Text-like Encoding of Collaborative Information in Large Language Models for Recommendation

Anonymous ACL submission

Abstract

When adapting Large Language Models for Recommendation (LLMRec), it is crucial to integrate collaborative information. Existing methods achieve this by learning collaborative embeddings in LLMs' latent space from scratch or by mapping from external models. However, they fail to represent the information in a text-like format, which may not align optimally with LLMs. To bridge this gap, we introduce BinLLM, a novel LLMRec method that seamlessly integrates collaborative information through text-like encoding. BinLLM converts collaborative embeddings from external models into binary sequences — a specific text format that LLMs can understand and operate on directly, facilitating the direct usage of collaborative information in text-like format by LLMs. Additionally, BinLLM provides options to compress the binary sequence using dot-decimal notation to avoid excessively long lengths. Extensive experiments validate that BinLLM introduces collaborative information in a manner better aligned with LLMs, resulting in enhanced performance.

1 Introduction

Due to the remarkable power of large language models (LLMs), there is a growing focus on adapting them for recommender systems (LLMRec), which has seen significant progress in the past year (Bao et al., 2023b,a,c; Harte et al., 2023; Rajput et al., 2023; Wei et al., 2024). In recommendation, collaborative information, which delineates the co-occurrence patterns among user-item interactions, has emerged as a pivotal component in modeling user interests, especially for active users and items (Zhang et al., 2023b). However, this information exists in a different modality from textual data and thus presents a challenge in directly leveraged by LLMs like textual information (Zhang et al., 2023b; Li et al., 2023b). To enhance recommendation quality, it is undoubtedly crucial to seamlessly integrate collaborative information into LLMs.

To date, two integration strategies have emerged. The first strategy resembles latent factor models (Koren et al., 2009) by incorporating additional tokens and corresponding embeddings into LLMs to represent users and items, subsequently fitting interaction data to implicitly capture collaborative information within the embeddings (Zheng et al., 2023; Hua et al., 2023). However, this approach suffers from low learning efficacy due to the inherent low-rank nature of the information, leading to tokenization redundancy within LLMs (Delétang et al., 2023; Zhang et al., 2023b). To address these challenges, an alternative approach leverages an external latent factor model to capture the information, which is then mapped into the LLM token embedding space (Zhang et al., 2023b; Li et al., 2023c; Liao et al., 2023), circumventing the need to learn it from scratch. While effective, this method introduces the additional overhead of training the mapping model.

Whether learning collaborative information directly from scratch in the LLM token embedding space or mapping it from external models, the resulting representations diverge significantly from the LLM's original textual-level encoding. This, to a certain extent, hampers the full utilization of LLMs' capabilities, as LLMs are initially trained on textual data and excel at processing textually encoded information. For instance, introducing new tokens alters the generative space of LLMs, potentially compromising their original functionalities, let alone capitalizing on their capabilities. Therefore, exploring text-like encoding of collaborative information in LLMs holds immense promise. Nevertheless, it poses challenges due to the inherent differences between textual and collaborative information modalities (Zhang et al., 2023b).

In this study, we delve into the central theme of encoding collaborative information in LLMs for

recommendation, an area of promise yet not explored in LLMRec. The crux lies in transforming collaborative information into a sequence formatted like text. We believe that this text-like sequence need not be comprehensible to humans; rather, it should be interpretable by LLMs for effective utilization, such as facilitating reasoning tasks like discerning user and item similarities through sequence comparisons. Thus, this text sequence does not necessarily have to adhere to conventional natural language patterns.

To this end, we introduce *BinLLM*, an innovative LLMRec approach that integrates collaborative information into LLMs using a text-like encoding strategy. We transform the collaborative embeddings obtained from external models into binary sequences, treating them as textual features directly usable by LLMs. This design is motivated by two primary considerations: 1) the feasibility of binarizing collaborative embeddings without compromising performance (Tan et al., 2020); 2) LLMs can naturally perform bitwise operations or do so after instruction tuning (Savelka et al., 2023), enabling the comparison of similarities between binarized sequences. Taking a step further, we explore representing the binary sequence in dot-decimal notation (Abusafat et al., 2021), resulting in shorter representations, akin to converting binary sequences to IPv4 addresses. By fine-tuning LLMs with recommendation instruction data containing such encoded collaborative information, we could leverage both textual semantics and collaborative data for recommendation without modifying the LLMs.

The main contributions of this work are summarized as follows:

- We emphasize the significance of text-like encoding for collaborative information in LLMRec to enhance alignment with LLMs.
- We introduce *BinLLM*, a novel method that efficiently encodes collaborative information textually for LLMs by converting collaborative embeddings into binary sequences.
- We perform comprehensive experiments on two datasets, showcasing the effectiveness of our approach through extensive results.

