Bridging Items and Language: A Transition Paradigm for Large Language Model-Based Recommendation

Xinyu Lin
xylin1028@gmail.com
National University of Singapore
Singapore

Wenjie Wang*
wenjiewang96@gmail.com
National University of Singapore
Singapore

Yongqi Li
liyongqi0@gmail.com
The Hong Kong Polytechnic University
Hong Kong SAR, China

Fuli Feng*
fulifeng93@gmail.com
University of Science and Technology of China
Hefei, China

See-Kiong Ng
seekiong@nus.edu.sg
National University of Singapore
Singapore

Tat-Seng Chua
dcscts@nus.edu.sg
National University of Singapore
Singapore

ABSTRACT

Harnessing Large Language Models (LLMs) for recommendation is rapidly emerging, which relies on two fundamental steps to bridge the recommendation item space and the language space: 1) **item indexing** utilizes identifiers to represent items in the language space, and 2) **generation grounding** associates LLMs' generated token sequences to in-corpus items. However, previous methods exhibit inherent limitations in the two steps. Existing ID-based identifiers (*e.g.,* numeric IDs) and description-based identifiers (*e.g.,* titles) either lose semantics or lack adequate distinctiveness. Moreover, prior generation grounding methods might generate invalid identifiers, thus misaligning with in-corpus items.

To address these issues, we propose a novel **Trans**ition paradigm for LLM-based **Rec**ommender (named TransRec) to bridge items and language. Specifically, TransRec presents multi-facet identifiers, which simultaneously incorporate ID, title, and attribute for item indexing to pursue both distinctiveness and semantics. Additionally, we introduce a specialized data structure for TransRec to ensure generating valid identifiers only and utilize substring indexing to encourage LLMs to generate from any position of identifiers. Lastly, TransRec presents an aggregated grounding module to leverage generated multi-facet identifiers to rank in-corpus items efficiently. We instantiate TransRec on two backbone models, BART-large and LLaMA-7B. Extensive results on three real-world datasets under diverse settings validate the superiority of TransRec.

*Corresponding author. This research is supported by A*STAR, CISCO Systems (USA) Pte. Ltd and National University of Singapore under its Cisco-NUS Accelerated Digital Economy Corporate Laboratory (Award I21001E0002), and the National Natural Science Foundation of China (62272437).

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671884

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

LLM-based Recommendation, Item Indexing, Generation Grounding, Multi-facet Identifier

ACM Reference Format:

Xinyu Lin, Wenjie Wang*, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Bridging Items and Language: A Transition Paradigm for Large Language Model-Based Recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3637528.3671884

1 INTRODUCTION

Large Language Models (LLMs) have achieved remarkable success across diverse domains [3, 22, 37] due to their emergent competencies, including possessing rich knowledge [18], instruction following [43], and in-context learning [29]. Recently, there has been a notable surge in exploring the benefits of adapting LLMs to recommendations. In particular, LLMs have showcased the potential in discerning nuanced item semantics [55, 63], understanding multiple user interests [11, 62], and generalizing to cold-start item recommendations [2, 6, 13]. In light of these, the prospect of harnessing LLMs as recommender systems, *i.e.,* LLM-based recommenders, emerges as a particularly promising avenue for further exploration.

Given that recommender systems and LLMs work in the item space and the language space, respectively, the key to building LLM-based recommenders lies in bridging the item space and the language space (refer to Figure 1). Here, the item space includes all the existing items on the recommender platform. To bridge the two spaces, it involves two essential steps: item indexing and generation grounding. The item indexing step assigns each item with a unique identifier (*e.g.,* item title or numeric ID) in natural language, and subsequently the user's historical interactions are converted into a sequence of identifiers. To yield recommendations, given the identifier sequence of the user's historical interactions, the generation grounding step utilizes LLMs to generate a token sequence in the language space and inversely ground the token

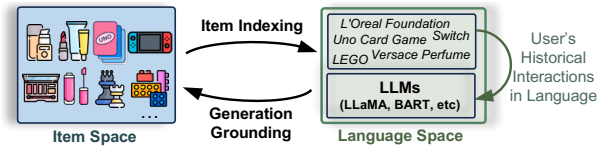

Figure 1: Illustration of the two pivotal steps for LLM-based recommenders: item indexing and generation grounding.

sequence to the items in the item space. However, existing work has intrinsic limitations in both steps.

Item indexing. Previous studies can be categorized into ID-based identifiers [15, 26] and description-based identifiers [2, 8].

- ID-based identifiers utilize numeric IDs (*e.g.,* "15308") to represent items, effectively capturing the uniqueness of items [15, 26]. Nevertheless, IDs lack explicit semantics[1] and hinder the knowledge generalization of LLMs. Worse still, LLMs require sufficient interactions to fine-tune each ID identifier, decreasing the generalization ability to large-scale and cold-start recommendations.
- Description-based identifiers adopt semantic descriptions (*e.g.,* titles and attributes) to index items [2, 8]. However, item descriptions lack adequate distinctiveness due to the existence of common words (*e.g.,* "Will" and "Be" in movie titles). Moreover, item descriptions might not consistently align with user-item interactions: two items with similar descriptions may not have similar interactions. This divergence is a possible reason for the wide usage of IDs in existing feature-based recommendations.

Generation grounding. In previous work, LLMs autoregressively generate a sequence of tokens via beam search, and then ground the token sequence to the identifiers of items via exact matching [15]. Nevertheless, unconstrained generation over the whole vocabulary of LLMs may yield invalid item identifiers, leading to out-of-corpus recommendations [40]. Thus, additional matching strategies (*e.g.,* L2 distance matching [1]) are necessary to ground the out-of-corpus identifiers to existing identifiers, which however is computationally expensive (*cf.* Section 4.3.2).

Drawing upon the above insights, we establish some key objectives for the two steps. For **item indexing**, we posit that item identifiers should at least satisfy two criteria: 1) *distinctiveness* that ensures the items are distinguishable from each other; and 2) *semantics* that guarantees the full utilization of the rich knowledge in LLMs, enhancing the generalization abilities of LLM-based recommenders. For **generation grounding**, we consider constrained generation to ensure generating valid identifiers without additional matching. Nonetheless, a crucial challenge is that constrained generation heavily relies on the generation quality of the initial tokens during beam search. Because constrained generation strictly starts from the first token of valid identifiers [7], LLMs cannot generate the ideal item if the generated first token is incorrect. As such, we consider *position-free constrained generation* to allow LLMs to generate from any position in the valid identifiers.

To improve item indexing and generation grounding as the bridge between the item and language spaces, we propose a novel transition paradigm for LLM-based recommenders (shorted as TransRec). Specifically, 1) as depicted in Figure 2(a), TransRec

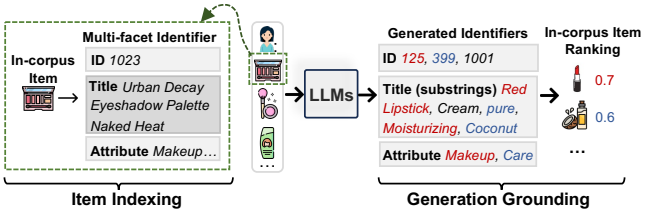

Figure 2: Overview of TransRec. Item indexing assigns each item a multi-facet identifier. For generation grounding, TransRec generates a set of identifiers in each facet and then grounds them to in-corpus items for ranking.

indexes items with multi-facet identifiers, which simultaneously considers item IDs, titles, and attributes (*e.g.,* category) as the identifiers in separate facets to pursue both distinctiveness and semantics. And 2) for generation grounding, to achieve position-free constrained generation, we introduce FM-index, a data structure that supports LLMs to generate any segment of valid identifiers. Besides, to further enhance the position-free generation ability of LLMs, we propose also using substrings of identifiers to index items (*e.g.,* "Lipstick-Red" in "Everyday Elegance Lipstick-Red") for instruction tuning. Lastly, LLMs will generate valid identifiers in each facet as in Figure 2(b), and we present an aggregated grounding module to leverage generated identifiers to rank in-corpus items. To validate the effectiveness of TransRec, we conduct extensive experiments on three real-world datasets under diverse settings, including full training and few-shot training with warm- and cold-start testings. Empirical results on two backbone models BART-large [30] and LLaMA-7B [50] reveal the superiority of TransRec over traditional models and LLM-based models. The code and data are at https://github.com/Linxyhaha/TransRec/.

