How Can Recommender Systems Benefit from Large Language Models: A Survey

Jianghao Lin[1*], **Xinyi Dai**[2*], **Yunjia Xi**[1], **Weiwen Liu**[2], **Bo Chen**[2], **Xiangyang Li**[2], **Chenxu Zhu**[2], **Huifeng Guo**[2], **Yong Yu**[1], **Ruiming Tang**[2†], **Weinan Zhang**[1†]

[1]Shanghai Jiao Tong University, Shanghai, China
[2]Huawei Noah's Ark Lab, Shenzhen, China
{chiangel,xiyunjia,yyu,wnzhang}@sjtu.edu.cn,
{daixinyi3,liuweiwen8,chenbo116,lixiangyang34,zhuchenxu1,huifeng.guo,tangruiming}@huawei.com

arXiv:submit/4945443 [cs.IR] 9 Jun 2023

Abstract

Recommender systems (RS) play important roles to match users' information needs for Internet applications. In natural language processing (NLP) domains, large language model (LLM) has shown astonishing emergent abilities (*e.g.*, instruction following, reasoning), thus giving rise to the promising research direction of adapting LLM to RS for performance enhancements and user experience improvements. In this paper, we conduct a comprehensive survey on this research direction from an application-oriented view. We first summarize existing research works from two orthogonal perspectives: where and how to adapt LLM to RS. For the **"WHERE"** question, we discuss the roles that LLM could play in different stages of the recommendation pipeline, *i.e.*, feature engineering, feature encoder, scoring/ranking function, and pipeline controller. For the **"HOW"** question, we investigate the training and inference strategies, resulting in two fine-grained taxonomy criteria, *i.e.*, whether to tune LLMs or not, and whether to involve conventional recommendation model (CRM) for inference. Detailed analysis and general development trajectories are provided for both questions, respectively. Then, we highlight key challenges in adapting LLM to RS from three aspects, *i.e.*, efficiency, effectiveness, and ethics. Finally, we summarize the survey and discuss the future prospects. We also actively maintain a GitHub repository for papers and other related resources in this rising direction[1].

1 Background

With the rapid development of online services, recommender systems (RS) have become increasingly important to match users' information needs [Dai *et al.*, 2021; Fu *et al.*, 2023b] and alleviate the problem of information overloading [Guo *et al.*, 2017]. Despite the different forms of application tasks (*e.g.*, top-N recommendation, or sequential recommendation), the common learning objective for a deep learning based recommender system is to estimate a given user's preference towards each candidate item, and finally arrange a ranked list of items presented to the user [Lin *et al.*, 2021; Xi *et al.*, 2023].

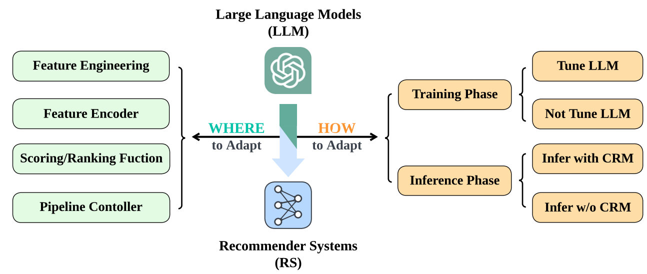

Figure 1: The decomposition of the research question about adapting large language models to recommender systems. We analyze the question from two orthogonal perspectives: (1) where to adapt LLM, and (2) how to adapt LLM.

On the other hand, in the field of natural language processing (NLP), large language model (LLM) has shown impressive emergent abilities (*e.g.*, reasoning [Huang and Chang, 2022], in-context few-shot learning [Brown *et al.*, 2020]), as well as the vast reservoir of open-world knowledge compressed in their pretrained model parameters [Zhao *et al.*, 2023]. While LLM is making remarkable breakthroughs in various deep learning applications, it is natural to propose the following research question:

How can recommender systems benefit from large language models for performance enhancements and user experience improvements?

In this paper, we aim to conduct an in-depth survey on the adaption of large language models to recommender systems. We study this research question from an application-oriented view and cover a broad range of the latest research works, which we argue is valuable and instructive for recommender system developments. As shown in Figure 1, we comprehensively analyze the latest research progresses, and decompose the research question above from two perspectives:

- **"WHERE"** question focuses on where to adapt LLM for

[1]https://github.com/CHIANGEL/Awesome-LLM-for-RecSys
* Jianghao Lin and Xinyi Dai are the co-first authors.
† Ruiming Tang and Weinan Zhang are the co-corresponding authors.

RS, and discusses the roles that LLM could play at different parts of the modern deep learning based recommender system pipeline, *i.e.*, feature engineering, feature encoder, scoring/ranking function, and pipeline controller.

- **"HOW"** question centers on how to adapt LLM for RS, where two orthogonal taxonomy criteria are carried out: (1) whether we will freeze the parameters of the large language model during the training phase, and (2) whether we will involve conventional recommendation models (CRM) during the inference phase.

Moreover, we would like to make two further statements before we move on to the details of this survey paper:

- To provide a thorough survey and a clear development trajectory, we broaden the scope of large language models, and bring those relatively smaller language models (*e.g.*, BERT [Devlin *et al.*, 2018], GPT2 [Radford *et al.*, 2019]) into the discussion as well.
- We focus on works that leverage LLM together with their pretrained parameters to handle textual features via prompting, and exclude works that simply apply pretraining paradigms from NLP domains to pure ID-based traditional recommendation models (*e.g.*, BERT4Rec [Sun *et al.*, 2019]). Interested readers can refer to [Yu *et al.*, 2022a; Liu *et al.*, 2023b].

The rest of this paper is organized as follows. Section 2 and Section 3 thoroughly analyze the aforementioned taxonomies from two perspectives (*i.e.*, **"WHERE"** and **"HOW"**), followed by detailed discussion and analysis of the general development trajectories. In Section 4, we highlight five key challenges for the adaption of LLM to RS from three aspects (*i.e.*, **efficiency**, **effectiveness**, and **ethics**), which mainly arise from the unique characteristics of recommender systems especially in industrial applications. Finally, Section 5 concludes this survey and draws a hopeful vision for future prospects in research communities of LLM-enhanced recommender systems.

2 Where to Adapt LLM

To answer the **"WHERE"** question about adapting LLM to the recommendation domain, we first analyze the pipeline of modern deep learning based recommender systems, and abstract it into several key components as follows:

- **User data collection** collects users' explicit (ratings) or implicit (click signals) feedback from online services by presenting recommended items to users.
- **Feature engineering** is the process of selecting, manipulating, transforming, and augmenting the raw data collected online into structured data (*e.g.*, one-hot encoding).
- **Feature encoder** takes as input the structured data, and generates the neural embeddings for scoring/ranking functions in the next stage. In most cases, it is formulated as the embedding layer for one-hot encoded categorical features.
- **Scoring/Ranking function** is the core part of recommendation, where various neural methods are designed to select the top-relevant items to satisfy users' information needs.

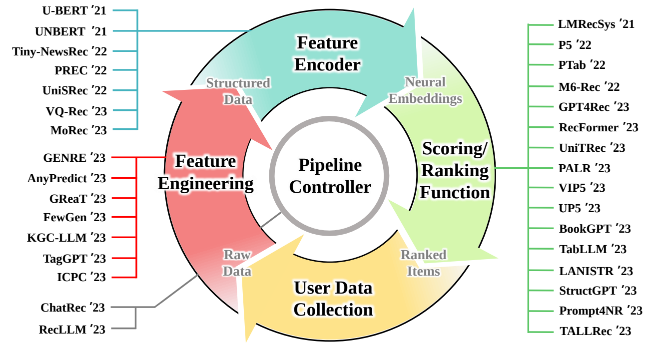

Figure 2: The illustration of deep learning based recommender system pipeline. We list representative works that adapt LLM to different parts of the system pipeline denoted by different colors.

