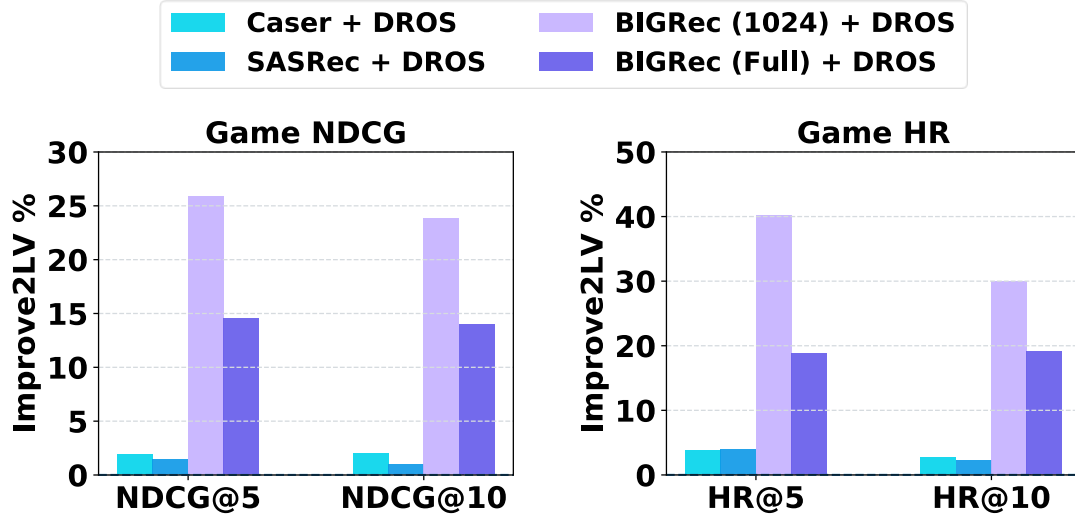


Fig. 6. Performance improvement of SASRec, Caser, BIGRec (1024), and BIGRec (full) incorporated with the DROS on the Game dataset. The improvements are related to the larger value of the two combined models' performances, denoted as 'Improve2LV'. Notably, all the baselines here (SASRec, Caser, and DROS) are trained on the full dataset.

larger values of $K$. Additionally, the performance is substantially better than when using only popularity information for recommendation (as shown in Table 3). These results indicate that injecting popularity information can be beneficial for grounding the output of LLMs to actual items in BIGRec, further enhancing its performance.

• **Introducing collaborative information.** We next study the performance of BIGRec when incorporating the collaborative information encoded by conventional models (*e.g.,* DROS) into it using the method in Equation (3). For comparison, we also consider incorporating collaborative information into conventional models. The performances are shown in Tabel 3, and performance improvements are shown in Figure 6. From the table and figure, we draw on two observations: 1) incorporating the collaborative information into both BIGRec and traditional models could bring model performance improvements; 2) incorporating collaborative information into BIGRec yields a more significant enhancement compared to incorporating information into a different conventional model. These results indicate that

BIGRec relies more on more information (semantic information) that is different from collaborative information to generate recommendations, thus incorporating collaborative information could bring more gains.

The two experiments indicate that there is significant potential for advancing recommendation accuracy even further by investigating more effective and efficient techniques for enhancing the grounding to the actual item space in BIGRec.

## 5 CONCLUSION AND FUTURE WORK

In this paper, we aim to thoroughly discuss the performance across all ranks for LLM4Rec. In order to fully exploit the potential of LLMs, we formulate a bi-step grounding paradigm for recommendation (BIGRec), which follows a generative methodology that can effectively incorporate statistical information. Our empirical findings indicate that BIGRec trained on 1024 samples considerably outstrips that of traditional recommender models trained on an identical number of samples, attaining performance on par even with conventional models trained on the full dataset in most cases. Furthermore, we have scaled up the training data for LLM-based recommendations. The results indicate that LLMs rely heavily on semantics for recommendations while ignoring some other useful information. Based on this insight, we have infused popularity and collaborative information into the LLM, further enhancing its capability and validating our assumption.