In summary, this work offers several significant contributions:

- We highlight existing problems and the key objectives in the two fundamental steps of bridging the item and language spaces in LLM-based recommenders.
- We propose a new TransRec paradigm with multi-facet identifiers and position-free constrained generation, seamlessly bridging items and language for LLM-based recommendations.
- We conduct extensive experiments under various recommendation settings, demonstrating the effectiveness of TransRec.

2 PRELIMINARY

This section first introduces the process of establishing LLM-based recommenders, including instruction tuning and generation grounding. And then, we highlight the two key steps to bridge between the item space and the language space, and reveal the vulnerabilities of existing work.

- **Task formulation.** Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items, respectively. We represent the historical interaction sequence of a user u by $\mathcal{S}_u = [i_u^1, i_u^2, \ldots, i_u^L]$ in a chronological order, where $u \in \mathcal{U}$, $i_u \in \mathcal{I}$, and $L = |\mathcal{S}_u|$. Given a user's historical interactions, the sequential recommendation task[2] is to predict this user's next liked item $i_u^{L+1} \in \mathcal{I}$.

[1] Item indexing methods that aim to integrate semantics into numeric IDs are compared and discussed in Section 4.2 and Section 5.

[2] We mainly focus on sequential recommendation as it considers the crucial temporal aspect in real-world scenarios, showing notable practical significance [49, 56, 57].

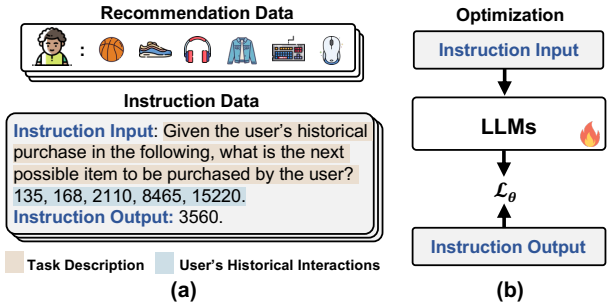

Figure 3: Illustration of instruction tuning of LLMs. (a) depicts the conversion from recommendation data to instruction data; (b) presents the optimization of LLMs based on the instruction data.

To leverage LLMs for recommendation, existing work mainly resorts to instruction tuning, to narrow the intrinsic discrepancy between the LLMs' pre-training data and the recommendation data. After instruction tuning, LLMs are instructed to generate recommended items based on the user's historical interactions. In the following, we introduce the preliminary knowledge of instruction tuning and generation grounding.

2.1 Instruction Tuning

Instruction tuning involves three phases: item indexing, data reconstruction, and LLM optimization.

• **Item indexing.** For each item $i \in \mathcal{I}$ in recommendation data, item indexing assigns an identifier $\tilde{i} \in \tilde{\mathcal{I}}$ in natural language, where $\tilde{\mathcal{I}}$ is the identifier corpus. To achieve item indexing, exiting work can be categorized into ID-based identifiers (*e.g.,* numeric IDs [15, 26]), and description-based identifiers (*e.g.,* titles [1] and attributes [8]).

• **Data reconstruction**. Based on the item identifiers, each user's interaction sequence $\mathcal{S}_u = [i_u^1, i_u^2, \ldots, i_u^L]$ could be converted to a sequence of identifiers $\tilde{\mathcal{S}}_u = [\tilde{i}_u^1, \tilde{i}_u^2, \ldots, \tilde{i}_u^L]$. Thereafter, instruction data $\mathcal{D}_{instruct} = \{(x, y)\}$ are constructed for instruction tuning, where x and y denote the instruction input and output, respectively. Specifically, as shown in Figure 3(a), the instruction input contains the *task description* illustrating the recommendation task; and the *user's historical interactions* $[\tilde{i}_u^1, \tilde{i}_u^2, \ldots, \tilde{i}_u^{L-1}]$ in natural language. The instruction output is usually set to the identifier of the next-interacted item, *i.e.,* $y = \tilde{i}_u^L$.

• **LLM optimization**. Given the instruction data $\mathcal{D}_{instruct}$, the learnable parameters ($\theta \in \Theta$) of an LLM can be optimized by minimizing the negative log-likelihood of instruction output y conditioned on input x:

$$\min_{\theta \in \Theta} \{\mathcal{L}_\theta = -\sum_{t=1}^{|y|} \log P_\theta(y_t | y_{<t}, x)\}, \tag{1}$$

where y_t is the t-th token of y, and $y_{<t}$ represents the token sequence preceding y_t.

2.2 Generation Grounding

After the instruction tuning, we can effectively leverage LLMs to generate recommendations via the generation grounding step, *i.e.,* generating token sequences by LLMs and grounding them to the in-corpus items.

• **Generation.** Given an instruction input x, which contains the *task description* and the *user's historical interactions* $[\tilde{i}_u^1, \tilde{i}_u^2, \ldots, \tilde{i}_u^L]$, LLM-based recommender autoregressively generates a token sequence $\hat{y}$ step by step via beam search. Formally, when beam size = 1, at each time step t, we have

$$\hat{y}_t = \arg\max_{v \in \mathcal{V}} P_\theta(v | \hat{y}_{<t}, x), \tag{2}$$

where $\mathcal{V}$ is the token vocabulary of the LLM. The LLM-based recommender keeps generating until it meets stopping criteria (*e.g.,* $\hat{y}_t$ is the stop token "EOS").

• **Grounding.** The generated token sequence $\hat{y}$ in the language space is then grounded to a set of existing identifiers as recommendations:

$$\{\tilde{i} | \tilde{i} \in \tilde{\mathcal{I}}\} \leftarrow \text{Ground}(\hat{y}),$$

where Ground$(\cdot)$ is the grounding approach, such as exact matching [15], and distance-based matching [1].

To sum up, bridging the item space and the language space for building LLM-based recommenders involves two fundamental steps: ***item indexing*** and ***generation grounding***. Upon the two steps, LLMs can follow the standard operations of data reconstruction and instruction tuning in the language space. However, previous work suffers from intrinsic limitations in the two steps. For item indexing, existing ID-based identifiers and description-based identifiers either lose semantics or lack adequate distinctiveness, leading to the underutilization of rich knowledge in LLMs or losing salient features crucial to the recommendation. Notably, little effort has been made to study or compare the ID- and description-based identifiers specifically for LLM-based recommendation (related work is discussed in Section 5), where the potential enhancement from the incorporation of both IDs and semantics specifically for LLM-based recommendation deserves more exploration. For generation grounding, Eq. (2) allows for generating any token from the LLMs' vocabulary at each step, potentially leading to out-of-corpus identifiers. Additional matching approaches [1] can mitigate the out-of-corpus issue, which however is time-consuming (*cf.* Section 4.3.2).

3 METHOD

To strengthen LLM-based recommenders from the two crucial steps, we propose a novel transition paradigm TransRec, which involves multi-facet item indexing and generation grounding.

3.1 Multi-facet Item Indexing

To alleviate the intrinsic limitations of existing item indexing methods, we postulate two criteria for item identifiers: 1) distinctiveness to ensure the items are distinguishable from each other; and 2) semantics to make full utilization of rich knowledge in LLMs, enhancing the generalization abilities.

3.1.1 ***Multi-facet Identifier.*** Well meeting the above two criteria, we propose multi-facet identifiers for the item indexing step. In particular, we simultaneously incorporate three facets to represent an item from different aspects:

• **Numeric ID** guarantees the distinctiveness among items. We assign each item a unique random numeric ID, denoted by P (*e.g.,* "3471"). By tuning over user-item interactions described by unique

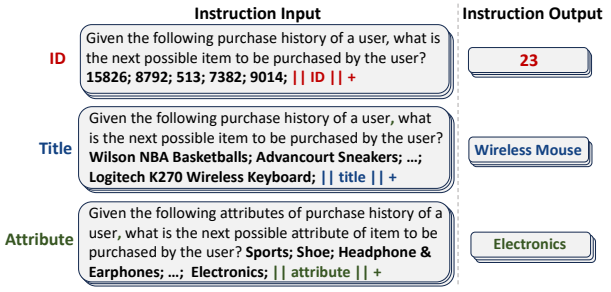

Figure 4: Illustration of the reconstructed data based on the multi-facet identifiers. The bold texts in black refer to the user's historical interactions.

