GraphLoRA: Structure-Aware Low-Rank Adaptation for Large Language Model Recommendation

Lin Mu[1], Guoji Wang[1], Li Ni[1], Lei Sang[1], Zhize Wu[2], Peiquan Jin[3], Yiwen Zhang[1]*

[1]Anhui University, [2]Hefei University,

[3]University of Science and Technology of China,

{mulin, nili, sanglei, zhangyiwen}@ahu.edu.cn {wangguoji}@stu.ahu.edu.cn

wuzz@hfuu.edu.cn jpq@ustc.edu.cn

Abstract

Large Language Models (LLMs) have shown strong potential for recommendation (LLM-Rec) due to their powerful reasoning and generalization abilities. However, effectively aligning the textual semantics modeled by LLMs with the collaborative signals remains a key challenge. Existing methods either translate collaborative information into textual prompts or inject pre-trained embeddings into the LLM, both of which treat structural information as static input and fail to capture high-order relational dependencies. To bridge this gap, we propose **GraphLoRA**, a novel framework that generalizes low-rank adaptation from independent to structure-aware propagation. GraphLoRA embeds a trainable graph message-passing network within the low-rank adaptation pathway, enabling structural signals to propagate through the parameter space. This design allows collaborative topology to explicitly guide parameter updates, fostering deep integration between graph-structured and textual semantic information. Extensive experiments on multiple benchmarks demonstrate that GraphLoRA not only outperforms state-of-the-art LLM-based recommendation methods but also achieves superior generalization, effectively balancing structural reasoning capability with computational efficiency. Code is available at https://github.com/wgj15965/GraphLoRA.

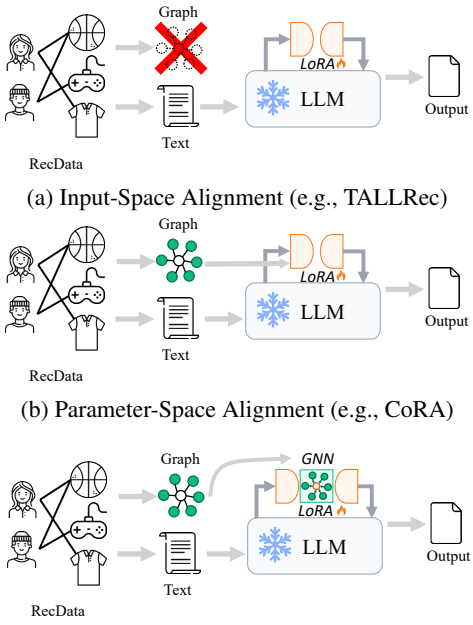

Figure 1: Comparison of collaborative alignment paradigms. **(a) Input-Space Alignment** converts interaction histories into textual prompts for the LLM input. **(b) Parameter-Space Alignment** injects static, externally encoded embeddings into LLM weights (e.g., via LoRA). **(c) Structure-Aware Alignment (Ours)** integrates a learnable GNN within the LoRA bottleneck (between A and B) to perform dynamic message passing directly in the parameter space.

1 Introduction

Recommender systems play a central role in personalized information access, enabling users to efficiently discover relevant content (Wu et al., 2022). Recently, large language models (LLMs) have emerged as powerful recommenders due to their remarkable capabilities in reasoning, generalization, and contextual understanding (Liu et al., 2025a; Wu et al., 2024). However, a fundamental challenge persists: effectively align textual semantics—naturally handled by LLMs—with collaborative information derived from user–item interactions, which is inherently structured. Bridging this gap is essential for unleashing the full potential of LLM-based recommendation.

Early efforts in LLM-based recommendation primarily relied on *Input-Space Alignment* (see Figure 1(a)), where collaborative information was translated into prompts or input tokens. For example, TALLRec (Bao et al., 2023) linearized user-item interaction histories into textual prompts,

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 13208–13218
July 2-7, 2026

while CoLLM (Zhang et al., 2025) projected user and item embeddings-obtained from matrix factorization (MF)-into soft tokens. Although these methods allow LLMs to access collaborative signals, they force the model to passively "read" structural information rather than internally reason about relational dependencies. Moreover, linearized or tokenized representations struggle to capture the complex, high-order topology of user–item interaction graphs, leading to a loss of structural inductive bias (Huang et al., 2024).

To strengthen internal modeling, recent research has pivoted towards *parameter-space alignment*, as illustrated in Figure 1(b) (e.g., CoRA (Liu et al., 2025b)). Instead of modifying textual input, CoRA directly injects collaborative signals into LLM's parameters (e.g., via LoRA), enabling the model to internalize user–item relationships within its parameter space. This design overcomes the limitations of input-space alignment by allowing the LLM to encode collaborative patterns rather than merely read them. However, CoRA still depends on externally pre-trained embeddings (e.g., from matrix factorization) that are injected as static weights. Consequently, structural information remains external and cannot be jointly refined with the model's semantic representations. To address this limitation, we advocate for *structure-aware alignment* within the LLM itself. We reinterpret LoRA not as a passive fine-tuning module, but as a learnable reasoning pathway capable of propagating collaborative signals in the parameter space. This perspective generalizes the conventional LoRA formulation—from independent low-rank adaptation to structure-aware low-rank propagation—thereby enabling the LLM to internalize collaborative topology and jointly refine semantic and structural representations.