2 Methodology

In this section, we introduce our BinLLM method, starting with presenting the model architecture and followed by a description of the tuning method.

2.1 Model Architecture

Figure 1 depicts the model architecture of BinLLM, comprising two main components: prompt generation and LLM prediction. Similar to previous approaches, we convert recommendation data into prompts and then input them directly into LLMs for prediction. However, the key distinction of BinLLM is that it represents collaborative information in a text-like format by converting collaborative embeddings into binary sequences. We next delve into the specifics of these two components.

2.1.1 Prompt Construction

As depicted in Figure 1, we construct prompts using a template featuring empty fields, encompassing both textual fields (*e.g.,* "<ItemTitleList>") and ID fields (*e.g.,* "<UserID>"). By populating these fields with corresponding users' data, we can generate personalized prompts for recommendation purposes. The textual fields are utilized to incorporate textual information, which can be directly filled with corresponding textual data from the recommendation dataset, such as historical item titles in the "<ItemTitleList>" fields. The ID fields are designated for embedding collaborative information, which is acquired through a Text-like Encoding (TE) module. Next, we delve into the encoding process of collaborative information.

Text-like Encoding of Collaborative Information. To better integrate with LLMs, we aim to encode collaborative information in a text-like format. To accomplish this, we convert collaborative information into a binary sequence, enabling LLMs to perform bitwise operations for reasoning. The encoding model involves two components: 1) Collaborative Model, a conventional latent factor module capable of encoding collaborative information as numerical latent vectors (*i.e.,* collaborative embeddings). 2) Binarization & Compression Module, utilized to transform collaborative embeddings into binary sequences or further compressed formats.

• **Collaborative model.** Given a user u and an item i, the collaborative model generates corresponding embeddings for them, denoted as $\boldsymbol{e}_u$ and $\boldsymbol{e}_i$, respectively. Formally,

$$\boldsymbol{e}_u = f_c(u;\theta), \quad \boldsymbol{e}_i = f_c(i;\theta), \tag{1}$$

where f_c represents the collaborative model parameterized by θ. Here, $\boldsymbol{e}_u \in \mathcal{R}^d$ and $\boldsymbol{e}_i \in \mathcal{R}^d$ are d-dimensional embeddings that encode collaborative information for the user and item, respectively.

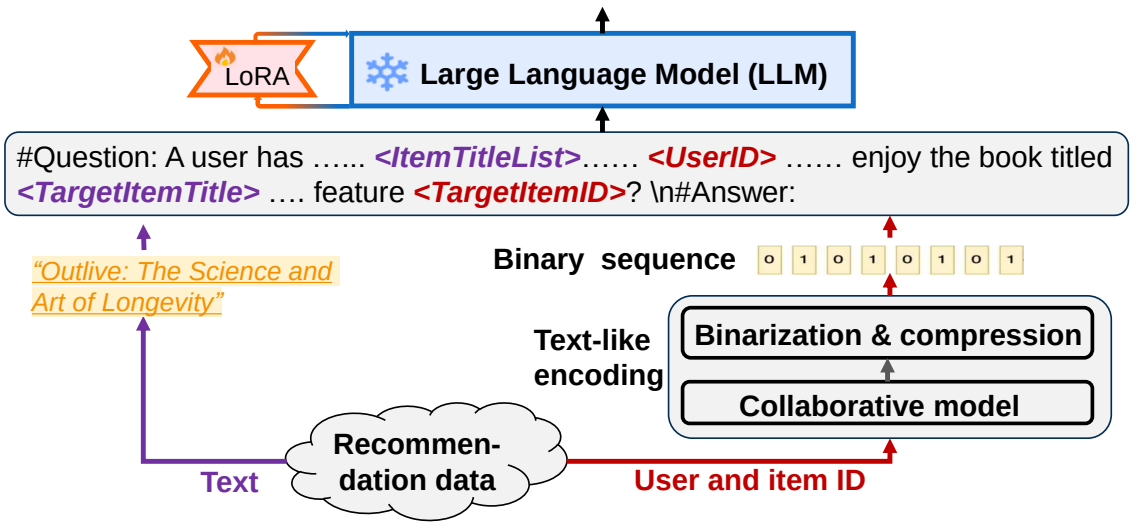


Figure 1: Model architecture overview of our BinLLM. The purple line is used to fill the text fields in the prompt template, introducing textual information like item titles, while the red line is used to fill the ID fields in the prompt template, introducing collaborative information.