- **Pipeline controller** monitors and controls the operations of the recommendation pipeline mentioned above. It can even provide fine-grained control over different stages for recommendation (*e.g.*, matching, ranking, reranking)

Next, we will elaborate on the adaptation of LLM to different parts of the recommendation pipeline, except for user data collection.

2.1 LLM for Feature Engineering

In the feature engineering stage, LLM takes as input the original data (*e.g.*, user profiles, item descriptions), and generates auxiliary textual features as data augmentations, where prompting and templating techniques are involved to extract the open-world knowledge and reasoning ability from the LLM. GReaT [Borisov *et al.*, 2023] tunes a generative language model to synthesize realistic tabular data as augmentations for the training phase. Carranza *et al.* [2023] explore to train a differentially private (DP) large language model for synthetic user query generation, in order to address the privacy problem in recommender systems. GENRE [Liu *et al.*, 2023c] applies manually designed prompts to obtain additional news summarization, user profiles, and synthetic news pieces for news recommendation. AnyPredict [Wang *et al.*, 2023] leverages LLM to consolidate datasets with different feature fields, and align out-domain datasets for a shared target task. Other works also utilize LLM to further enrich the training data from different perspectives, e.g., knowledge graph completion [Chen *et al.*, 2023], tag generation [Li *et al.*, 2023a], and user interest modeling [Christakopoulou *et al.*, 2023].

2.2 LLM as Feature Encoder

In conventional recommender systems, the structured data are usually formulated as one-hot encodings, and a simple embedding layer is adopted as the feature encoder to obtain dense embeddings. With the emergence of language models, researchers propose to adopt LLM as auxiliary textual feature encoders to gain two major benefits: (1) further enriching the user/item representations with semantic information for the later neural recommendation models; (2) achieving cross-domain[2] recommendation with natural language as the bridge, where feature fields might not be shared.

For item representation enhancement, LLM is leveraged as feature encoders for scenarios with abundant textual features available (*e.g.*, item title, textual body, description), including but not limited to: document ranking [Zou *et al.*, 2021; Liu *et al.*, 2021], news recommendation [Zhang *et al.*, 2021a; Wu *et al.*, 2021; Wu *et al.*, 2022; Yu *et al.*, 2022b; Liu *et al.*, 2022b], tweet search [Zhang *et al.*, 2022], tag selection [He *et al.*, 2022], and code example recommendation [Rahmani *et al.*, 2023]. As for user-side enrichment, U-BERT [Qiu *et al.*, 2021] ameliorates the user representation by encoding review texts into dense vectors via BERT.

Apart from user/item representation improvement, adopting LLM as feature encoders also enables transfer learning and cross-domain recommendation, where natural language serves as the bridge to link the heterogeneous information from different domains. ZESRec [Ding *et al.*, 2021] applies BERT to convert item descriptions into universal continuous representations for zero-shot recommendation. In UniSRec [Hou *et al.*, 2022], the item representations are learned for cross-domain sequential recommendation via a fixed BERT model followed by a lightweight MoE-enhanced network. Built upon UniSRec, VQ-Rec [Hou *et al.*, 2023a] introduces vector quantization techniques to better align the textual embeddings generated by LLMs to the recommendation space. Fu *et al.* [2023a] further explore layerwise adaptor tuning on the language model to obtain better embeddings over textual features from different domains.

2.3 LLM as Scoring/Ranking Function

In the stage of scoring/ranking, the ultimate goal of LLM is to provide a ranked list of items $[i_k]_{k=1}^N, i_k \in \mathcal{I}$, where $\mathcal{I}$ is the universal item set (next item prediction is a special case where $N = 1$). Such a goal could be achieved by various kinds of tasks specially designed for LLM (*e.g.*, rating prediction, item ID generation). According to different tasks to be solved by LLM, we classify them into three categories: (1) item scoring task, (2) item generation task, and (3) hybrid task.

Item Scoring Task

In item scoring tasks, the large language model serves as a pointwise function $F(u, i), \forall u \in \mathcal{U}, \forall i \in \mathcal{I}$, which estimates the score of each candidate item i for the target user u. Here $\mathcal{U}$ and $\mathcal{I}$ denote the universal set of users and items, respectively. The final ranked list of items is obtained by sorting the score, requiring N forwards of function $F(u, i)$:

$$[i_k]_{k=1}^N = \mathrm{Sort}\left(\{F(u, i_k)\}_{k=1}^N\right). \quad (1)$$

PTab [Liu *et al.*, 2022a] models the prediction task as a text classification problem, and tunes the language model based on pure textual inputs generated by prompting. Kang *et al.* [2023] finetune large language model for rating prediction in a regression manner, which exhibit a surprising performance by scaling the model size of finetuned LLM up to 11 billion. RecFormer [Li *et al.*, 2023b] estimates the matching score between the semantic representation of user interaction sequence and candidate items, respectively. Another line of research intends to concatenate the item description (*e.g.*, title) to the user behavior history with different prompts, and estimates the score as the overall perplexity [Mao *et al.*, 2023], log-likelihood [Sileo *et al.*, 2022], or joint probability [Zhang *et al.*, 2021b] of the prompting text.

The methods mentioned above generally follow the conventional paradigm of recommendation models, where the output of LLM is fed into a delicately designed projection layer to calculate the final score for classification or regression tasks. Recently, researchers also propose to enable LLM to directly output the score or user's preference towards a target item in natural language manners (*e.g.*, integers 1-5 for rating, yes/no for preference). Prompt4NR [Zhang and Wang, 2023] transforms the score estimation into a cloze [MASK] prediction task for binary key answer words (*e.g.*, related/unrelated, good/bad) with multi-prompt ensembling. TabLLM [Hegselmann *et al.*, 2023] and TALLRec [Bao *et al.*, 2023] train the decoder-only LLM to follow instructions and answer a binary question appended behind the contextual prompting information. Zhiyuli *et al.* [2023] instruct LLM to predict the user rating in textual manner, and restrict the output format as a value with two decimal places through manually designed prompts.

Item Generation Task

In item generation tasks, the large language model serves as a generative function $F(u)$ to directly produce the final ranked list of items, requiring only one forward of function $F(u)$:

$$[i_k]_{k=1}^N = F(u),\ s.t.\ i_k \in \mathcal{I},\ k = 1, \cdots, N. \quad (2)$$

GPT4Rec [Li *et al.*, 2023c] tunes a large language model to produce the title of next item according to the user's behavior history via multi-beam generation. VIP5 [Geng *et al.*, 2023] frames the next item recommendation task as a generative task, and utilizes a sequence-to-sequence model to generate the index of the next recommended item. Hua *et al.* [2023b] also explore better ways for item indexing (*e.g.*, sequential indexing, collaborative indexing) in order to enhance the performance of such index generation tasks. Chen [2023], Wang and Lim [2023] and Hou *et al.* [2023b] include a pre-filtered set of item candidates in the input prompts and apply LLM to directly produce the final ranked list. This task highly relies on the intrinsic reasoning ability of LLM. Besides, FaiR-LLM [Zhang *et al.*, 2023a] and UP5 [Hua *et al.*, 2023a] intend to address the fairness issue when adapting LLM for item generation tasks.