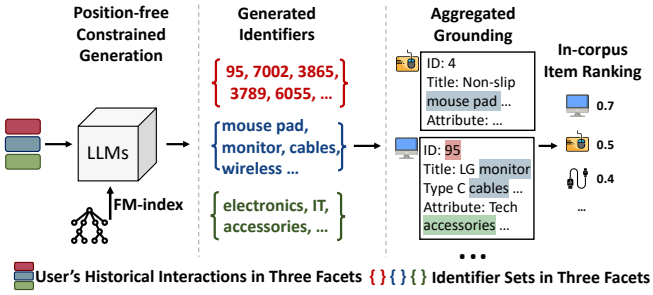

Figure 5: Demonstration of the generation grounding step in TransRec. Red, blue, and green denote the facets of ID, title, and attribute, respectively.

IDs, LLMs are incentivized to align the numeric IDs with the user-item interactions, which can capture crucial Collaborative Filtering (CF) knowledge. In essence, items with similar interactions are endowed with similar ID representations.

• **Item title** ensures rich semantics that can capitalize the wealth of world knowledge in LLMs. An item title, denoted by T, *e.g.,* "Rouge Coco Hydrating Creme Lipstick Chanel #432", typically contains a concise and descriptive name, conveying some general information about the item.

• **Item attribute** serves as a complementary facet to inject semantics, particularly in cases where item titles may be less informative or unavailable. For an item that has multiple attributes such as "Makeup, Eyes, Multicolor, Gluten Free", we denote each attribute as a and the complete attributes as $A = [a_1, a_2, \ldots, a_n]$.

In summary, for each item i in the recommendation data, we can obtain the multi-facet identifier $\tilde{i} = \{P, T, A\}$. Based on the multi-facet identifiers, we then construct the instruction data in language space for the instruction tuning of LLMs.

3.1.2 ***Data Reconstruction.*** As shown in Figure 4, we convert each user's interaction sequence $\mathcal{S}_u = [i_u^1, i_u^2, \ldots, i_u^L]$ into instruction data in three facets, separately. For each facet, we construct instruction input and output based on the user's interaction sequence. We fix the templates of task descriptions[3], and mainly focus on the reconstruction of the user's historical interactions and the instruction output in the following.

To form the user's historical interactions for the ID facet, we convert the first $L-1$ items in $\mathcal{S}_u$ to their numeric IDs, and then separate each item with a semicolon. A sequence of ID-facet identifiers is denoted as "$P_1; P_2; \ldots; P_{L-1}$"[4]. Likewise, we can obtain the user's historical interactions in the title and attribute facets, referred to as "$T_1; T_2; \ldots; T_{L-1}$", and "$A_1; A_2; \ldots; A_{L-1}$", respectively. As for the instruction output, for the ID facet, we use the numeric ID of the last item in the user's interaction sequence, *i.e.,* P_L. For the title facet, we utilize substrings t with arbitrary length $l \in \{1, \ldots, |T|\}$, sampled from the title T. This is to encourage LLMs to generate from any positions that are possibly relevant to the user's interests. Here, for each user's interaction sequence, we sample K substrings of the last item's title and construct K instruction input-output pairs. Lastly, for the attribute facet, each attribute $a \in A_L$ is independently used as one instruction output, resulting in $|A_L|$ instruction input-output pairs. We denote the sets of the instruction data from ID, title, and attribute facets as $\mathcal{D}_{ID}$, $\mathcal{D}_{title}$, and $\mathcal{D}_{attr}$, respectively. Moreover, to explicitly distinguish different facet data, we add a facet prefix after the instruction input as shown in Figure 4.

Based on the reconstructed instruction data $\mathcal{D}_{instruct} = \mathcal{D}_{ID} \cup \mathcal{D}_{title} \cup \mathcal{D}_{attr}$, the LLM is optimized via Eq. (1). Note that we only employ a single LLM in TransRec for the instruction tuning.

3.2 Multi-facet Generation Grounding

After instruction tuning, the next step of TransRec is generation grounding (see Figure 5), which aims to deliver in-corpus item recommendations based on the user's historical interactions.

3.2.1 ***Position-free Constrained Generation.*** Out-of-corpus identifiers and over-reliance on the quality of initially generated tokens are two critical problems in the generation process. To tackle the issues, we consider LLMs to conduct position-free constrained generation. Remarkably, we introduce FM-index [12], a specialized data structure, that simultaneously supports position-free and constrained generation.

• **FM-index.** FM-index is a special prefix tree that supports search from any position [4]. This capability enables FM-index to 1) find all valid successor tokens of a given token; and 2) allow the generation to start from any token of the valid identifiers[5]. Specifically, taking the item in Figure 2(a) as an example, we flatten the multi-facet identifier as "<IDS> 1023 <IDE> Urban Decay Eyeshadow Palette Naked Heat <AS> Makeup <AE> <AS> Eyes <AE>", where "<IDS>, <IDE>, <AS>, <AE>" are the special tokens that indicate the start and the end of each ID and fine-grained attribute, respectively. The flattened identifier will then be stored in the Wavelet Tree [16]. Given a start token (*e.g.,* "BOS") or a token sequence, the FM-index can find a list of all possible successor tokens in $O(V \log(V))$, where V is the vocabulary size of the LLMs (refer to Appendix A.1 for detailed explanations).

• **Identifier generation.** Given the user's historical interactions in the format of instruction input as in Figure 4, TransRec generates valid identifiers in each facet via constrained beam search [10] based on the FM-index. Notably, the special tokens are utilized to indicate which facet is being generated. By constraining the starting token of generation, *e.g.,* "<IDS>", and the ending token of generation, *e.g.,* "<IDE>", TransRec generates a set of valid ID identifiers and

[3]Full templates can be found at https://github.com/Linxyhaha/TransRec/.
[4]For notation brevity, we omit the subscript u representing the user.
[5]Other potential Trie algorithms in existing constrained generation merely allow starting from the first token of valid identifiers [7, 19].

attribute identifiers that belongs to the items (see Figure 5). Besides, TransRec generates valid substrings of title identifiers from any position through FM-index, and thus we do not need to set special start and end tokens for the title. The position-free generation ability of LLMs is also enhanced by the instruction tuning process, where substrings are set as the instruction output (*cf.* Section 3.1.1). We follow [4] to keep track of all the partially decoded sequences and obtain a set of generated identifiers for each facet. We denote the generated identifiers for ID, title, and attribute facets as $\mathcal{P}_g$, $\mathcal{T}_g$, and $\mathcal{A}_g$, respectively.

3.2.2 ***Aggregated Grounding.*** To ground the generated identifiers to the in-corpus items and also rank in-corpus items for recommendations, we introduce an aggregated grounding module, which contains intra-facet and inter-facet aggregations.