• **Binarization & compression.** After obtaining the collaborative embeddings, this component is used to convert them into binary sequences, with the option to compress the sequences.

Binarization. To binarize the collaborative embeddings, we generally follow the mechanism proposed by Tan et al. (2020). Firstly, we transform the collaborative embeddings into a suitable space using a fully connected layer and then apply the sign function to obtain the binary results. Formally, for collaborative embeddings $\boldsymbol{e}_u$ and $\boldsymbol{e}_i$ of user u and item i, they are converted into binary sequences as follows:

$$\boldsymbol{h}_u = sign(\sigma(W\boldsymbol{e}_u+b)) \quad \boldsymbol{h}_i = sign(\sigma(W\boldsymbol{e}_i+b)), \tag{2}$$

where $\boldsymbol{h}_u \in \{0,1\}^d$ and $\boldsymbol{h}_i \in \{0,1\}^d$ denote the obtained binary representation of collaborative information for the user and item, respectively. Here, $W \in \mathcal{R}^{d\times d}$ and $b \in \mathcal{R}^d$ are the weights and bias for the fully connected layer, $\sigma(\cdot)$ represents the *tanh* activation function, and sign$(\cdot)$ denotes the sign function. For a numerical value x, we have:

$$sign(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{else} \end{cases}. \tag{3}$$

Through this method, we convert the numerical collaborative embeddings into binary sequences (e.g., '010110....'). These sequences can be directly inputted into LLMs and utilized for operations such as computing logical 'AND', thereby aiding in user preference reasoning.

Compression. A limitation of binary sequences is their relatively long length, which poses a challenge for LLMs not proficient in handling lengthy sequences. Moreover, long sequences can constrain the inference efficiency of LLMRec. We thus consider compressing the binary sequences while keeping them leverageable by LLMs. Given that IPv4 (Peterson and Davie, 2007) is originally encoded from binary sequences and the Web includes sufficient knowledge about IPv4, the LLMs trained on the Web data could potentially understand the dot-decimal notation used by IPv4. Therefore, we consider compressing the binary embeddings in dot-decimal notations (Abusafat et al., 2021). We convert every eight binary digits into a decimal number, ranging from 0 to 255, and use the full stop (dot) as a separation character. Here is an example of compressing a 32-bit binary sequence:

$$\underbrace{10101100}_{172.}\underbrace{00010000}_{16.}\underbrace{11111110}_{254.}\underbrace{00000001}_{1}. \tag{4}$$

Here, "172.16.254.1" is the compressed result, which significantly reduces the representation length. Notably, the compression is optional, and its usage depends on the length of the original binary sequence.

2.1.2 LLM Prediction

Once the empty fields in the prompt template are filled, the resulting prompt is fed into the LLMs for prediction. Similar to prior research, given the absence of specific recommendation pre-training

in LLMs, we introduce an additional LoRA module (Hu et al., 2022) for recommendation prediction. Formally, for a generated prompt p, the prediction can be formulated as:

$$\hat{y} = LLM_{\hat{\Phi}+\Phi'}(p), \tag{5}$$

where $\hat{\Phi}$ represents the pre-trained LLM's parameters, Φ' denotes the LoRA model parameters, and $\hat{y}$ represents the prediction results, which could be the predicted next item or the predicted likelihood of liking a candidate item, depending on the task.

2.2 Training

In our model architecture, two modules require training: the text-like encoding module and the LoRA module. The tuning for the text-like encoding module focuses on learning to generate the binary sequence for collaborative information, independent of the LLMs. The tuning for LoRA aims to instruct the LLM in making recommendations by leveraging collaborative information. We now present the two tuning paradigms, respectively.

2.2.1 Pre-training for Text-like Encoding

To train the text-like encoding module, we directly utilize the binarized representation from Equation (2) to fit the training data. Formally, let $\mathcal{D}$ denote the training data, and $(u, i, t) \in \mathcal{D}$ denote an interaction between user u and item i with label t. We train the module by minimizing the following optimization problem:

$$\underset{\theta,W,b}{minimize} \sum_{(u,i,t)\in\mathcal{D}} \ell(t, \boldsymbol{h}_u^\top \boldsymbol{h}_i), \tag{6}$$

where $\{\theta, W, b\}$ denote the model parameters in our text-like encoding module as discussed in Section 2.1.1, $\boldsymbol{h}_u$ and $\boldsymbol{h}_i$ denote the binary representations obtained from Equation (2), $\boldsymbol{h}_u^\top \boldsymbol{h}_i$ represents the predicted likelihood of user u liking item i, and $\ell(\cdot)$ denotes the common recommendation loss, in this work, the binary cross-entropy loss.