Hybrid Task

In hybrid tasks, the large language model serves in a multi-task manner, where both the item scoring and generation tasks could be handled by a single LLM through a unified language interface. The basis for supporting this hybrid functionality is that large language models are inherent multi-task learners [Brown *et al.*, 2020; Ouyang *et al.*, 2022]. P5 [Geng *et al.*, 2022], M6-Rec [Cui *et al.*, 2022] and InstructRec [Zhang *et al.*, 2023b] tune the encoder-decoder models for better alignment towards a series of recommendation tasks

[2]Different domains means data sources with different distributions, *e.g.*, scenarios, datasets, platforms, etc.

including both item scoring and generation tasks via different prompting templates. Other works [Liu *et al.*, 2023a; Sun *et al.*, 2023; Dai *et al.*, 2023] manually design task-specific prompts to call a unified central LLM (*e.g.*, ChatGPT API) to perform multiple tasks, including but not restricted to pointwise rating prediction, pairwise item comparison, and listwise ranking list generation.

2.4 LLM for Pipeline Controller

As the model size scales up, LLM tends to exhibit emergent behaviors that may not be observed in previous smaller language models, *e.g.*, in-context learning and logical reasoning [Wei *et al.*, 2022; Zhao *et al.*, 2023]. With such emergent abilities, LLM is no longer just a part of the recommender system mentioned above, but could actively participate in the pipeline control over the system, possibly leading to a more interactive and explainable recommendation process. Chat-REC [Gao *et al.*, 2023] leverages ChatGPT to bridge the conversational interface and traditional recommender systems, where it is required to infer user preferences, decide whether or not to call the backend recommendation API, and further modify (*e.g.*, filter, rerank) the returned item candidates before presenting them to the user. RecLLM [Friedman *et al.*, 2023] further extends the permission of LLM, and proposes a roadmap for building an integrated conversational recommender system, where LLM is able to manage the dialogue, understand user preference, arrange the ranking stage, and even provide a controllable LLM-based user simulator to generate synthetic conversations.

2.5 Discussion

We could observe that the development trajectory about where to adapt LLM to RS is fundamentally aligned with the progress of large language models. Back to year 2021 and early days in 2022, the parameter sizes of pretrained language models are still relatively small (*e.g.*, 340M for BERT, 1.5B for GPT2-XL). Therefore, earlier works usually tend to either incorporate these small-scale language models as simple textual feature encoders, or as scoring/ranking functions finetuned to fit the data distribution from recommender systems.

As the model size gradually increases, researchers discover that large language models have gained emergent abilities (*e.g.*, instruction following, reasoning), as well as a vast amount of open-world knowledge with powerful text generation capacities. Equipped with these amazing features brought by large-scale parameters, LLM starts to not only deepen the usage in the feature encoder and scoring/ranking function stage, but also move beyond and extend their roles into other stages of the recommendation pipeline. For instance, in the feature engineering stage, we could instruct LLM to generate reliable auxiliary features and synthetic training data [Liu *et al.*, 2023c]. In this way, open-world knowledge from LLM is injected into the recommendation model, which is usually a closed-domain system. Not to mention, participating in the pipeline control further requires sufficient logical reasoning and tool utilization capabilities, which are possessed by LLM.

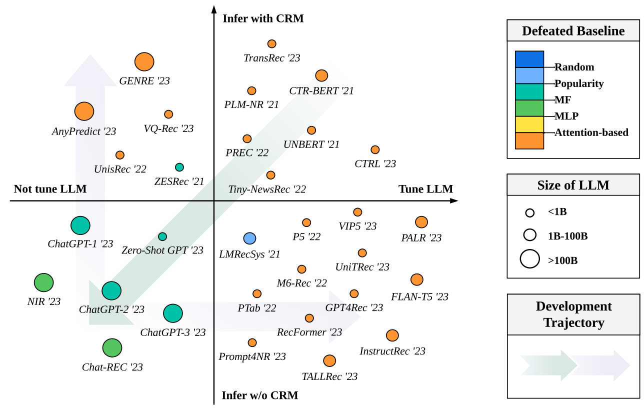

Figure 3: Four-quadrant classification about how to adapt LLM to RS. Each circle in the quadrants denotes one research work with the corresponding model name attached below the circle. The size of each circle means the largest size of LLM leveraged in the research work. The color of each circle indicates the best compared baseline that the proposed model defeats as reported in the corresponding paper. For example, the **green** circle of **Chat-REC** in quadrant 3 denotes that it utilizes a large language model with size larger than 100B (*i.e.*, ChatGPT) and defeats the MF baseline. Besides, we summarize the general development trajectory with light-colored arrows. Abbreviations: MF is short for matrix factorization; MLP is short for multi-layer perceptron.

In summary, we believe that, as the abilities of large language models are further explored, they will form gradually deeper couplings and bindings with multiple stages of the recommendation pipeline. Even further, we might need to customize large language models specifically tailored to satisfy the unique requirements of recommender systems [Lin and Zhang, 2023].

3 How to Adapt LLM

To answer the **"HOW"** question about adapting LLM to RS, we carry out two orthogonal taxonomy criteria to distinguish the adaptation of LLMs to RS, resulting in a four-quadrant classification shown in Figure 3:

- **Tune/Not Tune LLM** denotes whether we will tune LLM during the training phase. The definition of tuning LLM includes both full finetuning and other parameter-efficient finetuning methods (*e.g.*, LoRA [Hu *et al.*, 2021]).
- **Infer with/without CRM** denotes whether we will involve conventional recommendation models (CRM) during the inference phase. Note that there are works that only use CRM to serve as independent pre-ranking functions to generate candidate item set for LLM. We categorize them as "infer without CRM", since the CRM is independent of LLM, and could be decoupled from the final recommendation task.

In Figure 3, we use different marker sizes to indicate the size of the large language model the research works adapt, and use different colors to indicate the best baseline they have defeated in terms of item recommendation. Thus, a few works are not included since they do not provide traditional recommendation evaluation, *e.g.*, RecLLM [Friedman *et al.*, 2023] only investigates the system architecture design to involve LLM for RS pipeline control without experimental evaluation.

Given the four-quadrant taxonomy, the overall development trajectory generally follows the light-colored arrows in Figure 3. Accordingly, we will introduce the latest research works in the order of quadrant 1, 3, 2, 4.

3.1 Tune LLM; Infer with CRM (Quadrant 1)

Existing works in quadrant 1 mainly focus on applying relatively smaller pretrained language models (*e.g.*, BERT) to the field of news recommendation [Zhang *et al.*, 2021a; Wu *et al.*, 2021; Liu *et al.*, 2022b; Yu *et al.*, 2022b] and e-commercial advertisement [Muhamed *et al.*, 2021; Li *et al.*, 2023d]. As discussed in Section 2.5, the primary roles of these small-scale language models are only to serve as feature encoders for semantic representation enhancement. Consequently, a conventional recommendation model (CRM) is required to make the final recommendation, with generated textual embeddings as auxiliary inputs. Additionally, the small model size makes it affordable to fully finetune the language model during the training phase. TransRec [Fu *et al.*, 2023a] proposes layerwise adaptor tuning over BERT and ViT models to ensure the training efficiency and multi-modality enhanced representations. As shown in Figure 3, since CRM is involved and LLM is tunable, the research works in quadrant 1 could better align to the data distribution of recommender systems and thus all achieve satisfying performance. However, they only leverage small-scale language models as feature encoders, and thus the key capacities (*e.g.*, reasoning, instruction following) of large foundation models still remain underexplored in this quadrant.