• **Intra-facet aggregation.** We first ground the generated identifiers within each facet to the in-corpus items. Specifically, given the generated identifiers of one facet (*e.g.,* $\mathcal{T}_g$), we can aggregate them to the in-corpus item based on their coverage on each item's identifier (*e.g.,* $\mathcal{T}_g \cap T$). However, directly summing up the identifier scores[6] in the coverage set is infeasible due to the monotonic probability decrease in autoregressive generation [4]. For example, "51770" will have a smaller probability than "517", thus hindering the accurate grounding. To address this issue, we follow [4] to balance the scores by integrating token frequency:

$$s(\hat{y}) = \max\{0, \log \frac{P(\hat{y}|x)(1-P(\hat{y}))}{P(\hat{y})(1-P(\hat{y}|x))}\}, \tag{3}$$

where $\hat{y}$ is the generated identifier, $P(\hat{y}|x)$ is the token score given by the LLMs, and $P(\hat{y})$ is the unconditional probability that measures the frequency of tokens. For intuitive understanding, Eq. (3) can mitigate the issue by upweighting the tokens that are more distinctive, *i.e.,* less frequent. We obtain the unconditional probability $P(\hat{y})$ by

$$P(\hat{y}) = \frac{F(\hat{y}, \tilde{\mathcal{I}})}{\sum_{d \in \tilde{\mathcal{I}}} |d|}, \tag{4}$$

where $F(\hat{y}, \tilde{\mathcal{I}})$ represents the number of token $\hat{y}$ in the identifier corpus $\tilde{\mathcal{I}}$. Then, the grounding score for item i corresponding to user u in facet f is

$$s_f(u, i) = \sum_{\hat{y} \in \tilde{i}} (s(\hat{y}))^{\gamma}, \tag{5}$$

where $\tilde{i}$ is the multi-facet identifiers of the item, and γ is a hyper-parameter to control the strength of intra-facet grounding scores.

• **Inter-facet aggregation.** To aggregate the grounding scores from three facets, we should consider the disparate influence of each facet in different scenarios. For instance, when seeking books, the titles may become crucial, while for restaurants, the category (*e.g.,* Mexican cuisine) possibly takes precedence. As such, TransRec balances the strength of each facet for the final ranking. Formally, the final grounding score for user u and item i is obtained by:

$$s(u, i) = \sum_{f} s_f(u, i) + b_f, \tag{6}$$

where $f \in \{ID, title, attribute\}$, and b_f is the bias hyper-parameter that balances the strength between facets. According to the final grounding scores, we can obtain a ranking list of in-corpus items as in Figure 5 and return top-ranked items as recommendations.

• **Instantiation.** TransRec is a model-agnostic method that can be applied to any backbone LLMs and diverse tuning techniques. To investigate the feasibility of TransRec on LLMs with different sizes and architectures, we instantiate TransRec on two LLMs, *i.e.,* BART-large [30] and LLaMA-7B [50]. BART-large is an encoder-decoder model with 406M parameters, and LLaMA-7B is a decoder-only model with 7B parameters.

[6]Here, the identifier score is the probability of generated identifier given by LLMs.

4 EXPERIMENTS

In this section, we conduct experiments on three real-world datasets to answer the following research questions:

- **RQ1:** How does our proposed TransRec perform compared to both traditional and LLM-based recommenders?
- **RQ2:** How does TransRec perform under few-shot setting, on both warm- and cold-start recommendation?
- **RQ3:** How does each component of TransRec (*e.g.,* each facet of identifier, constrained generation, and aggregated grounding) affect its effectiveness?

4.1 Experimental Settings

4.1.1 ***Datasets.*** We conduct experiments on three popular benchmark datasets: 1) **Beauty** is the collection of user interactions with beauty products from Amazon review datasets[7]. 2) **Toys** is also one representative recommendation dataset drawn from Amazon review datasets, where each toy product has substantial meta information. 3) **Yelp**[8] is a popular restaurant dataset with rich user interactions on extensive dining places. For the three datasets, we assign each item a random unique numeric ID. We then use the assigned IDs, item titles, and item categories for the ID, title, and attribute facets for TransRec, respectively. Following [15], we adopt the leave-one-out strategy[9] to split the datasets into training, validation, and testing sets. In addition, we consider two training settings: 1) **full training** uses all users' interactions in the training set to train the models; and 2) **few-shot training** randomly selects N users' interactions to train the models, where $N = 1024$ or 2048. The statistics of the datasets are summarized in Table 9 in Appendix.

4.1.2 ***Baselines.*** We compare TransRec with both traditional recommenders (MF, LightGCN, SASRec, ACVAE, and DCRec) and LLM-based recommenders (P5, SID, SemID+IID, CID+IID, and TIGER). 1) **MF** [46] is one of the most representative collaborative filtering models, which decomposes the user-item interactions into the user and the item matrices. 2) **LightGCN** [20] is a graph-based model which linearly propagates the user and item representations from the neighborhood. 3) **SASRec** [27] is a representative sequential recommender model that adopts self-attention mechanism to learn the item dependency from user's interactions. 4) **ACVAE** [57] incorporates contrastive learning and adversarial training into the VAE-based sequential recommender model. 5) **DCRec** [58] employs contrastive learning and disentangles user interest and conformity to mitigate the popularity bias. 6) **P5** [15] is a unified

[7]https://jmcauley.ucsd.edu/data/amazon/.
[8]https://www.yelp.com/dataset.
[9]For each user's interactions, we use the last item as the testing item, the item before the last item as the validation item, and the rest as training items.

framework, which uses additional data (*e.g.,* user's reviews) for pre-training of LLMs on various tasks. We train P5 on sequential tasks for a fair comparison and assign random numeric IDs to items to prevent potential data leakage issue (*cf.* Appendix A.6). 7) **SID** [26] leverages collaborative information by sequential indexing. The items interacted consecutively by a user are assigned consecutive numeric IDs. 8) **SemID+IID** [26] designs the numeric ID based on the items' meta information such as attributes, where items with similar semantics have similar numeric IDs. 9) **CID+IID** [26] considers the co-occurrence matrix of items to design the numeric ID, where items that co-occur in user-item interactions will have similar numeric IDs. 10) **TIGER** [45] generates item identifier through a trainable codebook, which utilizes the item title and item descriptions to create new item tokens entailed by semantics.

- **Evaluation.** We adopt the widely used metrics Recall@K and NDCG@K to evaluate the models [41, 59], where K is 5 or 10.

4.1.3 ***Implementation Details.*** We employ BART and LLaMA as backbone LLMs for TransRec, and we denote the two variants as "TransRec-B" and "TransRec-L", respectively. For TransRec-B, we follow [15, 26] to sample subsequences of user's interactions for training, which is widely used in sequential recommender models [21]. As for LLaMA, the training on subsequences is involved in the training objectives of decoder-only architecture [66], and only uses the entire user sequence for training. Besides, for each user's interaction sequence, we iteratively discard the first item in the sequence until the length of instruction input does not exceed the maximum input length of LLMs (1024 for BART and 512 for LLaMA). TransRec is trained with Adam [28] (TransRec-B) and AdamW [42] (TransRec-L) on four NVIDIA RTX A5000 GPUs. We fully tune the model parameters of TransRec-B and perform the parameter-efficient fine-tuning technique LoRA [25] to tune TransRec-L. For a fair comparison, we set the beam size to 20 for TransRec and all LLM-based baselines. Detailed hyper-parameter settings for baselines and TransRec are presented in Appendix A.2.

4.2 Overall Performance (RQ1)

The results of the baselines and TransRec with BART as the backbone model under the full training setting are presented in Table 1, from which we have the following observations:

- Among traditional recommenders, sequential methods (SASRec, ACVAE, DCRec) surpass non-sequential methods (MF and LightGCN) on both Beauty and Toys. The better performance stems from the sequential modeling of the user's interaction sequence, which captures dynamic shifts in user interests and intricate item dependencies. Moreover, ACVAE usually outperforms other traditional recommenders. This is because adversarial training and contrastive learning encourage high-quality user representation and enhance the discriminability between items.
- CID+IID consistently yields better performance than SemID+IID, which is consistent with the findings in [26]. This is reasonable since CID+IID leverages the co-occurrence of items to construct hierarchical numeric IDs, *i.e.,* items with similar interactions have similar IDs. As such, the IDs are integrated with collaborative information, which strengthens the key advantage of ID-based identifiers. In contrast, SemID+IID simply constructs IDs based on items' meta information, *i.e.,* items with similar semantics have similar identifiers. However, this can lead to misalignment between item identifiers and user behavior, thus degrading the performance (*cf.* Section 1).
- TIGER usually achieves comparable performance to most of the traditional recommenders and surpasses SemID+IID on both Beauty and Toys. The better performance is attributed to 1) the additional utilization of description for capturing semantics; and 2) the learnable codebook to learn nuanced semantics compared to the manually defined semantic IDs (SemID+IID). Besides, P5 yields unsatisfactory performance on the three datasets. We believe that the inconsistency with the observations in [15] is due to the sequential indexing strategy, which potentially leads to data leakage [45].
- TransRec consistently yields the best performance across the three datasets, validating the superiority of our proposed transition paradigm. Notably, TransRec outperforms LLM-based recommenders by a large margin without requiring information on cold-start items to construct item identifiers. This further demonstrates the strong generalization ability of TransRec (see more analysis on the generalization of TransRec in Appendix A.7). The superiority of TransRec is attributed to 1) the utilization of multi-facet identifiers, which simultaneously satisfies semantics and distinctiveness to leverage the rich knowledge in LLMs and capture salient item features; and 2) the constrained and position-free generation that guarantees in-corpus item generation and mitigates the over-reliance on initial tokens.