Notably, the sign function lacks smoothness, and its gradient is ill-defined as zero, posing an apparent challenge for back-propagation. To enable training the model in an end-to-end fashion, we approximate the gradient using the straight-through estimator (STE), following the approach outlined by Tan et al. (2020). That is, we directly use the gradients of the output as the gradients of the input for the sign function.

2.2.2 LoRA Tuning

To tune the LoRA module, we consider two tuning methods: intuitive tuning and two-step tuning.

Intuitive tuning: This method directly tunes the LoRA module from scratch with the prompts that contain the collaborative information.

Two-step tuning: In intuitive tuning, a potential challenge arises in scenarios like rating prediction tasks, where binary representations can serve as highly effective features with relatively low learning complexity[1]. Incorporating collaborative information from scratch might cause the model to overly depend on these features, potentially neglecting other attributes akin to learning shortcut features. To address this, we propose an additional two-step tuning strategy. Initially, we train the model using a prompt that excludes collaborative information. Subsequently, we refine the model further by fine-tuning it using the complete prompt that contains the collaborative information.

3 Experiments

In this section, we conduct experiments to answer the following research questions:
RQ1: Does BinLLM effectively incorporate collaborative information into LLMs to improve recommendation performance? How does its performance compare with that of existing methods?
RQ2: How do our design choices influence the performance of the proposed method BinLLM?

3.1 Experimental Settings

Recommendation Task. Given that this is an initial exploration of text-like encoding for collaborative information, our experiments primarily concentrate on the click/rating prediction task, with other recommendation tasks being ignored. Specifically, we aim to predict whether a user u (comprising other profile information such as historical interactions) would click on/like a given candidate item i. The task aligns with that of CoLLM, which investigates the utilization of collaborative information for recommendation through embedding mapping in latent space. Hence, our experimental setup generally follows that of CoLLM.

Datasets. We conduct experiments on two representative datasets:

[1]Because the model could achieve satisfactory results by solely performing bitwise "AND" operations on the collaborative representations of the given user and candidate item, referencing the learning process of binary representation.

Table 1: Statistics of the processed datasets.

Dataset	#Train	#Valid	#Test	#User	#Item
ML-1M	33,891	10,401	7,331	839	3,256
Amazon-Book	727,468	25,747	25,747	22,967	34,154

- **ML-1M** (Harper and Konstan, 2016): This refers to a widely recognized movie recommendation benchmark dataset, MovieLens-1M[2], provided by GroupLens research. The dataset comprises user ratings for movies and includes textual information for users and items, such as movie titles.
- **Amazon-Book** (Ni et al., 2019): This pertains to the "Books" subset within the renowned Amazon Product Review dataset[3]. This dataset aggregates user reviews of books from Amazon, encompassing both the review score and review comments. Additionally, it includes textual information about the items.

For dataset processing, we adhere entirely to the setup of CoLLM, encompassing label processing and data selection/splitting methods. The statistics of the processed datasets are presented in Table 1.

Compared Methods. In this work, we implement BinLLM with Matrix Factorization (Koren et al., 2009) as the collaborative model in its text-encoding module. To assess the effectiveness of BinLLM, we compare it with four categories of methods: conventional collaborative filtering methods (MF, LightGCN, SASRec, DIN), LLMRec methods without integrating collaborative information (ICL, Prompt4NR, TALLRec), LLMRec methods with integrated collaborative information (PersonPrompt, CoLLM), and methods combining language models and collaborative models (CTRL).

- **MF (Koren et al., 2009)**: This refers to a classic latent factor-based collaborative filtering method — Matrix Factorization.
- **LightGCN (He et al., 2020)**: This is one representative graph-based collaborative filtering method, utilizing graph neural networks to enhance collaborative information modeling.
- **SASRec (Kang and McAuley, 2018)**: This is a representative sequential-based collaborative filtering method that utilizes self-attention for modeling user preferences.
- **DIN (Zhou et al., 2019)**: This is a representative collaborative Click-Through Rate (CTR) model, which employs target-aware attention to activate the most relevant user behaviors, thereby enhancing user interest modeling.
- **CTRL (DIN) (Li et al., 2023b)**: This is a state-of-the-art (SOTA) method for combining language and collaborative models through knowledge distillation. We implement its collaborative model as DIN.
- **ICL (Dai et al., 2023a)**: This is an In-Context Learning-based LLMRec method, which directly asks the original LLM for recommendations.
- **Prompt4NR (Zhang and Wang, 2023)**: This is a state-of-the-art (SOTA) soft prompt tuning-based LLMRec method. Initially designed to leverage the language model (LM), we extend it to utilize LLMs, taking the implementation in CoLLM (Zhang et al., 2023b).
- **TALLRec (Bao et al., 2023b)**: This is a state-of-the-art LLMRec method that aligns LLMs with recommendations through instruction tuning.
- **PersonPrompt (Li et al., 2023a)**: This is a LLMRec method, which integrates collaborative information by adding new tokens and token embeddings to represent users and items. It could be regarded as a personalized soft-prompt tuning method.
- **CoLLM (Zhang et al., 2023b)**: This is a state-of-the-art LLMRec method that integrates collaborative information by mapping collaborative embeddings into the latent space of the LLM. We consider two implementations: CoLLM-MF, which utilizes MF to extract collaborative embeddings, and CoLLM-DIN, which uses the DIN to extract collaborative embeddings.