3.2 Not Tune LLM; Infer w/o CRM (Quadrant 3)

With the emergence of large foundation models, especially ChatGPT, researchers intend to analyze the zero-shot or few-shot performance of LLM in recommendation domains, where LLM is frozen and CRM is not involved. Sileo *et al.* [2022] apply zero-shot learning on GPT-2 by inferring the next item according to the user's behavior history, which merely defeats the random baseline. Other works [Wang and Lim, 2023; Liu *et al.*, 2023a; Sun *et al.*, 2023; Dai *et al.*, 2023] investigate zero-shot and few-shot recommendation setting based on the ChatGPT API, with delicate prompt engineering to instruct the LLM to perform tasks like rating prediction, pairwise comparison, and listwise ranking. Chat-REC [Gao *et al.*, 2023] instructs ChatGPT to not only serve as the score/ranking function, but also take control over the recommendation pipeline, *e.g.*, deciding when to call an independent pre-ranking model API. As illustrated in Figure 3, although a larger model size might bring performance improvement, the zero-shot or few-shot learning of LLM is still much inferior compared with the light-weight CRM tuned on the training data, indicating the importance of in-domain collaborative knowledge from recommender systems.

3.3 Not Tune LLM; Infer with CRM (Quadrant 2)

Research works in quadrant 2 utilize different key capabilities (*e.g.*, rich semantic information, reasoning ability) of LLM without tuning to assist CRM in better completing recommendation tasks.

Early works [Ding *et al.*, 2021; Hou *et al.*, 2022; Hou *et al.*, 2023a] propose to extract transferable text embeddings from a fixed BERT model with rich semantic information. The text embeddings are then fed into several projection layers to better produce the cross-domain representations for trainable conventional recommendation models. The projection layers are designed as a single-layer neural network for ZESRec [Ding *et al.*, 2021], an MoE-enhanced network for UniSRec [Hou *et al.*, 2022], and a vector quantization based embedding lookup table for VQ-Rec [Hou *et al.*, 2023a]. We can observe from Figure 3 that the direct usage of a single-layer neural network as an adapter does not yield satisfactory results. However, with a carefully designed adapter module, the semantic representations from the fixed BERT parameters can be better aligned with the subsequent recommendation module, leading to impressive recommendation performances.

As discussed in Section 2.5, the emergent abilities and abundant open-world knowledge enable large foundation models to extend their roles to the feature engineering stage. AnyPredict [Wang *et al.*, 2023] leverages ChatGPT APIs to consolidate tabular samples to overcome the barrier across tables with varying schema, resulting in unified expanded training data for the follow-up conventional predictive models. GENRE [Liu *et al.*, 2023c] utilizes ChatGPT to perform news piece generation, user profiling, and news summarization, and thus augments the news recommendation model with LLM-generated features.

In these works, although LLM is frozen, the involvement of CRM for the inference phase generally guarantees better recommendation performance, comparing with works from quadrant 3 in Section 3.2 in terms of the best baseline they defeat.

3.4 Tune LLM; Infer w/o CRM (Quadrant 4)

Research works in quadrant 4 aim to finetune the large language models to serve as the scoring/ranking function based on the training data from recommender systems, excluding the involvement of CRM.

As an early attempt, LMRecSys [Zhang *et al.*, 2021b] tunes language models to estimate the score of each candidate item, resulting in unsatisfying performance. The reason might be that its scoring manners are somehow problematic, which may result from the limitations of the designed scoring method. Prompt4NR [Zhang and Wang, 2023] finetunes BERT by predicting the key answer words based on the prompting templates. PTab [Liu *et al.*, 2022a] transforms tabular data into text and finetunes a BERT model based on a masked language modeling task and classification tasks. UniTRec [Mao *et al.*, 2023] finetunes a BART model with a joint contrastive loss to optimize the discriminative score and a perplexity-based score. RecFormer [Li *et al.*, 2023b] adopts two-stage finetuning based on masked language modeling loss and item-item contrastive loss with LongFormer as the backbone model. P5 [Geng *et al.*, 2022], FLAN-T5 [Kang *et al.*, 2023], and InstructRec [Zhang *et al.*, 2023b] adopt T5 [Raffel *et al.*, 2020] as the backbone, and train the model in a sequence-to-sequence manner. GPT4Rec [Li *et* *al.*, 2023c] tunes GPT models as a generative function for next item prediction via causal language modeling.

The works mentioned above all adopt full finetuning, which could be considerably expensive and unscalable as the size of the language model continuously increases. To this end, PALR [Chen, 2023] fully finetunes LLaMA [Touvron *et al.*, 2023] based on only 20% of the user data, which not only achieves overall training efficiency but also demonstrates strong inductive learning capabilities of LLM. Besides, parameter-efficient finetuning methods are usually required to efficiently adapt LLM to RS, *e.g.*, option tuning for M6-Rec [Cui *et al.*, 2022], layerwise adaptor tuning for VIP5 [Geng *et al.*, 2023], and low-rank adaption (LoRA) [Hu *et al.*, 2021] for TALLRec [Bao *et al.*, 2023].

As shown in Figure 3, the performance of finetuning LLM based on recommendation data is promising with proper task formulation, even if the model size is still relatively small.

3.5 Discussion

We first conclude the necessity of collaborative knowledge injection when adapting LLM to RS, and then cast discussion on the relationship between the recommendation performance and the size of adapted LLM. Finally, we discuss an interesting property found in ChatGPT-like large language models.

Collaborative Knowledge is Needed

From Figure 3, we could observe a clear performance boundary between works from quadrant 3 and quadrant 1, 2, 4. Research works from quadrant 3 are inferior even though they adapt large-scale models, *i.e.*, ChatGPT. This indicates that the recommender system is a highly specialized area, which demands a lot of in-domain collaborative knowledge. LLM cannot learn such knowledge from its general pretraining corpus. Therefore, we have to involve in-domain collaborative knowledge for better performance when adapting LLM to RS, and there are generally two ways to achieve the goal (corresponding to quadrant 1, 2, 4):

- **Tune LLM** during the training phase, which injects collaborative knowledge from a data-centric aspect.
- **Infer with CRM** during the inference phase, which injects collaborative knowledge from a model-centric aspect.

As shown in Figure 3, we could observe a clear trajectory evolving from quadrant 3 to quadrant 2 and 4 through in-domain collaborative knowledge injection. Therefore, it is natural to draw the future prospect to further fill in the blank in quadrant 1, where we tune large foundation models for alignments and also involve CRM for inference.

Is Bigger Always Better?

By injecting in-domain collaborative knowledge from either data-centric or model-centric aspects, research works from quadrant 1, 2, 4 can achieve satisfying recommendation performance compared with attention-based baselines, except for a few cases. Among these studies, although we could observe that the size of adapted LLM gradually increases according to the timeline, a fine-grained cross comparison among them (*i.e.*, a unified benchmark) remains vacant. Hence, it is difficult to directly conclude that larger model size of LLM can definitely yield better results for recommender systems. We prefer to leave this as a open question for future works: *Is bigger language models always better for recommender systems? Or is it good enough to use small-scale language models in combination with collaborative knowledge injection?*

LLM is Good at Reranking Hard Samples

Although works in quadrant 3 suffer from inferior performance for zero/few-shot learning since little in-domain collaborative knowledge is involved, researchers [Ma *et al.*, 2023; Hou *et al.*, 2023b] have found that large language models such as ChatGPT are more likely to be a good reranker for *hard samples*. They introduce the filter-then-rerank paradigm which leverages a pre-ranking function from traditional recommender systems (*e.g.*, matching or pre-ranking stage in industrial applications) to pre-filter those easy negative items, and thus generate a set of candidates with harder samples for LLM to rerank. In this way, the listwise reranking performance of LLM (especially ChatGPT-like APIs) could be promoted. This finding is instructive for industrial applications, where we could require LLM to only handle hard samples and leave other samples for light-weight models for saving computational costs.