4.3 In-depth Analysis

4.3.1 ***Few-shot Training (RQ2).*** To study how TransRec performs under limited data, we conduct few-shot training with randomly selected N users' interactions, where N is set as 1024 or 2048. To evaluate the models, we involve N users in few-shot training (*i.e.,* warm-start users) and another randomly selected N users that have not been seen in few-shot training (*i.e.,* cold-start users) for testing. In addition, we split the testing set into warm and cold sets, where interactions between warm-start users and warm-start items belong to the warm set, otherwise the cold set.

We compare both TransRec-B and TransRec-L with competitive baselines. The results on warm and cold sets over Beauty[10] are presented in Table 2. It is observed that 1) traditional methods yield competitive performance on warm-start recommendation. This is because these ID-based methods are effective in capturing collaborative information from user-item interactions. ACVAE yields better performance on cold sets, as it discards user embedding, enabling effective generalization for cold-start users. 2) Medium-sized LLMs (CID+IID and TransRec-B) struggle with few-shot training. This indicates that a considerable amount of data is necessary to adapt medium-sized LLMs to perform well on recommendation tasks, which is also consistent with previous work [1]. The better performance of CID+IID compared to TransRec-B possibly arises from the utilization of all item information to assign identifiers, as opposed to our strict reliance solely on warm items. 3) Notably, TransRec-L outperforms all the baselines, particularly surpassing by a large margin on the cold set. This

[10]Results on Yelp and Toys with similar observations are omitted to save space.

Table 1: Overall performance comparison between the baselines and TransRec instantiated on BART on three datasets. The best results are highlighted in bold and the second-best results are underlined. * implies the improvements over the second-best results are statistically significant (p-value < 0.01) under one-sample t-tests.

Model	Beauty				Toys				Yelp			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
MF	0.0294	0.0474	0.0145	0.0191	0.0236	0.0355	0.0153	0.0192	0.0220	0.0381	0.0138	0.0190
LightGCN	0.0305	0.0511	0.0194	0.0260	0.0322	0.0508	0.0215	0.0275	0.0255	0.0427	0.0163	0.0218
SASRec	0.0380	0.0588	0.0246	0.0313	0.0470	0.0659	0.0312	0.0373	0.0183	0.0296	0.0116	0.0152
DCRec	0.0452	0.0635	0.0327	0.0385	0.0498	0.0674	0.0335	0.0406	0.0207	0.0328	0.0115	0.0154
ACVAE	0.0503	0.0710	0.0356	0.0422	0.0488	0.0679	0.0350	0.0411	0.0211	0.0356	0.0127	0.0174
P5	0.0059	0.0107	0.0033	0.0048	0.0031	0.0069	0.0022	0.0034	0.0039	0.0062	0.0024	0.0031
SID	0.0350	0.0494	0.0254	0.0301	0.0164	0.0218	0.0120	0.0139	0.0218	0.0332	0.0161	0.0187
SemID+IID	0.0290	0.0429	0.0200	0.0245	0.0145	0.0260	0.0069	0.0123	0.0196	0.0304	0.0141	0.0160
CID+IID	0.0484	0.0703	0.0337	0.0412	0.0169	0.0276	0.0104	0.0154	0.0265	0.0417	0.0184	0.0233
TIGER	0.0377	0.0567	0.0249	0.0310	0.0278	0.0426	0.0176	0.0223	0.0183	0.0298	0.0119	0.0156
TransRec-B	**0.0504**	**0.0735***	**0.0365***	**0.0450***	**0.0518***	**0.0764***	**0.0360***	**0.0420***	**0.0354***	**0.0457***	**0.0262***	**0.0306***

Table 2: Performance comparison under the few-shot setting. The bold results highlight the superior performance compared to the best LLM-based recommender baseline. "TransRec-B" and "TransRec-L" denote using BART and LLaMA as backbone LLM, respectively.

N-shot	Model	Warm		Cold	
		R@5	N@5	R@5	N@5
1024	**LightGCN**	0.0205	0.0125	0.0005	0.0003
	ACVAE	0.0098	0.0057	0.0047	0.0026
	CID+IID	0.0100	0.0066	0.0085	0.0071
	TransRec-B	0.0042	0.0028	0.0029	0.0021
	TransRec-L	**0.0141**	**0.0070**	**0.0159**	**0.0097**
2048	**LightGCN**	0.0186	0.0117	0.0005	0.0004
	ACVAE	0.0229	0.0136	0.0074	0.0044
	CID+IID	0.0150	0.0101	0.0078	0.0062
	TransRec-B	0.0057	0.0031	0.0045	0.0026
	TransRec-L	**0.0194**	**0.0112**	**0.0198**	**0.0124**

highlights the remarkable generalization ability of recently emerged LLMs with vast knowledge base, enabling more effective adaptation to the recommendation task with limited data.

4.3.2 ***Ablation Study (RQ3).*** To analyze the effect of each facet in multi-facet identifiers, we remove the ID, title, and attribute separately, referred to as "w/o ID", "w/o title", and "w/o attribute", respectively. We also test TransRec with a single facet, *i.e.,* ID, title, and attribute, respectively. In addition, we disable the FM-index and conduct unconstrained generation, denoted as "w/o FM-index". Results on Beauty are presented in Figure 6. From the figure, we can find that: 1) removing either ID, title, or attribute facet will decrease the performance, indicating the effectiveness of each facet in representing an item for LLM-based recommendation. 2) Discarding the ID or title facet typically results in more significant performance reductions compared to removing the attribute facet. This is reasonable since removing IDs falls short of meeting distinctiveness criteria for identifiers, hindering LLMs from capturing salient features of items. Meanwhile, titles tend to possess more intricate semantics than attributes, exerting a more significant effect on enhancing recommendations. The crucial role of title facet is also indicated by the performance of TransRec with each single facet. 4) It is not surprising that removing FM-index fails to give

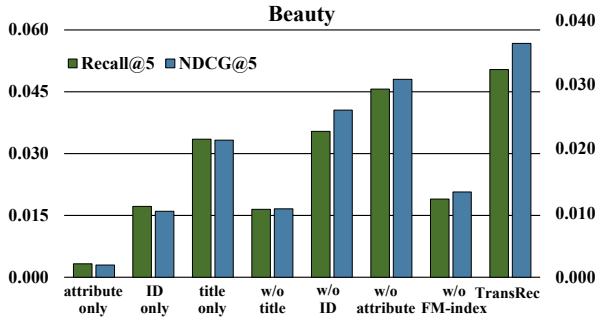

Figure 6: Ablation study of each facet (*i.e.,* ID, title, and attribute) of multi-facet identifier and the FM-index.

appropriate recommendations (inferior performance of "w/o FM-index"), because it may generate our-of-corpus identifiers. This implies the necessity of position-free constrained generation.

- **Effect of grounding strategies.** We also compare the aggregated grounding module of TransRec with three potential grounding strategies. Following [1], we utilize LLMs to extract the representations of generated identifiers and the identifiers of in-corpus items, respectively. We then calculate the dot product, the negative L2 distance, and the cosine similarity between the generated identifiers and each in-corpus item as the grounding score for each in-corpus item, respectively, as three strategies. From the results in Table 3, we can observe that 1) potential grounding strategies fail to yield satisfying results. This is because these strategies utilize the representations extracted from LLMs, thus relying heavily on the semantics similarity. As such, they may inaccurately ground the generated tokens that lack meaningful semantics to the valid identifiers. 2) TransRec is more time-efficient compared to other grounding strategies. Because the three strategies introduce extra LLMs' forward process for extracting the representations, causing high computation burdens.