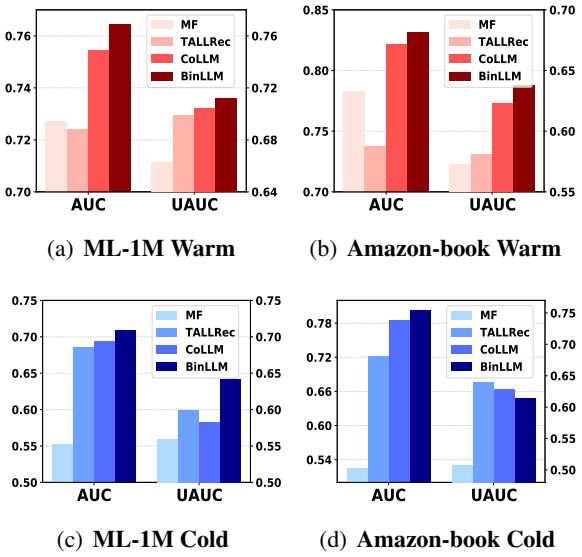
Hyper-parameters and Evaluation Metrics. For all methods, we strictly adhere to the hyperparameter settings outlined in the CoLLM paper (Zhang et al., 2023b), with Vicuna-7B used as the employed LLM. It's worth noting that for our method, we set the dimension of the collaborative embeddings (*i.e.*, the length of the binary representations in Equation (2)) to 32 by default. Considering the length is not very large, we choose not to perform compression in our text-like encoding module by default. We tune the hyper-parameters based on the AUC metric on the validation dataset.

Regarding evaluation metrics, we employ two widely used metrics for click/rating prediction: AUC (Area under the ROC Curve), which measures the overall prediction accuracy, and UAUC (AUC

[2] https://grouplens.org/datasets/movielens/1m/
[3] https://nijianmo.github.io/amazon/index.html

Table 2: Overall performance comparison on the ML-1M and Amazon-Book datasets. "Collab." denotes collaborative recommendation methods. "Rel. Imp." denotes the relative improvement of BinLLM compared to baselines, averaged over the two metrics.

Dataset		ML-1M			Amazon-Book		
Methods		AUC	UAUC	Rel. Imp.	AUC	UAUC	Rel. Imp.
Collab.	MF	0.6482	0.6361	12.9%	0.7134	0.5565	14.7%
	LightGCN	0.5959	0.6499	15.8%	0.7103	0.5639	14.2%
	SASRec	0.7078	0.6884	3.0%	0.6887	0.5714	15.3%
	DIN	0.7166	0.6459	5.6%	0.8163	0.6145	2.0%
LM+Collab.	CTRL (DIN)	0.7159	0.6492	5.4%	0.8202	0.5996	3.0%
LLMRec	ICL	0.5320	0.5268	35.8%	0.4820	0.4856	50.7%
	Prompt4NR	0.7071	0.6739	4.1%	0.7224	0.5881	10.9%
	TALLRec	0.7097	0.6818	3.3%	0.7375	0.5983	8.2%
LLMRec+Collab.	PersonPrompt	0.7214	0.6563	4.5%	0.7273	0.5956	9.9%
	CoLLM-MF	0.7295	0.6875	1.5%	0.8109	0.6225	1.7%
	CoLLM-DIN	0.7243	0.6897	1.7%	0.8245	**0.6474**	-1.0%
Ours	BinLLM	**0.7425**	**0.6956**	-	**0.8264**	0.6319	-



(a) **ML-1M Warm** (b) **Amazon-book Warm**

(c) **ML-1M Cold** (d) **Amazon-book Cold**

Figure 2: Performance comparison in warm and cold scenarios on ML-1M and Amazon-Book. The left y-axis represents AUC, while the right one represents UAUC.

averaged over users), which provides insights into the ranking quality for users.