4 Challenges from Industrial Applications

Since the research of recommender systems is highly application-oriented, in this section, we highlight the key challenges in adapting LLM to RS, which mainly arise from the unique characteristics of recommender systems and industrial applications. Accordingly, we will also discuss the preliminary efforts done by existing works, as well as other possible solutions. The following challenges are proposed from three aspects: (1) **efficiency** (training efficiency, inference latency), (2) **effectiveness** (in-domain long text modeling, ID indexing & modeling), and (3) **ethics** (fairness).

4.1 Training Efficiency

There are two key aspects to keep good performance of modern deep learning based recommender systems: (1) enlarge the volumes of training data (*e.g.*, billion-level training samples), and (2) increase the model update frequency (from day-level to hour-level, or even minute-level). Both of them highly require the training efficiency. Although, as suggested in Section 3.5, tuning LLM (possibly with CRM) is a promising approach to align LLM to RS for better performance, it actually brings prohibitive adaptation costs in terms of both memory usage and time consumption. Therefore, how to ensure the efficiency when we involve LLM in the training phase is a key challenge for industrial applications.

Existing works mainly propose to leverage parameter-efficient finetuning strategies (*e.g.*, option tuning [Cui *et al.*, 2022] and layerwise adaptor tuning [Geng *et al.*, 2023]), which mainly solve the memory usage problem, but the time consumption is still high.

From an industrial perspective, we suggest adopting the long-short update strategy, when we leverage LLM for feature engineering and feature encoder. To be specific, we can cut down the training data volume and relax the update frequency for LLM (*e.g.*week-level) while maintaining full training data and high update frequency for CRM. The basis to support this approach is that researchers [Chen, 2023; Zhou *et al.*, 2023] point out that LLM has strong inductive learning capacities to produce generalized and reliable outputs via a handful of supervisions. In this way, LLM can provide aligned in-domain knowledge to CRM, while CRM act as a frequently updated adapter for LLM.

4.2 Inference Latency

Online recommender systems are usually real-time services and extremely time-sensitive, where all stages (*e.g.*, matching, ranking, reranking) should be done within around tens of milliseconds. The involvement of LLM during the inference phase gives rise to the inference latency problem. The inference time of the LLM is expensive, not to mention the additional time cost brought by prompt template generation.

Pre-computing and caching the outputs or middle representations of LLM is the common strategy to ensure low-latency inference when involving LLM during the inference phase. When adapting the LLM as the scoring/ranking functions, M6-Rec [Cui *et al.*, 2022] proposes the multi-segment late interaction strategy. The textual features of user and item are split into finer-grained segments that are more static, e.g., by representing each clicked item as an individual segment. Then, we can pre-compute and cache the encoded representations of each segment using the first several transformer layers, while the rest of the layers are leveraged to perform late interaction between segments when the recommendation request arrives. Other works like UniSRec [Hou *et al.*, 2022] and VQ-Rec [Hou *et al.*, 2023a] simply adopt language models as feature encoders. Hence it is straightforward to directly cache the dense embeddings produced by the language model.

Moreover, we could seek ways to *reduce the size of model* for final inference, where methods have been well explored in other deep learning domains, *e.g.*, distillation [Jiao *et al.*, 2019], pruning [Chen *et al.*, 2020], and quantization [Zafrir *et al.*, 2019]. For instance, CTRL [Li *et al.*, 2023d] propose to perform contrastive learning to distill the semantic knowledge from LLM to CRM which is then finetuned for the inference phase. These strategies generally serve as a tradeoff between the model performance and inference latency. Alternatively, we could involve LLM in the feature engineering stage, which does not bring extra burden of computation to the inference phase.

4.3 In-Domain Long Text Modeling

When adapting LLM, we have to construct in-domain textual inputs via prompting templates and insert proper instructions and demonstrations at the front if needed. However, the general guideline of industrial recommender systems requires longer user history, larger candidate set and more features to achieve better recommendation performance, possibly leading to long-text inputs for LLM. Such long-text inputs from RS domains (*i.e.*, in-domain long texts) could result in two key challenges as follows.

First, Hou *et al.* [2023b] discover that LLM has difficulty in dealing with long texts especially when we extend the text with longer user history or larger candidate set, even though the total number of input tokens does not exceed the length of the context window (*e.g.*, 512 for BERT, 4096 for ChatGPT). The reason might be that the distribution of in-domain long text is quite different from the pretraining corpora of LLM. Furthermore, an excessively long-text input will cause the memory inefficiency problem, and might even break the context window limitation, leading to partial information lost and inferior outputs from LLM.

To this end, it is of great importance to investigate how to properly filter, select, and arrange the textual information as the input for LLM during prompting engineering, as well as how to instruct or tune the LLM to better align with the distribution of these in-domain long-text inputs. Besides, in NLP domains, a range of works are proposed to address the context window limitation (*e.g.*, sliding windows [Wang *et al.*, 2019], memory mechanism [Ding *et al.*, 2020]), which could be considered in recommender systems.

4.4 ID Indexing & Modeling

In recommender systems, there exists a kind of pure ID features that inherently contains no semantic information (*e.g.*, user ID, item ID). If we include these ID features in the prompting text, the tokenization is actually unmeaningful to language models (*e.g.*, user ID AX1265 might be tokenized as [AX, 12, 65]). Many works [Cui *et al.*, 2022; Hou *et al.*, 2023a] tend to directly abandon these ID features (*e.g.*, replacing item IDs with item titles or descriptions) for unified cross-domain recommendation via the natural language interface, since the IDs are usually not shared in different domains. However, some works [Geng *et al.*, 2022; Yuan *et al.*, 2023] point out that bringing ID features can greatly promote the recommendation performance, although sacrificing the cross-domain generalization ability. Therefore, it is still an open question about whether we should retain the ID features or not, which divides the key challenges regarding ID indexing & modeling into two directions.

On the one hand, we could sacrifice the cross-domain generalization ability to obtain better in-domain recommendation performance by keeping the ID features. P5 [Geng *et al.*, 2022] and its variants [Geng *et al.*, 2023; Hua *et al.*, 2023a; Hua *et al.*, 2023b] remain the ID features as textual inputs in the prompting templates. P5 designs a whole-word embedding layer to assign the same whole-word embedding for tokens from the same ID feature. The whole-word embeddings will be added to the token embeddings in the same way as position embeddings in language models. Based on P5, Hua *et al.* [2023b] further explore various item ID indexing strategies (*e.g.*, sequential indexing, collaborative indexing) to ensure the IDs of similar items consist of similar sub-tokens. RecFormer [Li *et al.*, 2023b] and UniSRec [Hou *et al.*, 2022] omit the item IDs in prompting texts, but introduce additional ID embeddings at either bottom embedding layer or top projection layer. In this line, researchers should focus on how to associate LLM with ID features via carefully designed ID indexing & modeling strategies.