In view of the aforementioned discussions, it is inferred that employing a bi-step grounding approach in LLM-based recommendations shows promise. However, several unresolved issues persist. Firstly, in terms of the initial stage of grounding in the recommendation space, our experiments reveal that satisfactory recommendation performance can be achieved with a relatively small number of samples. Besides, further increasing the sample size can potentially enhance the performance. However, there are two opposing considerations. On one hand, training an LLM incurs considerable costs. Is it feasible to select appropriate samples for constructing the recommendation space, thereby reducing the associated expenses? On the other hand, the question of whether all items should belong to the same recommendation space remains a topic of debate.

Furthermore, concerning the second grounding step, the methodology we currently employ is rather crude, extracting embeddings using the decoder model and computing similarity is an elementary operation. Despite our ability to regulate the production of samples with a large beam size, this approach is highly time-consuming. Therefore, it is imperative to find a more efficient method for connecting with actual items. Moreover, we have confirmed the viability of integrating item popularity and collaborative information during the grounding process. We hope to incorporate more conventional recommendation features more effectively to improve model performance further.

## REFERENCES

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[2] Qingyao Ai, Ting Bai, and et al. 2023. Information Retrieval Meets Large Language Models: A Strategic Report from Chinese IR Community. *arXiv preprint arXiv:2307.09751* (2023).

[3] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *Recsys short* (2023).

[4] Lisa Beinborn, Teresa Botschen, and Iryna Gurevych. 2018. Multimodal grounding for language processing. *arXiv preprint arXiv:1806.06371* (2018).

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/

1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[6] Zheng Chen. 2023. PALR: Personalization Aware LLMs for Recommendation. *arXiv preprint arXiv:2305.07622* (2023).

[7] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality. https://lmsys.org/blog/2023-03-30-vicuna/

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).

[9] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2020. MV-RNN: A Multi-View Recurrent Neural Network for Sequential Recommendation. *IEEE Trans. Knowl. Data Eng.* 32, 2 (2020), 317–331. https://doi.org/10.1109/TKDE.2018.2881260

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).* Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[11] Sihao Ding, Fuli Feng, Xiangnan He, Jinqiu Jin, Wenjie Wang, Yong Liao, and Yongdong Zhang. 2022. Interpolative Distillation for Unifying Biased and Debiased Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 40–49. https://doi.org/10.1145/3477495.3532002

[12] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential user-based recurrent neural network recommendations. In *Proceedings of the eleventh ACM conference on recommender systems.* 152–160.

[13] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378* (2023).

[14] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender Systems in the Era of Large Language Models (LLMs). *arXiv preprint arXiv:2307.02046* (2023).

[15] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. 2020. Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. *ACM Trans. Inf. Syst.* 39, 1 (2020), 10:1–10:42. https://doi.org/10.1145/3426723

[16] Luke Friedman, Sameer Ahuja, David Allen, Terry Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging Large Language Models in Conversational Recommender Systems. *arXiv preprint arXiv:2305.07961* (2023).

[17] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems.* 299–315.

[18] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR.*

[19] 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.

[20] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022.* OpenReview.net. https://openreview.net/forum?id=nZeVKeeFYf9

[21] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2023. A critical study on data leakage in recommender system offline evaluation. *ACM Transactions on Information Systems* 41, 3 (2023), 1–27.

[22] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. 2022. Prompting visual-language models for efficient video understanding. In *European Conference on Computer Vision.* Springer, 105–124.

[23] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018.* IEEE Computer Society, 197–206.

[24] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

[25] Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining.* 1748–1757.

[26] Jiatong Li, Yunqing Liu, Wenqi Fan, Xiao-Yong Wei, Hui Liu, Jiliang Tang, and Qing Li. 2023. Empowering Molecule Discovery for Molecule-Caption Translation with Large Language Models: A ChatGPT Perspective. *arXiv preprint arXiv:2306.06615* (2023).

[27] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879* (2023).

[28] Ruyu Li, Wenhao Deng, Yu Cheng, Zheng Yuan, Jiaqi Zhang, and Fajie Yuan. 2023. Exploring the Upper Limits of Text-Based Collaborative Filtering Using Large Language Models: Discoveries and Insights. *arXiv preprint arXiv:2305.11700* (2023).

[29] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16.* Springer, 121–137.