Building upon this insight, we propose **GraphLoRA**, a novel framework that realizes this ***structure-aware alignment*** (Figure 1(c)). GraphLoRA embeds a differentiable graph message-passing module within the LoRA bottleneck (between A and B), constructing a learnable structural pathway directly in the low-rank latent space. Specifically, collaborative representations are first projected into the low-rank latent space, where high-order neighborhood information is dynamically aggregated through graph message passing, and the enriched structural embeddings are then projected back to the parameter space to update the model. This mechanism allows collaborative information to explicitly guide parameter updates, fostering deep interaction between collaborative topology and semantic understanding. Through joint optimization of the structural encoder and the LLM adaptation under a unified objective, GraphLoRA effectively distills neighborhood-aware structural signals—achieving deep structure–semantic synergy while preserving the LLM's linguistic competence.

The contributions can be summarized as follows:

- We propose **GraphLoRA**, which integrates a graph message-passing module within the LoRA bottleneck, generalizing independent adaptation to structure-aware propagation and enabling LLMs to internalize collaborative topology.
- We design a joint optimization framework that couples graph-structured reasoning with semantic adaptation, yielding representations that are both structure-aware and linguistically coherent.
- Extensive experiments on multiple benchmark datasets demonstrate that GraphLoRA consistently outperforms state-of-the-art LLM-based recommendation methods under compact parameter settings. Additional ablation and efficiency studies further confirm its robustness, parameter efficiency, and structural interpretability across diverse recommendation scenarios.

2 Related Works

2.1 LLM-based Recommendation

The application of large language models (LLMs) in recommender systems has evolved rapidly. Early works primarily utilized in-context learning (Dong et al., 2024), leveraging the extensive world knowledge of LLMs via carefully crafted prompts to perform recommendation tasks without parameter updates (Gao et al., 2023; Dai et al., 2023). While cost-effective, these methods often struggle to capture domain-specific collaborative signals.

To address this, subsequent research shifted towards instruction tuning with *input-space alignment*. Foundational frameworks like P5 (Geng et al., 2022) laid the groundwork by unifying recommendation tasks into sequence generation. Following this, a representative method, TALLRec (Bao et al., 2023), reformulated interaction

data into linear textual instructions. To further incorporate dense collaborative signals, methods like CoLLM (Zhang et al., 2025) and LlaRA (Liao et al., 2024) embedded user/item ID embeddings from external encoders directly into the input prompt as soft tokens, while BinLLM (Zhang et al., 2024) encoded such information into binary sequences. However, this paradigm often results in a separation between collaborative modeling and LLM reasoning. Mapping complex graph topology into linear sequences or static tokens can impose an information bottleneck, effectively flattening the structural context before it reaches the LLM.

2.2 Graph-Enhanced LLMs and Structure-Aware Tuning

Recognizing the importance of structural data, researchers have explored integrating graph neural networks (GNNs) (Kipf and Welling, 2017) with LLMs. One line of research adopts a projector-based approach, where a graph encoder is aligned with the LLM via a projection layer. For instance, GraphGPT (Tang et al., 2024a) and HiGPT (Tang et al., 2024b) employed graph instruction tuning to align graph structural knowledge with the LLM's token space, while GIMLET (Zhao et al., 2023) unified modalities for molecule tasks. Despite their effectiveness, tuning the projector or the entire model can be computationally expensive.

Another emerging direction focuses on parameter-space alignment via parameter-efficient fine-tuning (PEFT) (Ding et al., 2023). Although general-purpose PEFT methods like Adapter (Houlsby et al., 2019) and Prefix-Tuning (Li and Liang, 2021) have proven effective in NLP, and recent advancements continue to optimize the density and efficiency of adaptation matrices (e.g., DenseLoRA (Mu et al., 2025)), recommendation works specifically leverage low-rank adaptation (LoRA) (Hu et al., 2022) to modify model weights. A representative method, CoRA (Liu et al., 2025b), injected collaborative signals directly into these adaptation matrices. To bridge this gap, our proposed GraphLoRA embeds a trainable message-passing GNN within the LoRA bottleneck, thereby enabling dynamic aggregation of high-order structural signals directly in the parameter space.

3 Background

3.1 Low-Rank Adaptation (LoRA)

LoRA (Hu et al., 2022) adapt LLMs by freezing pre-trained weights while injecting trainable rank decomposition matrices. For a linear layer with pre-trained weights $\mathbf{W}_0 \in \mathbb{R}^{d_{model} \times d_{model}}$, the weight update is parameterized by two low-rank matrices: a down-projection $\mathbf{A} \in \mathbb{R}^{r \times d_{model}}$ and an up-projection $\mathbf{B} \in \mathbb{R}^{d_{model} \times r}$, where $r \ll d_{model}$ is the intrinsic rank. The forward pass for an input hidden state $\mathbf{x} \in \mathbb{R}^{d_{model}}$ is computed as:

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \frac{\alpha}{r}\mathbf{B}\mathbf{A}\mathbf{x}, \tag{1}$$

where α is a scaling scalar. In GraphLoRA, we intervene in the latent space between $\mathbf{A}$ and $\mathbf{B}$ to inject collaborative signals.