4.3.3 ***Hyper-parameter Analysis.*** We further study the sensitivity of hyper-parameters to facilitate future applications of TransRec. The performance of TransRec with different values of γ and b_{ID} are presented in Figure 7. It is observed that 1) as γ varies from 1 to 3, the performance gradually improves. The possible reason is that the scales of original probabilities of LLMs are limited ($\gamma = 1$), leading

Table 3: Performance comparison of different grounding strategies. "Time" denotes the time cost of the grounding process per 5,000 users.

	Beauty				
	R@5↑	R@10↑	N@5↑	N@10↑	Time↓
Dot Product	0.0016	0.0020	0.0013	0.0014	571.07 s
Cosine Sim	0.0096	0.0121	0.0095	0.0101	578.89 s
L2 Distance	0.0201	0.0212	0.0148	0.0161	577.64 s
TransRec	**0.0504**	**0.0735**	**0.0365**	**0.0450**	**218.43 s**

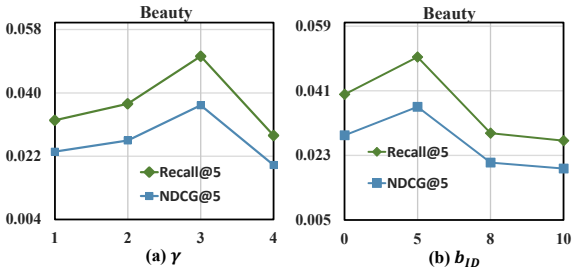

Figure 7: Effect of γ and b_{ID} in TransRec.

to insufficient discrepancies between identifiers and consequently reducing the informativeness of the ranking. 2) However, it is crucial not to indiscriminately increase γ, as this may lean towards recommendations of representative items. 3) We should carefully choose b_{ID} to balance the strength between facets since a small b_{ID} weakens the consideration of salient features while a large b_{ID} might undermine other facets, thus hurting recommendations. Analysis of b_{title} and $b_{attribute}$ are presented in Appendix A.8.

5 RELATED WORK

5.1 LLMs for Recommendation

Recently, leveraging LLMs for recommendation has received much attention due to LLMs' rich world knowledge, strong reasoning, and generalization abilities [5, 14, 31, 36, 53, 54, 64, 65]. Existing work on LLMs for recommendation can be mainly divided into two groups. 1) LLM-enhanced recommenders [6, 17, 23, 67], which consider LLMs as powerful feature extractors for enhancing the feature representation of users and items. Another line of research lies in 2) LLM-based recommenders [34, 44, 52], which directly leverage the LLMs as recommender systems. In the wake of ChatGPT's release and its remarkable prowess in reasoning, early studies delve into LLMs' zero-shot/few-shot recommendation capabilities through in-context learning [9, 24, 51]. However, due to the intrinsic gap between the LLMs' pre-training and recommendation tasks [6, 38], the recommendation abilities are somehow limited by merely using in-context learning [40]. Therefore, to narrow the gap and elicit the strong capabilities of LLMs, recent studies utilize instruction tuning for LLMs to improve the performance [2, 39, 62]. In this work, we highlight the two fundamental steps of LLM-based recommenders to bridge the item and language and spaces, showing existing problems and the key objectives in the two steps.

5.2 Item Indexing and Generation Grounding

To bridge the item space and the language space, the two vital steps are item indexing and generation grounding. Previous indexing methods can be categorized into two groups. 1) ID-based identifiers represent each item by a unique numeric ID, to learn salient features from user-item interactions [26, 33, 60]. 2) Description-based identifiers employ item descriptions to represent an item [8, 62]. Nevertheless, ID-based identifiers lack semantics, while identifier-based identifiers lack adequate distinctiveness. To alleviate the issues, [26, 45] propose to design IDs based on descriptions, which however is essentially a description-based identifier and yields less satisfying recommendations (*cf.* Section 4.2). Although [61] and [32] compare the two types of identifiers independently in a discriminative manner, TransRec employs the generative approach to leverage both semantics and distinctiveness for effective item indexing. A concurrent study [35] also explores multiview identifiers in passage ranking. However, it solely considers the semantics to strengthen the correlation between query and passages. In contrast, our study takes the initial endeavor to investigate multi-facet identifiers in LLM-based recommendation, highlighting the distinctiveness as a criterion for identifier design to capture the crucial CF knowledge.

The generation grounding step involves the generation of token sequences and the grounding to the in-corpus items. However, this step has received little scrutiny in previous studies. They mainly conduct autoregressive generation over the whole vocabulary and ground the generated tokens by exact matching, which may lead to out-of-corpus recommendations. [8] and [1] mitigate the out-of-corpus issue by utilizing the representations obtained by LLMs to match between generated tokens and in-corpus items, *e.g.*, L2 distance. However, these approaches inevitably suffer from high computational burdens as they require extra computations for representation extraction. Different from previous work, we propose an effective generation grounding step, which utilizes a specialized data structure to achieve guaranteed in-corpus and enhanced recommendations.

6 CONCLUSION AND FUTURE WORK

In this work, we highlighted the two fundamental steps for LLM-based recommenders: item indexing and generation grounding. To make full utilization of LLMs and strengthen the generalization ability of LLM-based recommenders, we posited that item identifiers should pursue distinctiveness and semantics. In addition, we considered position-free constrained generation for LLMs to yield accurate recommendations. To pursue these objectives, we proposed a novel transition paradigm, namely TransRec, to seamlessly bridge the language space and the item space. We utilized multi-facet identifiers to represent an item from ID, title, and attribute facets simultaneously. Besides, we employed the FM-index to guarantee high-quality generated identifiers. Furthermore, we introduced an aggregated grounding module to ground the generated identifiers to the items. Empirical results on three real-world datasets under diverse settings validated the superiority of TransRec in improving recommendation accuracy and generalization ability.

This work highlights the key objectives of indexing approaches and generation grounding strategies, leaving many promising directions for future exploration. Particularly, 1) although incorporating ID, title, and attribute is effective, it is worthwhile to automatically construct multi-facet identifiers to reduce the noises in natural descriptions; and 2) it is meaningful to devise better strategies for grounding modules, to effectively combine the ranking scores from different facets, such as using neural models in a learnable manner.

REFERENCES

[1] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Fuli Feng, Xiangnaan He, and Qi Tian. 2023. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv:2308.08434*.

[2] 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. In *RecSys*. ACM.

[3] William Berrios, Gautam Mittal, Tristan Thrush, Douwe Kiela, and Amanpreet Singh. 2023. Towards language models that can see: Computer vision through the lens of natural language. *arXiv:2306.16410*.

[4] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *NeurIPS* 35 (2022), 31668–31683.

[5] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2022. Language models are realistic tabular data generators. *arXiv:2210.06280*.

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

[7] Zhixuan Chu, Hongyan Hao, Xin Ouyang, Simeng Wang, Yan Wang, Yue Shen, Jinjie Gu, Qing Cui, Longfei Li, Siqiao Xue, et al. 2023. Leveraging large language models for pre-trained recommender systems. *arXiv:2308.10837*.

[8] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv:2205.08084*.

[9] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's Capabilities in Recommender Systems. *arXiv:2305.02182*.

[10] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. *arXiv:2010.00904*.

[11] Yang Deng, Yaliang Li, Wenxuan Zhang, Bolin Ding, and Wai Lam. 2022. Toward personalized answer generation in e-commerce via multi-perspective preference modeling. *TOIS* 40, 4 (2022), 1–28.

[12] Paolo Ferragina and Giovanni Manzini. 2000. Opportunistic data structures with applications. In *Proceedings 41st annual symposium on foundations of computer science*. IEEE, 390–398.

[13] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv:2303.14524*.

[14] Shijie Geng, Zuohui Fu, Juntao Tan, Yingqiang Ge, Gerard De Melo, and Yongfeng Zhang. 2022. Path language modeling over knowledge graphsfor explainable recommendation. In *WWW*. ACM, 946–955.