On the other hand, we could abandon the ID features to achieve unified cross-domain recommendation via natural language interface. Maintaining a unified model to serve various domains is very promising, especially when we involve large language model [Cui *et al.*, 2022; Hou *et al.*, 2023a]. In this direction, in order to achieve similar performance to those works that keep ID features, researchers could investigate ways to introduce ID features in an implicit manner, *e.g.*, contrastive learning between representations of LLMs and corresponding ID embeddings.

4.5 Fairness

Researchers have discovered that bias in the pretraining corpus could mislead LLM to generate harmful or offensive content, *e.g.*, discriminating against disadvantaged groups. Although there are strategies (*e.g.*, RLHF [Ouyang *et al.*, 2022]) to reduce the harmfulness of LLM, existing works have already detected the unfairness problem in recommender systems brought by LLM from both user-side [Hua *et al.*, 2023a; Zhang *et al.*, 2023a] and item-side [Hou *et al.*, 2023b] perspectives.

The user-side fairness in recommender systems requires similar users to be treated similarly at either individual level or group level. The user sensitive attributes should not be preset during recommendation (*e.g.*, gender, race). To this end, UP5 [Hua *et al.*, 2023a] proposes counterfactually fair prompting (CFP), which consists of a personalized prefix prompt and a prompt mixture to ensure fairness w.r.t. a set of sensitive attributes. Besides, Zhang *et al.* [2023a] introduce a benchmark named FaiRLLM, where authors comprise carefully crafted metrics and a dataset that accounts for eight sensitive attributes in recommendation scenarios where LLM is involved. Yet these studies only focus on the fairness issue in specific recommendation tasks (*e.g.*, item generation task) with limited evaluation metrics.

The item-side fairness in recommender systems ensures that each item or item group receives a fair chance to be recommended (*e.g.*, proportional to its merits or utility) [Patro *et al.*, 2020; Liu *et al.*, 2019; Singh and Joachims, 2018]. However, how to improve item-side fairness in LLM remains less explored. As a preliminary study, Hou *et al.* [2023b] observe that popularity bias exists when LLM serves as a ranking function, and alleviate the bias to some extents by designing prompts to guide the LLM focusing on users' historical interactions. Further studies on popularity bias and other potential item-wise fairness issues are still needed.

5 Conclusion and Future Prospects

This survey comprehensively summarizes the recent progress in adapting large language models to recommender systems from two perspectives: where and how to adapt LLM to RS.

- For the **"WHERE"** question, we analyze the roles that LLM could play at different stages of the recommendation pipeline, *i.e.*, feature engineering, feature encoder, scoring/ranking function, and pipeline controller.
- For the **"HOW"** question, we analyze the training and inference strategies, resulting in two orthogonal classification criteria, *i.e.*, whether to tune LLM, and whether to involve CRM for inference.

Detailed discussions and insightful development trajectories are also provided for each taxonomy perspective. As for future prospects, apart from the three aspects we have already highlighted in Section 4 (*i.e.*, **efficiency**, **effectiveness** and **ethics**), we would like to further express our hopeful vision for the future development of combining large language models and recommender systems:

- **A unified public benchmark** is of an urgent need to provide reasonable and convincing evaluation protocols, since (1) the fine-grained cross comparison among existing works remains vacant, and (2) it is quite expensive and difficult to reproduce the experimental results of recommendation models combined with LLM.
- **A customized large foundation model** for recommendation domains, which can take over control of the entire recommendation pipeline, enabling a new level of automation in recommender systems.

References

[Bao *et al.*, 2023] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *arXiv preprint arXiv:2305.00447*, 2023.

[Borisov *et al.*, 2023] Vadim Borisov, Kathrin Sessler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[Carranza *et al.*, 2023] Aldo Gael Carranza, Rezsa Farahani, Natalia Ponomareva, Alex Kurakin, Matthew Jagielski, and Milad Nasr. Privacy-preserving recommender systems with synthetic query generation using differentially private large language models. *arXiv preprint arXiv:2305.05973*, 2023.

[Chen *et al.*, 2020] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pretrained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

[Chen *et al.*, 2023] Jiao Chen, Luyi Ma, Xiaohan Li, Nikhil Thakurdesai, Jianpeng Xu, Jason HD Cho, Kaushiki Nag, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Knowledge graph completion models are few-shot learners: An empirical study of relation labeling in e-commerce with llms. *arXiv preprint arXiv:2305.09858*, 2023.

[Chen, 2023] Zheng Chen. Palr: Personalization aware llms for recommendation. *arXiv preprint arXiv:2305.07622*, 2023.

[Christakopoulou *et al.*, 2023] Konstantina Christakopoulou, Alberto Lalama, Cj Adams, Iris Qu, Yifat

Amir, Samer Chucri, Pierce Vollucci, Fabio Soldo, Dina Bseiso, Sarah Scodel, et al. Large language models for user interest journeys. *arXiv preprint arXiv:2305.15498*, 2023.

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

[Dai *et al.*, 2021] Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, and Yong Yu. An adversarial imitation click model for information retrieval. In *Proceedings of the Web Conference 2021*, pages 1809–1820, 2021.

[Dai *et al.*, 2023] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. Uncovering chatgpt's capabilities in recommender systems. *arXiv preprint arXiv:2305.02182*, 2023.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Ding *et al.*, 2020] Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems*, 33:12792–12804, 2020.

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

[Friedman *et al.*, 2023] Luke Friedman, Sameer Ahuja, David Allen, Terry Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. Leveraging large language models in conversational recommender systems. *arXiv preprint arXiv:2305.07961*, 2023.

[Fu *et al.*, 2023a] Junchen Fu, Fajie Yuan, Yu Song, Zheng Yuan, Mingyue Cheng, Shenghui Cheng, Jiaqi Zhang, Jie Wang, and Yunzhu Pan. Exploring adapter-based transfer learning for recommender systems: Empirical studies and practical insights. *arXiv preprint arXiv:2305.15036*, 2023.

[Fu *et al.*, 2023b] Lingyue Fu, Jianghao Lin, Weiwen Liu, Ruiming Tang, Weinan Zhang, Rui Zhang, and Yong Yu. An f-shape click model for information retrieval on multiblock mobile pages. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 1057–1065, 2023.

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

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

[Geng *et al.*, 2023] Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. Vip5: Towards multimodal foundation models for recommendation. *arXiv preprint arXiv:2305.14302*, 2023.

[Guo *et al.*, 2017] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

[He *et al.*, 2022] Junda He, Bowen Xu, Zhou Yang, DongGyun Han, Chengran Yang, and David Lo. Ptm4tag: sharpening tag recommendation of stack overflow posts with pre-trained models. In *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*, pages 1–11, 2022.

[Hegselmann *et al.*, 2023] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR, 2023.

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

[Hou *et al.*, 2023a] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*, pages 1162–1171, 2023.

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

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

[Hua *et al.*, 2023a] Wenyue Hua, Yingqiang Ge, Shuyuan Xu, Jianchao Ji, and Yongfeng Zhang. Up5: Unbiased foundation model for fairness-aware recommendation. *arXiv preprint arXiv:2305.12090*, 2023.

[Hua *et al.*, 2023b] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. *arXiv preprint arXiv:2305.06569*, 2023.

[Huang and Chang, 2022] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[Jiao *et al.*, 2019] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu.

Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[Kang *et al.*, 2023] Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*, 2023.

[Li *et al.*, 2023a] Chen Li, Yixiao Ge, Jiayong Mao, Dian Li, and Ying Shan. Taggpt: Large language models are zero-shot multimodal taggers. *arXiv preprint arXiv:2304.03022*, 2023.