3.2 Message-Passing GNNs

Recommendation data typically forms a user-item bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{I}, \mathcal{E})$. To capture high-order collaborative signals without conflicting with LoRA's notation, we describe GNNs using the general message passing neural network (MPNN) paradigm (Gilmer et al., 2017). MPNN defines structural learning via local aggregation. For a node u, its representation $\mathbf{e}_u^{(l)}$ at layer l is updated by aggregating messages from its neighbors $\mathcal{N}_u$:

$$\mathbf{m}_u^{(l)} = \text{AGG}\left(\left\{\phi(\mathbf{e}_v^{(l)}, \mathbf{e}_u^{(l)}) : v \in \mathcal{N}_u\right\}\right), \tag{2}$$

$$\mathbf{e}_u^{(l+1)} = \psi\left(\mathbf{e}_u^{(l)}, \mathbf{m}_u^{(l)}\right), \tag{3}$$

where $\phi(\cdot)$ is the message function, AGG$(\cdot)$ is a permutation-invariant aggregation operator (e.g., Mean, Sum), and $\psi(\cdot)$ is the update function. This unified view encompasses both general graph encoders (e.g., GraphSAGE (Hamilton et al., 2017), GAT (Veličković et al., 2018)) and recommendation-specific variants (e.g., LightGCN (He et al., 2020), NGCF (Wang et al., 2019)). In this work, we adopt the latter category as the structural backbone, given their proven effectiveness in modeling collaborative filtering signals.

4 Methodology

In this section, we present GraphLoRA, a unified structure-aware framework aligning collaborative representations with textual semantics. Formally, let the input sequence be $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, a hybrid sequence comprising standard text tokens and

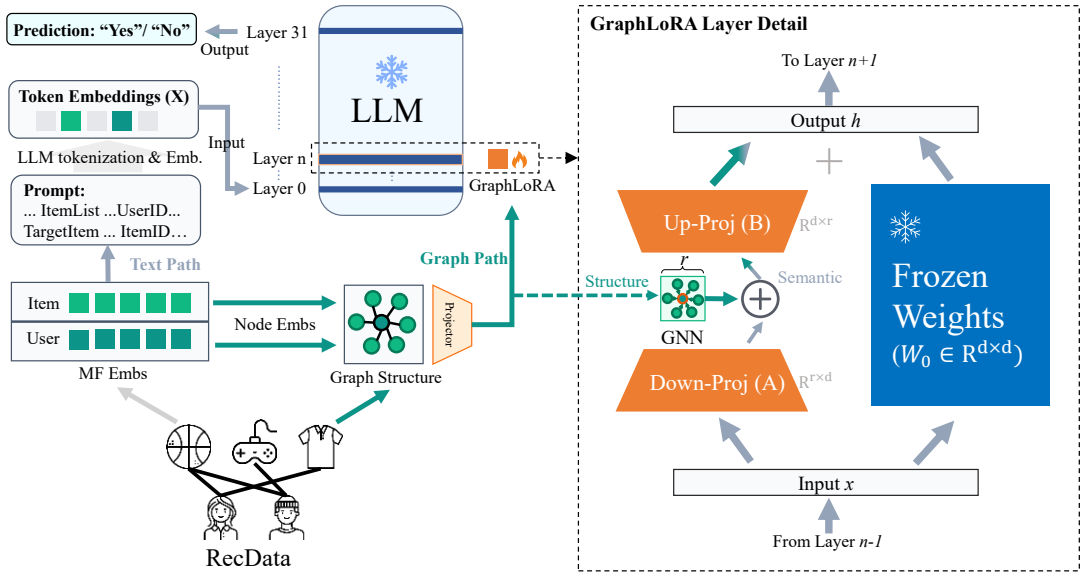

Figure 2: Overview of **GraphLoRA**. (Left) The end-to-end recommendation process fusing text prompts and collaborative signals. (Right) The detailed structure of the GraphLoRA layer, where a GNN-based structural encoder is injected between the low-rank matrices $\mathbf{A}$ and $\mathbf{B}$ to realize structure-aware low-rank propagation. Note: "Embs" refers to "Embeddings".

collaborative tokens (i.e., users $\mathbf{x}_{userid}$ and items $\mathbf{x}_{itemid}$). Here, $\mathbf{x}_t \in \mathbb{R}^{d_{model}}$ denotes the embedding of the t-th token. GraphLoRA specifically intervenes in the processing of these collaborative tokens.

As illustrated in Figure 2, the workflow proceeds in four stages: (1) *Collaborative Initialization*, defining the source embeddings; (2) *Dual-View Input Construction*, mapping these embeddings into the input sequence $\mathbf{X}$; (3) *Graph Structural Encoding*, refining the embeddings via message passing; and (4) *Structure-Aware Parameter Injection*, fusing structural signals into the processing of $\mathbf{x}_t$ in the parameter space.

4.1 Collaborative Initialization

To ground user and item identities, we employ a set of learnable embeddings rooted in matrix factorization (MF). We approximate the interaction matrix via latent factors $\mathbf{P} \in \mathbb{R}^{|\mathcal{U}| \times d_{emb}}$ for users and $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}| \times d_{emb}}$ for items, where d_{emb} denotes the collaborative embedding dimension. The unified collaborative embedding matrix is denoted as:

$$\mathbf{E} = [\mathbf{P}; \mathbf{Q}] \in \mathbb{R}^{(|\mathcal{U}|+|\mathcal{I}|) \times d_{emb}}. \tag{4}$$

Rationale. Instead of using frozen external features, we treat $\mathbf{E}$ as a shared, learnable state. It feeds both the LLM input and the graph encoder, and is jointly optimized with the LLM. This ensures collaborative signals remain dynamically aligned with the model's semantic reasoning.