[30] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149* (2023).

[31] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=Bkg6RiCqY7

[32] Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. 2018. Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 596–605. https://doi.org/10.1145/3219819.3219828

[33] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2021. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. *CoRR* abs/2110.05730 (2021). arXiv:2110.05730 https://arxiv.org/abs/2110.05730

[34] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, K. Selcuk Candan, Huan Liu, Leman Akoglu, Xin Luna Dong, and Jiliang Tang (Eds.). ACM, 813–823. https://doi.org/10.1145/3488560.3498433

[35] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019).

[36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[37] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[38] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*. 565–573.

[39] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

[40] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *CoRR* abs/2302.13971 (2023). https://doi.org/10.48550/arXiv.2302.13971 arXiv:2302.13971

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[42] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 6332–6338. https://doi.org/10.24963/ijcai.2019/883

[43] Zhenlei Wang, Shiqi Shen, Zhipeng Wang, Bo Chen, Xu Chen, and Ji-Rong Wen. 2022. Unbiased Sequential Recommendation with Latent Confounders. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, Frédérique Laforest, Raphaël Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Médini (Eds.). ACM, 2195–2204. https://doi.org/10.1145/3485447.3512092

[44] Zekun Wang, Ge Zhang, Kexin Yang, Ning Shi, Wangchunshu Zhou, Shaochun Hao, Guangzheng Xiong, Yizhi Li, Mong Yuan Sim, Xiuying Chen, et al. 2023. Interactive natural language processing. *arXiv preprint arXiv:2305.13246* (2023).

[45] Hongyi Wen, Xinyang Yi, Tiansheng Yao, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2022. Distributionally-robust Recommendations for Improving Worst-case User Experience. In *Proceedings of the ACM Web Conference 2022*. 3606–3610.

[46] 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).

[47] Lingxi Xie, Longhui Wei, Xiaopeng Zhang, Kaifeng Bi, Xiaotao Gu, Jianlong Chang, and Qi Tian. 2023. Towards AGI in Computer Vision: Lessons Learned from GPT and Large Language Models. *arXiv preprint arXiv:2306.08641* (2023).

[48] Chengfeng Xu, Jian Feng, Pengpeng Zhao, Fuzhen Zhuang, Deqing Wang, Yanchi Liu, and Victor S. Sheng. 2021. Long- and short-term self-attention network for sequential recommendation. *Neurocomputing* 423 (2021), 580–589. https://doi.org/10.1016/j.neucom.2020.10.066

[49] An Yan, Shuo Cheng, Wang-Cheng Kang, Mengting Wan, and Julian J. McAuley. 2019. CosRec: 2D Convolutional Neural Networks for Sequential Recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 2173–2176. https://doi.org/10.1145/3357384.3358113

[50] Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A Generic Learning Framework for Sequential Recommendation with Distribution Shifts. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[51] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. 2020. Parameter-Efficient Transfer from Sequential Behaviors for User Modeling and Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1469–1478. https://doi.org/10.1145/3397271.3401156

[52] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman (Eds.). ACM, 582–590. https://doi.org/10.1145/3289600.3290975

[53] Yan Zeng, Xinsong Zhang, Hang Li, Jiawei Wang, Jipeng Zhang, and Wangchunshu Zhou. 2022. $X^2$-VLM: All-In-One Pre-trained Model For Vision-Language Tasks. *CoRR* abs/2211.12402 (2022). https://doi.org/10.48550/arXiv.2211.12402 arXiv:2211.12402

[54] Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. *arXiv preprint arXiv:2305.07609* (2023).

[55] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001* (2023).

[56] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 4320–4326. https://doi.org/10.24963/ijcai.2019/600

[57] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal Intervention for Leveraging Popularity Bias in Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 11–20. https://doi.org/10.1145/3404835.3462875

[58] Yang Zhang, Tianhao Shi, Fuli Feng, Wenjie Wang, Dingxian Wang, Xiangnan He, and Yongdong Zhang. 2023. Reformulating CTR Prediction: Learning Invariant Feature Interactions. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[59] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *CoRR* abs/2303.18223 (2023). arXiv:2303.18223

[60] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. arXiv:2306.05685 [cs.CL]

[61] 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).