3.2 Performance Comparison

In this subsection, we initially examine the overall performance of the compared methods and subsequently analyze their performance in warm-start and cold-start scenarios, respectively.

3.2.1 Overall Performance (RQ1)

We summarize the overall performance of the compared methods in Table 2. From the table, we draw the following observations:

- When compared to baselines, our BinLLM achieves the best performance overall, except when compared to CoLLM-DIN on the UAUC metric. These results confirm the superiority of BinLLM in leveraging both collaborative information and the power of LLMs to achieve better recommendation performance.
- Comparing LLMRec methods that integrate collaborative information with LLMRec methods that do not consider collaborative information, we observe that incorporating collaborative information generally improves performance and enables LLMRec to surpass traditional collaborative and LM-based methods. These results underscore the importance of integrating collaborative information into LLMs for recommendation.
- Comparing BinLLM with existing LLMRec methods that also consider collaborative information, our BinLLM consistently outperforms CoLLM-MF and PersonPrompt. Compared with CoLLM-DIN, BinLLM still achieves better results except for the UAUC metric on Amazon-book. Considering that CoLLM-DIN employs a more advanced collaborative model while BinLLM relies solely on MF, these results confirm that encoding collaborative information in a text-like manner better aligns with LLMs, allowing us to leverage their power for recommendation more effectively.
- Among LLMRec methods that consider collaborative information, PersonPrompt, which learns

Table 3: Results of the ablation studies on ML-1M and Amazon-Book, where “TO", “IO", “IT" denote “Text-Only", “ID-Only", “Intuitive-Tuning", respectively.

Datasets	ML-1M		Amazon-book	
Methods	AUC	UAUC	AUC	UAUC
BinMF	0.7189	0.6654	0.8087	0.5895
BinLLM-TO	0.7097	0.6818	0.7375	0.5983
BinLLM-IO	0.7307	0.6797	0.8173	0.5919
BinLLM-IT	0.7286	0.6842	0.8246	0.6165
BinLLM	0.7425	0.6956	0.8264	0.6319

token embeddings for users and items from scratch, performs the worst, significantly lagging behind others. This can be attributed to the low learning efficacy resulting from the introduction of additional tokens and token embeddings.

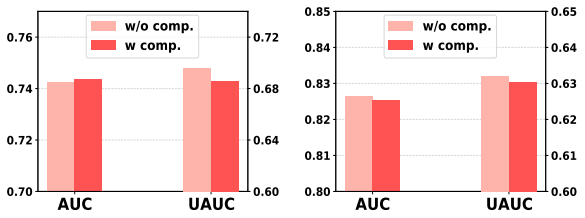
3.2.2 Warm and Cold Performance

When integrating collaborative information into LLMRec, one consideration is to enhance their warm-start performance, enabling them to achieve good performance in both warm-start and cold-start scenarios. We now investigate the performance in the two scenarios. Specifically, we adhere to the protocol outlined in the CoLLM paper (Zhang et al., 2023b) to partition the testing data into warm data and cold data based on the interaction count of users and items, and subsequently evaluate the model on them. We summarize the results in Figure 2. Here, we compare four representative methods: MF, TALLRec, CoLLM-MF, and BinLLM.

According to the figure, in the warm scenarios, TALLRec, an LLMRec method without considering collaborative information, performs worse than MF, while both CoLLM and BinLLM outperform MF, with BinLLM being the best. These results indicate that collaborative information is important for warm-start performance, and our text-like encoding has superiority in combining the information with LLMs. In the cold-start scenarios, all LLMRec methods outperform MF, confirming the superiority of LLMRec in cold-start scenarios. Moreover, BinLLM enhances the cold-start performance compared to CoLLM in most cases, possibly due to the binarized embeddings having better generalization.

3.3 In-depth Analyses (RQ2)

In this subsection, we conduct experiments to analyze the influence of BinLLM’s different components on its effectiveness.



(a) **ML-1M** (b) **Amazon-book**

Figure 3: Performance of BinLLM with (w comp.) and without compression (w/o comp.). The left y-axis represents AUC, while the right one represents UAUC.