4.2 Dual-View Input Construction

Here, we describe how collaborative data is mapped into the LLM's input space $\mathbf{X}$. We construct textual prompts using templates (Table 1). To align the collaborative dimension (d_{emb}) with the LLM's model dimension (d_{model}), we employ an input projection network $\mathcal{F}$.

For a collaborative placeholder token (e.g., `<UserID>` or `<TargetItemID>`) at position t within sequence b, its representation $\mathbf{x}_t$ is derived directly from the current embedding state:

$$\mathbf{x}_t = \mathcal{F}(\mathbf{e}_n), \quad \mathbf{x}_t \in \mathbb{R}^{d_{model}}, \tag{5}$$

where $\mathbf{e}_n$ corresponds to the embedding of the associated collaborative node (user or item) in $\mathbf{E}$. We record the set of collaborative anchor positions as $\mathcal{S} = \{(b, t)\}$, where b indexes the sequence (i.e., sample) within a mini-batch and t denotes the token position in that sequence. For each anchor $(b, t) \in \mathcal{S}$, we maintain a deterministic mapping $\pi(b, t) \mapsto n$ that links the collaborative token to its corresponding graph node n (user or item), enabling structure injection at the correct positions. At this stage, $\mathbf{x}_t$ contains identity information but lacks explicit topological context.

#Question: A user has given high ratings to the following books: <ItemTitleList>. Additionally, we have information about the user's preferences encoded in the feature <UserID>. Using all available information, make a prediction about whether the user would enjoy the book titled <TargetItemTitle> with the feature <TargetItemID>? Answer with "Yes" or "No".

#Answer: "Yes" or "No".

Table 1: Example of the prompt template used in GraphLoRA.

4.3 Graph Structural Encoding

Parallel to the LLM input processing, we generate high-order structural signals. Given the user-item subgraph and the shared embeddings $\mathbf{E}$, the graph encoder performs message passing across L layers. Following the MPNN paradigm (Section 3.2), the structural representation for node n evolves by aggregating messages from neighbors $\mathcal{N}_n$:

$$\mathbf{e}_n^{(k+1)} = \psi\Big(\mathbf{e}_n^{(k)}, \underset{v\in\mathcal{N}_n}{\text{AGG}}\left(\phi(\mathbf{e}_v^{(k)}, \mathbf{e}_n^{(k)})\right)\Big). \quad (6)$$

After L layers, we obtain the structure-enhanced representation $\mathbf{e}_n^{(L)} \in \mathbb{R}^{d_{emb}}$.

Bottleneck Alignment. Typically, d_{emb} is much larger than the LoRA intrinsic rank (r). To align these spaces, we employ a bottleneck projection $\mathbf{W}_{neck} \in \mathbb{R}^{r\times d_{emb}}$:

$$\mathbf{z}_n = \mathbf{W}_{neck}\mathbf{e}_n^{(L)}, \quad \mathbf{z}_n \in \mathbb{R}^r, \quad (7)$$

where $\mathbf{z}_n$ represents the compressed, task-specific structural signal ready for injection.

4.4 Structure-Aware Parameter Injection

Finally, we describe how the graph signal $\mathbf{z}_n$ is injected into the processing of $\mathbf{x}_t$. We adopt a sparse single-layer injection strategy targeting a specific layer l^*.

Recall that standard LoRA computes the update as $\frac{\alpha}{r}\mathbf{BAx}$. We intervene in the low-rank latent space between the down-projection $\mathbf{A}$ and up-projection $\mathbf{B}$. For a collaborative token $\mathbf{x}_t$ at position t (i.e., $(b,t) \in \mathcal{S}$), where the associated node is $n = \pi(b,t)$ (user or item), the injection flow is:

1. Semantic Compression: The input state $\mathbf{x}_t$ is compressed by matrix $\mathbf{A}$ to obtain the semantic intermediate representation $\mathbf{h}_{sem}$:

$$\mathbf{h}_{sem} = \mathbf{A}\mathbf{x}_t, \quad \mathbf{h}_{sem} \in \mathbb{R}^r. \quad (8)$$

2. Structural Fusion: We fuse the graph signal $\mathbf{z}_n$ with the semantic signal $\mathbf{h}_{sem}$ to produce the structure-enhanced latent state $\mathbf{h}_{latent}$:

$$\mathbf{h}_{latent} = \lambda_{lora}\mathbf{h}_{sem} + \lambda_{gnn}\mathbf{z}_n, \quad (9)$$

where $\lambda_{lora}, \lambda_{gnn}$ are balancing coefficients.

3. Manifold Projection: The fused signal is projected back to the LLM space by the up-projection matrix $\mathbf{B}$ to yield the structure-aware update, producing the final hidden state $\mathbf{h}$:

$$\Delta\mathbf{x}_t = \mathbf{B}\mathbf{h}_{latent}, \quad \mathbf{h} = \mathbf{W}_0\mathbf{x}_t + \frac{\alpha}{r}\Delta\mathbf{x}_t. \quad (10)$$

This formulation ensures that the graph structural signal explicitly guides the gradient updates.

For non-collaborative tokens ($(b,t) \notin \mathcal{S}$), the structural pathway is inactive (i.e., $\mathbf{z}_n$ is omitted), and the model follows the standard LoRA computation. Crucially, the GNN parameters and $\mathbf{W}_{neck}$ are jointly optimized with the LLM. During backpropagation, gradients flow through $\mathbf{B} \to \mathbf{z}_n \to \mathbf{W}_{neck} \to$ GNN, ensuring that $\mathbf{z}_n$ evolves into a *semantically aligned* structural bias that directly complements the reasoning over $\mathbf{x}_t$.