[Li *et al.*, 2023b] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. Text is all you need: Learning language representations for sequential recommendation. *arXiv preprint arXiv:2305.13731*, 2023.

[Li *et al.*, 2023c] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. Gpt4rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879*, 2023.

[Li *et al.*, 2023d] Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. Ctrl: Connect tabular and language model for ctr prediction. *arXiv preprint arXiv:2306.02841*, 2023.

[Lin and Zhang, 2023] Guo Lin and Yongfeng Zhang. Sparks of artificial general recommender (agr): Early experiments with chatgpt. *arXiv preprint arXiv:2305.04518*, 2023.

[Lin *et al.*, 2021] Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Shuai Li, Ruiming Tang, Xiuqiang He, Jianye Hao, and Yong Yu. A graph-enhanced click model for web search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1259–1268, 2021.

[Liu *et al.*, 2019] Weiwen Liu, Jun Guo, Nasim Sonboli, Robin Burke, and Shengyu Zhang. Personalized fairness-aware re-ranking for microlending. In *Proceedings of the 13th ACM conference on recommender systems*, pages 467–471, 2019.

[Liu *et al.*, 2021] Yiding Liu, Weixue Lu, Suqi Cheng, Daiting Shi, Shuaiqiang Wang, Zhicong Cheng, and Dawei Yin. Pre-trained language model for web-scale retrieval in baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3365–3375, 2021.

[Liu *et al.*, 2022a] Guang Liu, Jie Yang, and Ledell Wu. Ptab: Using the pre-trained language model for modeling tabular data. *arXiv preprint arXiv:2209.08060*, 2022.

[Liu *et al.*, 2022b] Qijiong Liu, Jieming Zhu, Quanyu Dai, and Xiaoming Wu. Boosting deep ctr prediction with a plug-and-play pre-trainer for news recommendation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2823–2833, 2022.

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

[Liu *et al.*, 2023b] Peng Liu, Lemei Zhang, and Jon Atle Gulla. Pre-train, prompt and recommendation: A comprehensive survey of language modelling paradigm adaptations in recommender systems. *arXiv preprint arXiv:2302.03735*, 2023.

[Liu *et al.*, 2023c] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. A first look at llm-powered generative news recommendation. *arXiv preprint arXiv:2305.06566*, 2023.

[Ma *et al.*, 2023] Yubo Ma, Yixin Cao, YongChing Hong, and Aixin Sun. Large language model is not a good few-shot information extractor, but a good reranker for hard samples! *arXiv preprint arXiv:2303.08559*, 2023.

[Mao *et al.*, 2023] Zhiming Mao, Huimin Wang, Yiming Du, and Kam-fai Wong. Unitrec: A unified text-to-text transformer and joint contrastive learning framework for text-based recommendation. *arXiv preprint arXiv:2305.15756*, 2023.

[Muhamed *et al.*, 2021] Aashiq Muhamed, Iman Keivanloo, Sujan Perera, James Mracek, Yi Xu, Qingjun Cui, Santosh Rajagopalan, Belinda Zeng, and Trishul Chilimbi. Ctr-bert: Cost-effective knowledge distillation for billion-parameter teacher models. In *NeurIPS Efficient Natural Language and Speech Processing Workshop*, 2021.

[Ouyang *et al.*, 2022] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[Patro *et al.*, 2020] Gourab K Patro, Arpita Biswas, Niloy Ganguly, Krishna P Gummadi, and Abhijnan Chakraborty. Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms. In *Proceedings of the web conference 2020*, pages 1194–1204, 2020.

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

[Radford *et al.*, 2019] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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

[Rahmani *et al.*, 2023] Sajjad Rahmani, AmirHossein Naghshzan, and Latifa Guerrouj. Improving code example recommendations on informal documentation using bert and query-aware lsh: A comparative study. *arXiv preprint arXiv:2305.03017*, 2023.

[Sileo *et al.*, 2022] Damien Sileo, Wout Vossen, and Robbe Raymaekers. Zero-shot recommendation as language modeling. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*, pages 223–230. Springer, 2022.

[Singh and Joachims, 2018] Ashudeep Singh and Thorsten Joachims. Fairness of exposure in rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2219–2228, 2018.

[Sun *et al.*, 2019] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.

[Sun *et al.*, 2023] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as reranking agent. *arXiv preprint arXiv:2304.09542*, 2023.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[Wang and Lim, 2023] Lei Wang and Ee-Peng Lim. Zero-shot next-item recommendation using large pretrained language models. *arXiv preprint arXiv:2304.03153*, 2023.

[Wang *et al.*, 2019] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. *arXiv preprint arXiv:1908.08167*, 2019.

[Wang *et al.*, 2023] Zifeng Wang, Chufan Gao, Cao Xiao, and Jimeng Sun. Anypredict: Foundation model for tabular prediction. *arXiv preprint arXiv:2305.12081*, 2023.

[Wei *et al.*, 2022] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[Wu *et al.*, 2021] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1652–1656, 2021.

[Wu *et al.*, 2022] Chuhan Wu, Fangzhao Wu, Tao Qi, Chao Zhang, Yongfeng Huang, and Tong Xu. Mm-rec: Visiolinguistic model empowered multimodal news recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2560–2564, 2022.

[Xi *et al.*, 2023] Yunjia Xi, Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Rui Zhang, Ruiming Tang, and Yong Yu. A bird's-eye view of reranking: from list level to page level. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 1075–1083, 2023.

[Yu *et al.*, 2022a] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Jundong Li, and Zi Huang. Self-supervised learning for recommender systems: A survey. *arXiv preprint arXiv:2203.15876*, 2022.

[Yu *et al.*, 2022b] Yang Yu, Fangzhao Wu, Chuhan Wu, Jingwei Yi, and Qi Liu. Tiny-newsrec: Effective and efficient plm-based news recommendation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5478–5489, 2022.

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

[Zafrir *et al.*, 2019] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE, 2019.

[Zhang and Wang, 2023] Zizhuo Zhang and Bang Wang. Prompt learning for news recommendation. *arXiv preprint arXiv:2304.05263*, 2023.

[Zhang *et al.*, 2021a] Qi Zhang, Jingjie Li, Qinglin Jia, Chuyuan Wang, Jieming Zhu, Zhaowei Wang, and Xiuqiang He. Unbert: User-news matching bert for news recommendation. In *IJCAI*, pages 3356–3362, 2021.

[Zhang *et al.*, 2021b] Yuhui Zhang, Hao Ding, Zeren Shui, Yifei Ma, James Zou, Anoop Deoras, and Hao Wang. Language models as recommender systems: Evaluations and limitations. 2021.

[Zhang *et al.*, 2022] Xinyang Zhang, Yury Malkov, Omar Florez, Serim Park, Brian McWilliams, Jiawei Han, and Ahmed El-Kishky. Twhin-bert: A socially-enriched pretrained language model for multilingual tweet representations. *arXiv preprint arXiv:2209.07562*, 2022.

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

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

[Zhao *et al.*, 2023] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[Zhiyuli *et al.*, 2023] Aakas Zhiyuli, Yanfang Chen, Xuan Zhang, and Xun Liang. Bookgpt: A general framework for book recommendation empowered by large language model. *arXiv preprint arXiv:2305.15673*, 2023.

[Zhou *et al.*, 2023] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*, 2023.

[Zou *et al.*, 2021] Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. Pre-trained language model based ranking in baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4014–4022, 2021.