[15] 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 *RecSys*. ACM, 299–315.

[16] Roberto Grossi, Ankur Gupta, and Jeffrey Scott Vitter. 2003. High-order entropy-compressed text indexes. In *SIAM*. Society for Industrial and Applied Mathematics Philadelphia.

[17] Taicheng Guo, Lu Yu, Basem Shihada, and Xiangliang Zhang. 2023. Few-shot News Recommendation via Cross-lingual Transfer. In *WWW*. ACM, 1130–1140.

[18] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *ICML*. PMLR, 3929–3938.

[19] Jie Hao, Yang Liu, Xing Fan, Saurabh Gupta, Saleh Soltan, Rakesh Chada, Pradeep Natarajan, Chenlei Guo, and Gökhan Tür. 2022. CGF: Constrained generation framework for query rewriting in conversational AI. In *EMNLP*. ACL, 475–483.

[20] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*. 639–648.

[21] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *ICLR*.

[22] Teakgyu Hong, Donghyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park. 2022. Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents. In *AAAI*, Vol. 36. AAAI press, 10767–10775.

[23] 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*. ACM, 585–593.

[24] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2023. Large language models are zero-shot rankers for recommender systems. *arXiv:2305.08845*.

[25] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv:2106.09685*.

[26] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2023. How to Index Item IDs for Recommendation Foundation Models. *arXiv:2305.06569*.

[27] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*. IEEE, 197–206.

[28] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *arXiv:1412.6980*.

[29] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *NeurIPS* 35 (2022), 22199–22213.

[30] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv:1910.13461*.

[31] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text Is All You Need: Learning Language Representations for Sequential Recommendation. In *KDD*. ACM, 1258–1267.

[32] 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:2305.11700*.

[33] Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. 2023. CTRL: Connect Tabular and Language Model for CTR Prediction. *arXiv:2306.02841*.

[34] Yongqi Li, Xinyu Lin, Wenjie Wang, Fuli Feng, Liang Pang, Wenjie Li, Liqiang Nie, Xiangnan He, and Tat-Seng Chua. 2024. A Survey of Generative Search and Recommendation in the Era of Large Language Models. *arXiv:2404.16924*.

[35] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview Identifiers Enhanced Generative Retrieval. In *ACL*. ACL.

[36] Yi Li, Jieming Zhu, Weiwen Liu, Liangcai Su, Guohao Cai, Qi Zhang, Ruiming Tang, Xi Xiao, and Xiuqiang He. 2022. PEAR: Personalized Re-ranking with Contextualized Transformer for Recommendation. In *WWW*. ACM, 62–66.

[37] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *ICRA*. IEEE, 9493–9500.

[38] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, et al. 2023. How Can Recommender Systems Benefit from Large Language Models: A Survey. *arXiv:2306.05817*.

[39] Xinyu Lin, Wenjie Wang, Yongqi Li, Shuo Yang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2024. Data-efficient Fine-tuning for LLM-based Recommendation. In *SIGIR*. ACM.

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

[41] Weiming Liu, Xiaolin Zheng, Chaochao Chen, Jiajie Su, Xinting Liao, Mengling Hu, and Yanchao Tan. 2023. Joint Internal Multi-Interest Exploration and External Domain Alignment for Cross Domain Sequential Recommendation. In *WWW*. ACM, 383–394.

[42] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv:1711.05101*.

[43] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *NeurIPS* 35 (2022), 27730–27744.

[44] Bo Peng, Ben Burns, Ziqi Chen, Srinivasan Parthasarathy, and Xia Ning. 2024. Towards Efficient and Effective Adaptation of Large Language Models for Sequential Recommendation. *arXiv:2310.01612*.

[45] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. 2023. Recommender Systems with Generative Retrieval. In *NeurIPS*. Curran Associates, Inc.

[46] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. AUAI Press, 452–461.

[47] Michael Schindler. 1997. A fast block-sorting algorithm for lossless data compression. In *Proc. Data Compression Conf*, Vol. 469. Citeseer.

[48] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *ACL*. ACL, 1715–1725.

[49] Yu Tian, Jianxin Chang, Yanan Niu, Yang Song, and Chenliang Li. 2022. When multi-level meets multi-interest: A multi-grained neural model for sequential recommendation. In *SIGIR*. ACM, 1632–1641.

[50] 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:2302.13971*.

[51] Lei Wang and Ee-Peng Lim. 2023. Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. *arXiv:2304.03153*.

[52] Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Learnable Tokenizer for LLM-based Generative Recommendation. *arXiv:2405.07314*.

[53] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *KDD*. ACM, 1929–1937.

[54] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *SIGIR*. ACM, 1652–1656.

[55] Chuhan Wu, Fangzhao Wu, Tao Qi, Chao Zhang, Yongfeng Huang, and Tong Xu. 2022. Mm-rec: Visiolinguistic model empowered multimodal news recommendation. In *SIGIR*. ACM, 2560–2564.
[56] Yueqi Xie, Peilin Zhou, and Sunghun Kim. 2022. Decoupled side information fusion for sequential recommendation. In *SIGIR*. ACM, 1611–1621.
[57] Zhe Xie, Chengxuan Liu, Yichi Zhang, Hongtao Lu, Dong Wang, and Yue Ding. 2021. Adversarial and contrastive variational autoencoder for sequential recommendation. In *WWW*. ACM, 449–459.
[58] Yuhao Yang, Chao Huang, Lianghao Xia, Chunzhen Huang, Da Luo, and Kangyi Lin. 2023. Debiased Contrastive Learning for Sequential Recommendation. In *WWW*. ACM.
[59] 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 *SIGIR*. ACM.
[60] Fajie Yuan, Guoxiao Zhang, Alexandros Karatzoglou, Joemon Jose, Beibei Kong, and Yudong Li. 2021. One person, one model, one world: Learning continual user representation without forgetting. In *SIGIR*. 696–705.
[61] 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. In *SIGIR*. 2639–2649.
[62] 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:2305.07001*.
[63] Qi Zhang, Jingjie Li, Qinglin Jia, Chuyuan Wang, Jieming Zhu, Zhaowei Wang, and Xiuqiang He. 2021. UNBERT: User-News Matching BERT for News Recommendation.. In *IJCAI*. 3356–3362.
[64] Zizhuo Zhang and Bang Wang. 2023. Prompt learning for news recommendation. In *SIGIR*. ACM, 227–237.
[65] Qihang Zhao. 2022. RESETBERT4Rec: A pre-training model integrating time and user historical behavior for sequential recommendation. In *SIGIR*. ACM, 1812–1816.
[66] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv:2303.18223*.
[67] Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. 2021. Pre-trained language model based ranking in Baidu search. In *KDD*. ACM, 4014–4022.

A APPENDIX

A.1 FM-index

FM-index is a compressed suffix array. Given the original text, *e.g.,* item title, it is first transformed into a matrix based on the Burrows-Wheeler Transform (BWT) [47], which turns the original text into a matrix sorted lexicographically. For example, given a string $XYXZ$, the transformed matrix is:

$	X	Y	X	Z
X	Y	X	Z	$
X	Z	$	X	Y
Y	X	Z	$	X
Z	$	X	Y	X,

Table 4: Example of transformed matrix based on BWT.

As shown in the example, the first column is the repeated token sorted lexicographically, and the last column is called the string's BWT. Based on the matrix, it can then access every token through self-indexing and can find all valid token successors. Notably, only the first and last columns are explicitly stored in the FM-index, contributing to its space efficiency. Then, the storage of these columns is managed by the Wavelet Tree, a hierarchical structure that employs wavelet transformations for compact encoding and rapid querying of the FM-index. This integrated approach enhances the overall performance and scalability of the data structure. More detailed explanations can also be found in [4].