4.5 Complexity Analysis

Parameter Efficiency. GraphLoRA attains structure awareness with minimal parameter overhead (e.g., $\sim 1.67\%$ over the LoRA-only baseline) via Bottleneck Alignment and Sparse Injection. Unlike input alignment requiring the full hidden space d_{model}, GraphLoRA projects signals into the bottleneck r. This reduces projection complexity from $\mathcal{O}(d_{model} \times d_{emb})$ to $\mathcal{O}(r \times d_{emb})$, avoiding high-dimensional redundancy. Moreover, aligning structure with the task-relevant low-rank subspace preserves high information density and enables more effective joint optimization with LoRA parameters. Together, these designs keep GraphLoRA lightweight while ensuring faithful and effective structural injection.

Computational Overhead. The graph message passing is performed on sampled subgraphs, yielding linear complexity $O(|\mathcal{E}_{sub}|)$. Structural embeddings are computed once per batch and injected into the target layer, decoupling structural encoding from the LLM's depth. As a result, GraphLoRA introduces only minimal training overhead while

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

Table 2: Statistics of the processed datasets.

maintaining an effective trade-off between structural reasoning performance and computational efficiency, which is empirically validated in Section 5.

5 Experiments

In this section, we evaluate the effectiveness, efficiency, and generalization of GraphLoRA.

5.1 Experimental Setup

Datasets. We conduct experiments on two standard benchmarks widely used in recommendation research: **ML-1M**[*] (Harper and Konstan, 2015) and **Amazon-Book**[†] (He and McAuley, 2016). To ensure a fair comparison with baseline methods, we strictly follow the data preprocessing protocols established in CoRA (Liu et al., 2025b). The detailed statistics of the processed datasets are listed in Table 2.

Evaluation Metrics. To comprehensively assess the recommendation performance, we adopt two widely recognized metrics: **AUC** (Area Under the ROC Curve) and **UAUC** (User-averaged AUC) (Liu et al., 2021). AUC evaluates the global ranking capability of the model across all samples, while UAUC calculates the AUC for each user individually and reports the average.

Implementation Details. We employ Vicuna-7B (Chiang et al., 2023) as the backbone LLM for our main experiments. The collaborative encoder uses MF ($d = 256$) and a 3-layer NGCF (Wang et al., 2019) to capture structural signals. To balance high-fidelity collaborative signals with extreme parameter efficiency, we adopt a first-order (1-hop) neighbor sampling strategy for direct GNN aggregation. We set the LoRA rank to $r = 8$ and adapt the query (q) and value (v) projections. GraphLoRA adopts a sparse single-layer injection strategy. Based on validation tuning, we inject structural signals into the 31st layer for ML-1M and the 15th layer for Amazon-Book, with $\lambda_{lora} = 1.0$ and $\lambda_{gnn} = 0.1$.

[*] https://grouplens.org/datasets/movielens/1m/

[†] https://jmcauley.ucsd.edu/data/amazon/index_2014.html

All experiments are conducted on 4 NVIDIA RTX 3090 (24GB) GPUs. We train end-to-end with BCE loss, using AdamW (for the LLM) and Adam (for the GNN). We grid-search learning rates in $\{5e^{-4}, 1e^{-4}, 5e^{-5}\}$ and weight decay in $\{1e^{-3}, 1e^{-4}\}$. Unless otherwise specified, we use a global batch size of 8, learning rate $1e^{-4}$, cosine scheduling with warm-up, and maximum sequence length 1024.

Baselines. We compare GraphLoRA with baselines from four paradigms: **(1) Conventional Collaborative Filtering:** MF (Koren et al., 2009), NGCF (Wang et al., 2019), LightGCN (He et al., 2020), and SASRec (Kang and McAuley, 2018). **(2) LLM-based Recommendation:** ICL (Dai et al., 2023), Prompt4NR (Zhang and Wang, 2023), and TALLRec (Bao et al., 2023). **(3) Input-Space Alignment:** PersonPrompt (Li et al., 2023), CoLLM (Zhang et al., 2025), and BinLLM (Zhang et al., 2024). **(4) Parameter-Space Alignment:** CoRA (Liu et al., 2025b), including CoRA-MF, CoRA-LightGCN, and CoRA-SAS.

Baseline configurations. We follow the default settings in the original papers and/or official implementations, and tune the remaining hyperparameters based on validation AUC. Adapter configurations are kept as in the original methods (e.g., CoRA uses $r = 16$ on $\{q, k, v, o\}$, while CoLLM/TALLRec/BinLLM use r=8 on $\{q, v\}$), and we report them in tables for transparency.

5.2 Performance Comparison

We compare GraphLoRA with representative baselines on ML-1M and Amazon-Book. Table 3 reports overall results, and Figure 3 further decomposes performance under warm/cold start regimes.

Overall Results. GraphLoRA consistently achieves the best performance across both datasets and all evaluation metrics. Compared with strong *input-space alignment* baselines, it delivers substantial gains, indicating that injecting collaborative signals directly into the parameter space enables tighter integration between structural information and prompt-conditioned semantics than shallow input augmentation.