Table 1: An organization of works on adapting large language models (LLM) to recommender systems (RS). We use the following abbreviations. **FFT**: full finetuning. **PT**: prompt tuning. **LAT**: layerwise adapter tuning. **OT**: option tuning. **T-FEW**: few-shot parameter efficient tuning. Note that only the largest models used in the corresponding papers are listed. If the version of the pretrained language model is not specified, we assume it to be the base version.

Model Name	LLM Backbone	LLM Tuning Strategy	RS Task	RS Scenario
Feature Engineering				
GReaT [Borisov *et al.*, 2023]	GPT2-Medium (355M)	FFT	N/A	Tabular
GENRE [Liu *et al.*, 2023c]	ChatGPT	Frozen	Retrieval Sequential RS	News
AnyPredict [Wang *et al.*, 2023]	ChatGPT	Frozen	N/A	Tabular
LLM4KGC [Chen *et al.*, 2023]	PaLM (540B) ChatGPT	Frozen	N/A	E-commerce
TagGPT [Li *et al.*, 2023a]	ChatGPT	Frozen	Item Tagging	Food Video
ICPC [Christakopoulou *et al.*, 2023]	LaMDA (137B)	FFT/PT	User Profiling	N/A
DPLLM [Carranza *et al.*, 2023]	T5-XL (3B)	FFT	Retrieval Privacy	Web Search
Feature Encoder				
U-BERT [Qiu *et al.*, 2021]	BERT-base (110M)	FFT	Rating Prediction	Business E-commerce
UNBERT [Zhang *et al.*, 2021a]	BERT-base (110M)	FFT	Sequential RS	News
PLM-NR [Wu *et al.*, 2021]	RoBERTa-base (125M)	FFT	Sequential RS	News
Pyramid-ERNIE [Zou *et al.*, 2021]	ERNIE (110M)	FFT	Ranking	Web Search
ERNIE-RS [Liu *et al.*, 2021]	ERNIE (110M)	FFT	Retrieval	Web Search
CTR-BERT [Muhamed *et al.*, 2021]	Customized BERT (1.5B)	FFT	CTR Prediction	E-commerce
ZESRec [Ding *et al.*, 2021]	BERT-base (110M)	Frozen	Sequential RS	E-commerce
UniSRec [Hou *et al.*, 2022]	BERT-base (110M)	Frozen	Sequential RS	E-commerce
PREC [Liu *et al.*, 2022b]	BERT-base (110M)	FFT	CTR Prediction	News
MM-Rec [Wu *et al.*, 2022]	BERT-base (110M)	FFT	Sequential RS	News
Tiny-NewsRec [Yu *et al.*, 2022b]	UniLMv2-base (110M)	FFT	Sequential RS	News
PTM4Tag [He *et al.*, 2022]	CodeBERT (125M)	FFT	Top-N RS	posts
TwHIN-BERT [Zhang *et al.*, 2022]	BERT-base (110M)	FFT	Social RS	posts
VQ-Rec [Hou *et al.*, 2023a]	BERT-base (110M)	Frozen	Sequential RS	E-commerce
IDRec vs MoRec [Yuan *et al.*, 2023]	BERT-base (110M)	FFT	Sequential RS	News Video E-commerce
TransRec [Fu *et al.*, 2023a]	RoBERTa-base (125M)	LAT	Cross-domain RS Sequential RS	News Video E-commerce
LSH [Rahmani *et al.*, 2023]	BERT-base (110M)	FFT	Top-N RS	Code
Scoring/Ranking Function (Item Scoring Task)				
LMRecSys [Zhang *et al.*, 2021b]	GPT2-XL (1.5B)	FFT	Top-N RS	Movie
PTab [Liu *et al.*, 2022a]	BERT-base (110M)	FFT	N/A	Tabular
UniTRec [Mao *et al.*, 2023]	BART (406M)	FFT	Sequential RS	News Question Social Media
Prompt4NR [Zhang and Wang, 2023]	BERT-base/110M	FFT	Sequential RS	News

Table 1 continued from previous page

Model Name	LLM Backbone	LLM Tuning Strategy	RS Task	RS Scenario
RecFormer [Li *et al.*, 2023b]	LongFormer/149M	FFT	Sequential RS	Product
TabLLM [Hegselmann *et al.*, 2023]	T0 (11B)	T-FEW	N/A	Tabular
Zero-shot GPT [Sileo *et al.*, 2022]	GPT2-Medium (355M)	Frozen	Rating Prediction	Movie
FLAN-T5 [Kang *et al.*, 2023]	FLAN-T5-XXL (11B)	FFT	Rating Prediction	Book Movie
BookGPT [Zhiyuli *et al.*, 2023]	ChatGPT	Frozen	Sequential RS Top-N RS Summary Recommendation	Book
TALLRec [Bao *et al.*, 2023]	LLaMA (7B)	LoRA	Sequential RS	Book Movie
Scoring/Ranking Function (Item Generation Task)				
GPT4Rec [Li *et al.*, 2023c]	GPT2 (110M)	FFT	Sequential RS	E-commerce
UP5 [Hua *et al.*, 2023a]	T5-base (223M)	FFT	Retrieval Sequential RS	Movie Insurance
VIP5 [Geng *et al.*, 2023]	T5-base (223M)	LAT	Sequential RS Top-N RS Explaination Generation	E-commerce
P5-ID [Hua *et al.*, 2023b]	T5-small (61M)	FFT	Sequential RS	Business E-commerce
FaiRLLM [Zhang *et al.*, 2023a]	ChatGPT	Frozen	Top-N RS	Music Movie
PALR [Chen, 2023]	LLaMA (7B)	FFT	Sequential RS	Movie E-commerce
ChatGPT-3 [Hou *et al.*, 2023b]	ChatGPT	Frozen	Sequential RS	Movie E-commerce
AGR [Lin and Zhang, 2023]	ChatGPT	Frozen	Conversational RS	N/A
NIR [Wang and Lim, 2023]	GPT-3 (175B)	Frozen	Sequential RS	Movie
Scoring/Ranking Function (Hybrid Task)				
P5 [Geng *et al.*, 2022]	T5-base (223M)	FFT	Rating Prediction Top-N RS Sequential RS Explanation Generation Review Summarization	Business E-commerce
M6-Rec [Cui *et al.*, 2022]	M6-base (300M)	OT	Retrieval Ranking Explanation Generation Conversational RS	E-commerce
InstructRec [Zhang *et al.*, 2023b]	Flan-T5-XL (3B)	FFT	Sequential RS Product Search Personalized Search Matching-then-reranking	E-commerce
ChatGPT-1 [Liu *et al.*, 2023a]	ChatGPT	Frozen	Rating Prediction Top-N RS Sequential RS Explanation Generation Review Summarization	E-commerce
ChatGPT-2 [Dai *et al.*, 2023]	ChatGPT	Frozen	Pointwise Ranking Pairwise Ranking List-wise Ranking	News Movie E-commerce
ChatGPT-4 [Sun *et al.*, 2023]	ChatGPT	Frozen	Passage Reranking	Web Search

Table 1 continued from previous page

Model Name	LLM Backbone	LLM Tuning Strategy	RS Task	RS Scenario
Pipeline Controller				
Chat-REC [Gao *et al.*, 2023]	ChatGPT	Frozen	Rating Prediction Top-N RS	Movie
RecLLM [Friedman *et al.*, 2023]	LLaMA (7B)	FFT	Conversational RS	Video