3.3.1 Ablation Study

We first further verify the benefits of introducing text-like encoding of collaborative information into LLMs. Specifically, we compare the default BinLLM with the following variants: 1) BinMF, which avoids using the LLM but directly utilizes the binary representations for recommendations like MF, 2) BinLLM-TO, which removes the ID field from BinLLM’s prompt template, *i.e.,* only using the text information, 3) BinLLM-IO, which removes the text field from BinLLM’s prompt, *i.e.,* only using the collaborative information. Additionally, we also study the influence of the two-step tuning by comparing a variant that employs intuitive tuning, denoted by BinLLM-IT. The comparison results are summarized in Table 3.

From the table, we make the following observations: 1) BinMF underperforms all BinLLM variants that consider collaborative information, confirming the superiority of leveraging LLMs for recommendation. 2) BinLLM-TO underperforms other BinLLM variants, indicating that introducing collaborative information is crucial for enhancing LLMRec performance. 3) BinLLM-IO generally underperforms BinLLM-IT and the default BinLLM, highlighting the importance of considering both textual and collaborative information. Lastly, comparing BinLLM-IT with the default BinLLM, BinLLM-IT consistently performs worse. This verifies our claims about tuning designs: directly tuning LLMs with prompts containing collaborative information from scratch may lead to underutilization of both textual and collaborative information.

3.3.2 The Influence of Compression

In the preceding experiments, we did not use compression for our text-like encoding of collaborative information by default. Here, we conduct experiments to study its influence by comparing BinLLM with compression (w comp.) and without compression (w/o comp.). The comparison results of

recommendation performance are summarized in Figure 3. According to the figure, BinLLM with compression generally shows comparable performance to BinLLM without compression. Moreover, when compared with baselines, the comparison trends are similar to BinLLM without compression (with only some differences observed for the UAUC metric on the ML-1M dataset when compared with CoLLM). These results indicate that compression can reduce the representation length while maintaining performance to a large extent.

As shown in Equation (4), the dot-decimal notation can compress the length of collaborative representation by approximately 2.5 times. However, in our experiments, the inference acceleration did not reach this level. This is because we only included the collaborative representations for the target user and items, which constitute a smaller part of the total prompt. Specifically, the inference time for BinLLM without compression and with compression was 106s and 93s on ML-1M, and 483s and 435s on Amazon, respectively. If considering collaborative information for all historically interacted items, as done by Liao et al. (2023), the expected inference acceleration would be more significant.

4 Related Work

• **Collaborative Information Modeling.** Collaborative information modeling is pivotal for personalized recommendations, and significant efforts have been dedicated to this area in traditional research. Initially, the information modeling relied on statistical methods (Sarwar et al., 2001). Subsequently, latent factor models became prevalent, leading to the development of prominent models such as MF (Koren et al., 2009) and FISM (Kabbur et al., 2013). Later, neural network-enhanced latent factor models made substantial advancements (He et al., 2017; Tang and Wang, 2018; Hidasi et al., 2016). These studies achieved remarkable success in both academia and industry, inspiring exploration into collaborative information modeling for LLMRec. In this study, we propose a method to encode collaborative information in a text-like format, making it suitable for LLM usage.

• **LLMRec.** As the impressive capabilities exhibited by LLMs, an increasing number of researchers in the recommendation community are now exploring the potential of applying LLMs to recommendation systems (Wu et al., 2023; Lin et al., 2023). This exploration can be categorized into two groups. The first group focuses on directly harnessing the abilities of LLMs by employing suitable prompts to stimulate their performance in recommendation scenarios (Dai et al., 2023b; Hou et al., 2023). On the other hand, another group of researchers argues that LLMs have limited exposure to recommendation tasks during pre-training, and recommendation data often possess personalized characteristics (Bao et al., 2023b; Zhang et al., 2023a). Consequently, it becomes crucial to explore tuning methods that can enhance the recommendation performance of LLMs. As researchers delve deeper into their studies, it has been discovered that LLMs often exhibit an excessive reliance on semantic knowledge for learning, while paying insufficient attention to the acquisition of collaborative information between entities (Bao et al., 2023a).

Researchers have initiated endeavors to incorporate collaborative information into LLMs. Some researchers attempt to look for ID encoding methods to introduce new tokens through vocabulary expansion and train these tokens from scratch (Zheng et al., 2023; Hua et al., 2023; Rajput et al., 2023). Among them, Hua et al. utilize statistical information, Zheng et al. and Rajput et al. employ vector quantization techniques. However, this approach often faces with low learning efficacy. Another group of researchers explores using a latent factor model to capture collaborative information (Zhang et al., 2023b; Li et al., 2023c; Liao et al., 2023), which is then mapped onto the semantic space of LLMs through a mapping layer. This method exhibits better learning efficacy but requires additional training of the mapping layer. Moreover, due to the non-text-like format of collaborative information, both sets of methods face challenges in aligning with the information processing mechanism in LLMs, limiting their performance.