Comparison with Parameter-Space Alignment. We compare GraphLoRA with the parameter-space alignment baseline CoRA. While CoRA injects high-capacity adaptations across all Transformer layers, GraphLoRA uses a sparse, targeted injec-

Dataset		ML-1M		Amazon-Book	
Methods (Settings)		AUC	UAUC	AUC	UAUC
Collab.	MF	0.6486	0.6396	0.7105	0.5543
	LightGCN	0.5858	0.6512	0.7026	0.5619
	NGCF	0.6248	0.5991	0.7091	0.5411
	SASRec	0.7005	0.6734	0.6675	0.5614
LLMRec	ICL	0.5119	0.5178	0.5180	0.5043
	Prompt4NR	0.7027	0.6713	0.6527	0.5011
	TALLRec	0.7044	0.6741	0.6583	0.4971
Input-Space	PersonPrompt	0.7014	0.6503	0.7113	0.5596
	CoLLM-MF ($r = 8, \{q, v\}$)	0.7028	0.6714	0.8021	0.5782
	CoLLM-LightGCN ($r = 8, \{q, v\}$)	0.7164	0.6842	0.7835	0.5663
	CoLLM-SAS ($r = 8, \{q, v\}$)	0.7059	0.6531	0.7538	0.5874
	BinLLM ($r = 8, \{q, v\}$)	0.7132	0.6815	0.8157	0.5724
Param-Space	CoRA-MF ($r = 16, \{q, k, v, o\}$)	<u>0.7361</u>	0.6884	<u>0.8179</u>	<u>0.6262</u>
	CoRA-LightGCN ($r = 16, \{q, k, v, o\}$)	0.7128	<u>0.6966</u>	0.7886	0.5689
	CoRA-SAS ($r = 16, \{q, k, v, o\}$)	0.7019	0.6517	0.7677	0.5961
Ours	**GraphLoRA** ($r = 8, \{q, v\}$)	**0.7472**	**0.7102**	**0.8205**	**0.6303**

Table 3: Performance comparison with state-of-the-arts. The best results are highlighted in **bold**, and the second-best results are <u>underlined</u>. Parameter configurations are reported for transparency. Notably, the baselines employ their optimal reported settings (e.g., $r = 16, \{q, k, v, o\}$ for CoRA), whereas GraphLoRA achieves SOTA performance under a strict, parameter-efficient constraint ($r = 8, \{q, v\}$).

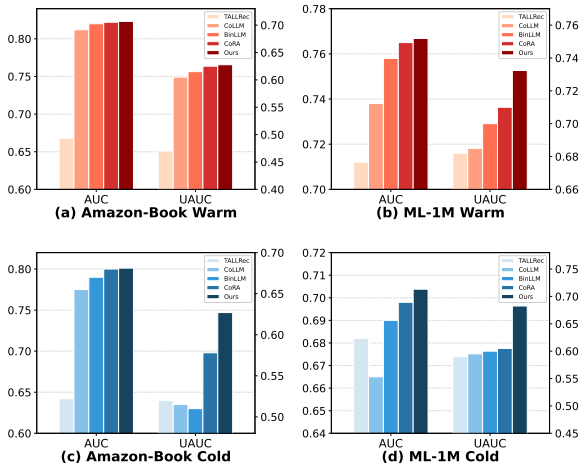

Figure 3: Warm/cold evaluation on Amazon-Book and ML-1M. The left and right y-axes correspond to AUC and UAUC, respectively.

tion with a small rank(e.g., r=8) at a single-layer. Despite this compact design, GraphLoRA achieves comparable or better performance, highlighting a more favorable performance–efficiency trade-off.

Warm and Cold Start Analysis. We split the test set into warm and cold subsets based on user interaction frequency (Figure 3). In warm-start, GraphLoRA achieves performance comparable to the strongest baseline, whereas in the cold-start scenario, it exhibits a more pronounced advantage. This observation aligns with our design intuition: the jointly optimized graph encoder aggregates neighborhood context, providing informative structural signals that help alleviate data sparsity for users with limited interaction histories.

Matched-Budget Fairness and Top-K Ranking. To further validate that GraphLoRA's superiority stems from its structure-aware injection mechanism rather than parameter configuration advantages, we re-evaluate the strongest baseline, CoRA-MF, under the exact same constrained parameter budget as GraphLoRA ($r = 8, \{q, v\}$). Furthermore, to provide a comprehensive evaluation beyond binary prediction, we introduce NDCG@10 to assess top-K ranking performance.

As shown in Table 4, when the parameter budget is strictly restricted, the baseline CoRA-MF experiences a severe "perception collapse" (e.g., UAUC dropping to 0.4995 on Amazon-Book). In contrast, GraphLoRA remains highly effective under identical low-rank constraints. Moreover, GraphLoRA consistently achieves superior NDCG@10 scores, demonstrating its ability to robustly internalize collaborative topology and translate it into high-quality top-list recommendations without relying on parameter inflation.

5.3 Ablation and Efficiency Analysis

Training Strategy and Components. We examine progressive settings motivated by the two roles

Dataset	Metric	CoRA-MF ($r = 8, \{q, v\}$)	GraphLoRA (Ours) ($r = 8, \{q, v\}$)
ML-1M	AUC	0.7227	**0.7472**
	UAUC	0.6794	**0.7102**
	NDCG@10	0.7636	**0.7847**
Amazon-Book	AUC	0.7562	**0.8205**
	UAUC	0.4995	**0.6303**
	NDCG@10	0.6990	**0.7067**

Table 4: Matched-budget comparison and Top-K ranking performance. Both methods are evaluated under identical parameter constraints ($r = 8$ on $\{q, v\}$).