A.2 Hyper-parameter Settings

The best hyper-parameters of models are selected by Recall on the validation set. For the traditional recommender models, we search the embedding size, learning rate, and weight decay from $\{16, 32, 64, 128, 256\}$, $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$, and $\{1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$, respectively. The search scopes for some model-specific hyper-parameters are as follows. For LightGCN, we tune the number of GCN layers and the dropout ratio in $\{1, 2, 3, 4, 5\}$ and $\{0.1, 0.3, 0.5\}$, respectively. We search the training sequence length in $\{10, 20, 50, 100\}$ for both SASRec and ACVAE. For SASRec, we select the number of attention blocks in $\{1, 2, 3\}$ and the number of attention heads in $\{1, 2, 4, 8\}$. For ACVAE, the weight of contrastive loss term β is searched in $\{0.1, 0.3, 0.5, 0.7\}$. For LLM-based recommenders [26, 45], we follow the searching scopes stated in their papers. For TransRec, we set the sampling number of substrings K as 5. The scaling factor for the intra-facet grounding scores γ and the bias terms for the intra-facet aggregation b_{ID}, b_{title}, and $b_{attribute}$ are searched in ranges of $\{1, 2, 3, 4\}$, $\{0, 5, 8, 10\}$,$\{-2, 0, 2, 5\}$, and $\{2, 5, 7, 10\}$, respectively.

A.3 Performance of TransRec on T5-small

To achieve fair comparisons, we employ a consistent backbone LLM and prompt template across all LLM-based methods. Specifically, we employ T5-small to instantiate SID, SemID+IID, CID+IID, and TransRec. Besides, to avoid the bonus from multiple prompt templates, we implement all methods with the same single prompt. From the results in Table 5, we can find that TransRec outperforms baselines by a large margin, which demonstrates the effectiveness of multi-facet identifier and generation grounding.

Table 5: Performance comparison between TransRec and various indexing methods with the same backend model T5.

	Beauty			
	Recall@5	**Recall@10**	**NDCG@5**	**NDCG@10**
SID	0.0113	0.0215	0.0069	0.0101
SemID+IID	0.0089	0.0192	0.0056	0.0090
CID+IID	0.0125	0.0230	0.0078	0.0112
TransRec-T5	0.0325	0.0493	0.0224	0.0278

A.4 Cost Analysis of FM-index

We analyze the storage and time costs of FM-index with results presented in Table 6 and Table 7, respectively. From the tables, we can find that both the storage costs, time costs, and update costs are trivial, facilitating the real-world application of FM-index.

Table 6: Storage and time costs of creating FM-index.

# Item	**10k**	**20k**	**30k**	**40k**	**50k**	**100k**
Storage (MB)	1.2	1.8	3.8	5.0	6.2	13.0
Time costs (s)	6.2	11.9	18.3	23.6	28.7	55.2

Table 7: Time costs of adding 1k items to FM-index.

# Existing item	**10k**	**20k**	**30k**	**40k**	**50k**	**100k**
Time costs (s)	1.1	1.1	1.6	1.6	2.0	3.8

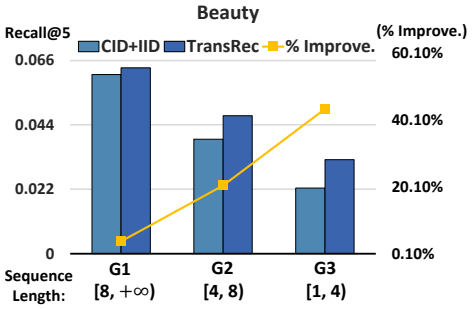

Figure 8: Performance over user groups with different lengths of historical interaction sequence.

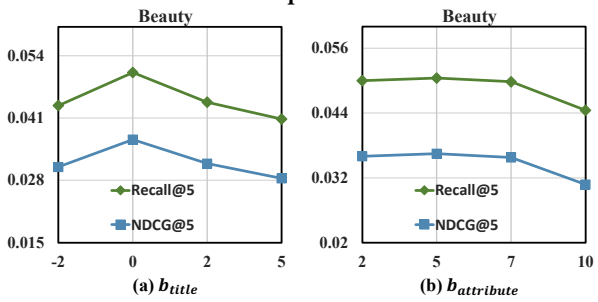

Figure 9: Effect of b_{title} and $b_{attribute}$ in TransRec.

Table 9: Statistics of three datasets.

Dataset	# User	# Item	# Interaction	Density (%)
Beauty	22,363	12,101	198,502	0.0734
Toys	19,412	11,924	167,597	0.0724
Yelp	30,431	20,033	316,354	0.0519

A.5 Constrained Generation with Trie

We employ Trie, a data structure that supports generating valid tokens strictly from the first token of the item identifier during constrained generation. From the results in Table 8, we can find that generating from any position outperforms generating from the first token, which is due to potential misalignment between the first token and the user preference. This validates the effectiveness of position-free generation supported by FM-index.

Table 8: Performance of constrained generation with Trie.

	Beauty			
	Recall@5	**Recall@10**	**NDCG@5**	**NDCG@10**
Trie	0.0186	0.0274	0.0118	0.0146
FM-index	**0.0504**	**0.0735**	**0.0365**	**0.0450**

A.6 Potential Data Leakage Issue of P5

As discussed in [45], the original sequential indexing method utilized in P5 [15] suffers from the potential data leakage issue. Specifically, P5 assigns consecutive numeric IDs to the interacted items in the user sequence, where the items in training and testing sets have a high probability of sharing the same token after tokenization. For instance, P5 may represent a user sequence as $[7391, 7392, \ldots, 7398, 7399]$, where 7399 is in the testing set. Based on the SentencePiece tokenizer [48], these numeric IDs will be tokenized into "73" and "91", "73" and "92", and so forth. As such, the item identifiers in the training and the testing sets will share the same token "73". This can lead to strong correlations between the historical interactions and next-interacted items, thus significantly benefiting the prediction accuracy of the testing item. However, such benefits are from the consecutive numbers, which are unattainable during the indexing process in real-world scenarios. To solve this issue, we adopt the datasets in P5 for experiments but rearrange the numeric ID for items with random numeric ID instead of consecutive IDs for both our method and P5.

A.7 User Group Evaluation

To analyze how TransRec improves the performance and the generalization ability of LLM-based recommenders, we test the performance of TransRec over different sparsity of users and compare it with the competitive baseline CID+IID. Specifically, we divide the users into three groups according to the sequence length of historical interactions. We select users with interactions larger or equal to 8 into group 1, denoted as "G1"; and then split the rest of the users with interactions larger or equal to 4 into group 2, otherwise group 3, denoted as "G2", and "G3", respectively. As such, from G1 to G3, the user sparsity increases. The results of the three groups on Beauty are presented in Figure 8. We can find that: 1) From G1 to G3, the performance of both CID+IID and TransRec decreases. This makes sense because it can be difficult to capture the user preference shifts from only a small number of interactions. Nevertheless, 2) TransRec consistently outperforms CID+IID under different levels of user sparsity. In particular, TransRec improves the performance of sparse users remarkably by a large margin (significant improvements on "G3"), indicating the strong generalization ability of TransRec.

A.8 Hyper-parameter Analysis

- **Effect of b_{title}.** We vary the bias for the title facet *i.e.,* b_{title}, and present the results in Figure 9. It is observed that TransRec achieves the best performance when $b_{title} = 0$, indicating that bias for the title facet may not necessarily need careful adjustment. One possible reason is that titles usually contain common words, resulting in a mild gap between the pre-training data and the titles. In contrast, IDs that are less common in pre-training data probably lead to a larger gap between the pre-training data and the identifiers, thereby requiring a larger bias to improve the strength of the ID facet. This is also evidenced by Figure 7(b), where TransRec achieves the best performance when $b_{ID} = 5$.
- **Effect of $b_{attribute}$.** The results of different bias values for the attribute facet $b_{attribute}$ are reported in Figure 9(b). From the results, we can find that 1) gradually increasing the bias for the attribute facet does not affect the performance too much, indicating that TransRec might be less sensitive to the attribute strength. This is reasonable since attribute entails some coarse-grained semantics, such as category, and color, which can be shared by extensive items. 2) Similar to the ID facet, strengthening either the attribute or title facet too much can hurt the performance. The reason is that letting the title or attribute facet dominate the grounding can decrease item discriminability.