5 Conclusion

In this study, we emphasize the importance of text-like encoding of collaborative information modeling to enhance recommendation performance for LLMRec. We introduce BinLLM, a novel approach designed to incorporate collaborative information in a text-like format by binarizing collaborative embeddings for LLMRec. This encoding allows the collaborative information to be utilized in a manner better aligned with how information is processed in LLMs. Extensive results demonstrate the superiority of BinLLM.

6 Limitations

Currently, this paper has certain limitations in experimental validation: 1) It relies solely on Vicuna-7B for experiments; 2) The current experiments focus solely on rating/click prediction tasks, neglecting other recommendation tasks like next-item prediction. In the future, we aim to expand experiments accordingly. Additionally, at the methodological level, similar to existing LLMRec methods, this paper faces challenges with low inference efficiency for real-world recommendation scenarios, particularly in the all-ranking setting. In the future, we could explore applying existing acceleration methods like pruning to improve speed. Moreover, exploring recommendation generation methods that avoid multiple inferences for individual users is another avenue worth exploring.

7 Ethical Considerations

In this paper, we present BinLLM, designed to encode collaborative information in a text-like format for LLMRec. Our method binarizes numerical embeddings and thus doesn't raise ethical concerns. Moreover, the data we use are publicly available and don't include sensitive details like gender. However, recommendations involve user behavioral data, which might raise privacy concerns, which are addressable through introducing the mechanism of user consent. Additionally, using LLMs may have hidden negative societal biases. We advocate for conducting thorough risk assessments and advise users to be wary of potential risks linked with model usage.

References

Fadi Abusafat, Tiago Pereira, and Henrique Santos. 2021. Roadmap of security threats between ipv4/ipv6. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pages 1–6. IEEE.

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

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

Keqin Bao, Jizhi Zhang, Yang Zhang, Wang Wenjie, Fuli Feng, and Xiangnan He. 2023c. Large language models for recommendation: Progresses and future directions. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 306–309.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023a. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*, page 1126–1132, New York, NY, USA. Association for Computing Machinery.

Sunhao Dai et al. 2023b. Uncovering chatgpt's capabilities in recommender systems. In *RecSys*, pages 1126–1132. ACM.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. 2023. Language modeling is compression.

F. Maxwell Harper and Joseph A. Konstan. 2016. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.

Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1096–1102.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 639–648. ACM.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182.

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

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 preprint arXiv:2305.08845*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of

large language models. In *The Tenth International Conference on Learning Representations*. OpenReview.net.

Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2023. How to index item ids for recommendation foundation models. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP '23, page 195–204, New York, NY, USA. Association for Computing Machinery.

Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, page 659–667, New York, NY, USA. Association for Computing Machinery.

Wang-Cheng Kang and Julian J. McAuley. 2018. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 197–206. IEEE Computer Society.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Lei Li, Yongfeng Zhang, and Li Chen. 2023a. Personalized prompt learning for explainable recommendation. *ACM Trans. Inf. Syst.*, 41(4).

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

Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023c. E4srec: An elegant effective efficient extensible solution of large language models for sequential recommendation. *arXiv preprint arXiv:2312.02443*.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, and Xiang Wang. 2023. Llara: Aligning large language models with sequential recommenders.

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 preprint arXiv:2306.05817*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Larry L Peterson and Bruce S Davie. 2007. *Computer networks: a systems approach*. Elsevier.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. 2023. Recommender systems with generative retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295.

Jaromir Savelka, Arav Agarwal, Marshall An, Chris Bogart, and Majd Sakr. 2023. Thrilled by your progress! large language models (gpt-4) no longer struggle to pass assessments in higher education programming courses. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 1*, ICER '23, page 78–92, New York, NY, USA. Association for Computing Machinery.

Qiaoyu Tan, Ninghao Liu, Xing Zhao, Hongxia Yang, Jingren Zhou, and Xia Hu. 2020. Learning to hash with graph neural networks for recommender systems. In *Proceedings of The Web Conference 2020*, WWW '20, page 1988–1998, New York, NY, USA. Association for Computing Machinery.

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

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *WSDM 2024*.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860*.

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

Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2023b. Collm: Integrating collaborative embeddings into large language models for recommendation. *arXiv preprint arXiv:2310.19488*.

Zizhuo Zhang and Bang Wang. 2023. Prompt learning for news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval,*

SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, pages 227–237. ACM.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Adapting large language models by integrating collaborative semantics for recommendation. *arXiv preprint arXiv:2311.09049*.

Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 5941–5948. AAAI Press.