Method Settings	ML-1M		Amazon-Book	
	AUC	UAUC	AUC	UAUC
(A) Training Strategy & Components				
1. LoRA-only(Frozen MF)	0.6981	0.6548	0.8012	0.6026
2. LoRA-only(Trainable MF)	0.7178	0.6952	0.8136	0.6153
3. GraphLoRA (Ours)	**0.7472**	**0.7102**	**0.8205**	**0.6303**
(B) Graph Encoder Variants				
GraphLoRA (GCN)	0.7417	0.6934	0.8168	0.6277
GraphLoRA (LightGCN)	0.7382	0.7034	0.8132	0.6170
GraphLoRA (NGCF)	**0.7472**	**0.7102**	**0.8205**	**0.6303**

Table 5: Ablation of (A) training strategy/components and (B) graph encoder variants on ML-1M and Amazon-Book (AUC/UAUC).

of MF in GraphLoRA: it anchors user/item identities in the prompt (via input projection) and provides initial states for graph aggregation. Specifically, (1) *LoRA-only (Frozen MF)* freezes MF embeddings and disables the graph-injection path; (2) *LoRA-only (Trainable MF)* jointly optimizes MF with the LLM, and uses the injection path *without* graph aggregation (identity mapping); (3) *GraphLoRA* is the full model with a graph encoder integrated into the LoRA bottleneck. As shown in Table 5(A), freezing MF yields the weakest performance. Making MF trainable improves the results, indicating that updating collaborative representations under the recommendation objective is beneficial. Building on this stronger starting point, the full GraphLoRA further boosts performance by enabling topology-aware aggregation and injecting the refined structural signal inside the LoRA bottleneck.

Graph Encoder Variants. We vary the graph encoder inside the same injection mechanism (Table 5(B)). Among GCN, LightGCN, and NGCF, NGCF achieves the best results in our setting, suggesting that richer interaction modeling can be beneficial for producing the injected structural signal.

Method	Params	Tr.	Inf.	AUC	UAUC
(A) Baselines & Overall					
LoRA-only ($r = 8, \{q, v\}$)	1.000	1.000	1.000	0.7178	0.6952
CoRA-MF ($r = 16, \{q, k, v, o\}$)	7.635	1.086	1.009	0.7361	0.6884
GraphLoRA (Ours) ($r = 8, \{q, v\}$)	**1.017**	**1.029**	**1.014**	**0.7472**	**0.7102**
(B) Injection Position Ablation					
Pre-A (GNN before A)	5.224	1.058	1.033	0.7333	0.6941
Middle (Ours) ($A \rightarrow$ GNN $\rightarrow B$)	**1.017**	**1.029**	**1.014**	**0.7472**	**0.7102**

Table 6: Efficiency and injection-position analysis on ML-1M. Params/Tr./Inf. are overhead ratios (Base = 1.00) for trainable parameters, training time, and inference time, respectively.

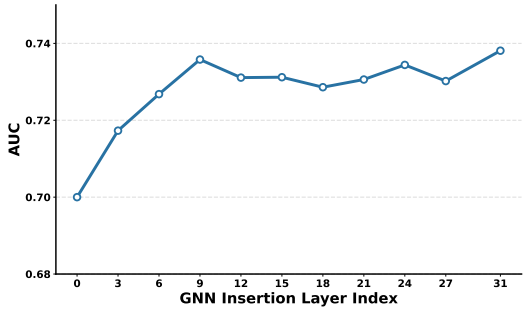

Figure 4: Effect of the GNN insertion layer on ML-1M (validation AUC).

Efficiency and Injection Position. Table 6 reports overheads normalized by the LoRA-only baseline (with trainable MF). GraphLoRA achieves superior performance with negligible costs (1.017× parameters, $\sim 1\%$ latency), significantly outperforming the heavy-budget CoRA-MF (7.635×). Regarding injection position, the *Pre-A* strategy operates in the high-dimensional space (d_{model}), causing parameter explosion (5.224×). In contrast, our **Middle** strategy targets the low-rank bottleneck ($r \ll d_{model}$). This design offers a **highly efficient structure-aware** alternative: by fusing signals within the task-specific bottleneck, it enables the graph topology to guide semantic adaptation compactly, fostering deep structure–semantic synergy within a constrained manifold.

Injection Layer Depth. We study how injection depth affects GraphLoRA by inserting the GNN module at different Transformer layers. On ML-1M, we sweep the insertion layer index and report validation AUC (Figure 4). Deeper-layer injection is generally more effective, with the best performance observed near the top layers (e.g., layer

Few-shot	SASRec	GRU-BERT	TALLRec	GraphLoRA
16	49.48	50.07	56.36	**60.90**
64	50.06	49.64	60.39	**61.08**
256	50.20	49.79	64.38	**67.90**

Table 7: Few-shot AUC on Book-Crossing under the TALLRec evaluation protocol

31 for ML-1M and layer 15 for Amazon-Book). This is consistent with recent interpretability research (Jin et al., 2025; Skean et al., 2024), which demonstrates that LLMs process information hierarchically: shallower layers primarily handle surface-level syntax, while deeper intermediate layers encode more abstract, task-relevant semantics optimal for aligning topology-aware collaborative signals.

5.4 Generalization Analysis

Few-Shot Performance and Backbone Robustness. To verify GraphLoRA robustness and generalization across architectures, we adopt the few-shot protocol from TALLRec (Bao et al., 2023) and switch the backbone from Vicuna to **LLaMA-7B**. Experiments are conducted on the Book-Crossing (Ziegler, 2005) dataset with 16, 64, and 256 training samples. As shown in Table 7, GraphLoRA consistently outperforms baselines across all regimes. This confirms that our topology-aware injection is not backbone-specific and effectively complements LLM reasoning even when explicit supervision is scarce.

6 Conclusion

In this study, we present GraphLoRA, a structure-aware framework that bridges collaborative signals and textual semantics within LLM. By embedding a graph message-passing module into the LoRA adaptation pathway, GraphLoRA generalizes traditional low-rank adaptation to structure-aware propagation, enabling joint reasoning over graph topology and textual semantics. Through unified optimization, it distills neighborhood-aware structural signals without disrupting linguistic competence. Extensive experiments across multiple datasets and evaluation settings demonstrate the effectiveness and efficiency of GraphLoRA, highlighting the potential of structure-aware parameter adaptation for future LLM-based recommendation research.

7 Limitations

This work provides an initial step towards structure-aware parameter-efficient adaptation for LLM-based recommendation. There remain several directions that are not fully explored in this paper. First, while we evaluate on representative benchmarks and backbones, validating GraphLoRA on a broader range of LLMs, domains, and recommendation tasks would further strengthen the generality of our conclusions. Second, our current study focuses on a targeted injection design; more systematic analysis of architectural choices (e.g., injection depth and module variants) across settings is left for future work. Finally, incorporating additional evaluation dimensions (e.g., robustness under different data distributions or efficiency under larger-scale deployments) would provide a more comprehensive understanding of the method's behavior.

8 Ethics Statement

Data Privacy. We utilize public, anonymized benchmarks devoid of Personally Identifiable Information (PII), strictly complying with their respective licenses. No private user data was collected or processed.

Biases and Societal Impact. While our work focuses on structural alignment, LLM-based recommenders inherently risk amplifying biases or creating echo chambers. Real-world deployments require rigorous fairness and safety evaluations to mitigate potential discrimination.

Environmental Impact. GraphLoRA promotes "Green AI." By freezing the LLM backbone and tuning only a minimal parameter set, our approach significantly reduces energy consumption and carbon footprint compared to full-model fine-tuning, supporting sustainable research.

9 Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.62206004, No.62572002, No.62272001, No.624065095), and the Natural Science Foundation of Anhui Province (No.2208085QF199, No.2508085MF159, No.2308085MF213).

References

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. TALLRec: An effective and efficient tuning framework to align large

language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, RecSys '23, pages 1007–1014. Association for Computing Machinery.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, pages 1126–1132.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. *Preprint*, arXiv:2301.00234.

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

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

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30.

F. Maxwell Harper and Joseph A. Konstan. 2015. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):1–19.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 507–517.

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)*, pages 639–648.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning (ICML)*, pages 2790–2799. PMLR.

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 *International Conference on Learning Representations (ICLR)*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2024. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*.

Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. 2025. Exploring concept depth: How large language models acquire knowledge and concept at different layers? In *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*, pages 558–573. Association for Computational Linguistics.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

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. 2023. Personalized prompt learning for explainable recommendation. *ACM Transactions on Information Systems*, 41(4):1–26.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4582–4597.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He.

2024. LLaRA: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1785–1795.

Qidong Liu, Xian Wu, Wanyu Wang, Yejing Wang, Yuanshao Zhu, Xiangyu Zhao, Feng Tian, and Yefeng Zheng. 2025a. LLMEmb: Large language model can be a good embedding generator for sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pages 12183–12191.

Yiyu Liu, Qian Liu, Yu Tian, Changping Wang, Yanan Niu, Yang Song, and Chenliang Li. 2021. Concept-aware denoising graph neural network for micro-video recommendation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, pages 1099–1108.

Yuting Liu, Jinghao Zhang, Yizhou Dang, Yuliang Liang, Qiang Liu, Guibing Guo, Jianzhe Zhao, and Xingwei Wang. 2025b. Cora: Collaborative information perception by large language model's weights for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pages 12246–12254.

Lin Mu, Xiaoyu Wang, Li Ni, Yang Li, Zhize Wu, Peiquan Jin, and Yiwen Zhang. 2025. DenseLoRA: Dense low-rank adaptation of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 10198–10211. Association for Computational Linguistics.

Oscar Skean, Md Rifat Arefin, Yann LeCun, and Ravid Shwartz-Ziv. 2024. Does representation matter? exploring intermediate layers in large language models. *arXiv preprint arXiv:2412.09563*.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024a. GraphGPT: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 491–500.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. 2024b. HiGPT: Heterogeneous graph language model. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2842–2853.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 165–174.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: A survey. *ACM Computing Surveys*, 55(5):1–37.

Yang Zhang, Keqin Bao, Ming Yan, Wenjie Wang, Fuli Feng, and Xiangnan He. 2024. Text-like encoding of collaborative information in large language models for recommendation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 9061–9079.

Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2025. Collm: Integrating collaborative embeddings into large language models for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 37(5):2329–2340.

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)*, pages 1157–1166.

Haiteng Zhao, Shengchao Liu, Chang Ma, Hannan Xu, Jie Fu, Zhi-Hong Deng, Lingpeng Kong, and Qi Liu. 2023. Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 6485–6509.

Cai-Nicolas Ziegler. 2005. Improving recommendation lists through topic diversification. In *WWW'05: Proceedings of the 14th international conference on World Wide Web*, pages 22–32. ACM.