Advances in Engineering Innovation Vol.16 Issue 11 EWA Publishing
Available Online: 12 December 2025 DOI: 10.54254/2977-3903/2025.30633

CoLLM-FaissRet: integrating collaborative information as a modality with efficient faiss retrieval for recommendation

Chuhong Zheng

Jacobs School of Engineering, University of California, San Diego, USA

c5zheng@ucsd.edu

Abstract. Recently, the idea of implementing Large Language Models (LLMs) into Recommender Systems (LLMRec) has shown considerable promise, especially when it comes to cold-start user issues. However, the current warm-start paradigms based on the use of LLMs tend to be categorised into three specific types, each of which possesses a severe drawback. To start with, the overall recommendation accuracy declines with the application of generic LLMs without modification. Second, fine-tuning these models can cause them to overuse semantic signals at the expense of collaborative signals. Third, hybrid methods that aim to combine collaborative-filtering functions with LLM representations are often unable to combine the two modalities effectively. Considering these deficiencies, we introduce Collaborative Large Language Model - Faiss based Retrieval (CoLLM-FaissRet), where collaborative evidence is viewed as a separate modality and fed directly to the LLM context. By exploiting the Faiss library to improve the speed of retrieval, the scheme also reduces the computational overhead of earlier efforts, but applies to both large-scale cold-start and warm-start recommendation cases. Unlike previous work, our experiments are hyper-optimised and use new datasets. We show that finding the optimal learning rates to apply to different components of the learning process can greatly improve the prediction accuracy of the recommender system. In addition, our methodology performs strongly in a wide range of recommendation environments. The implementation of our method is publicly available at: https://github.com/ChuhongZheng/CoLLM-FaissRet [1].

Keywords: LLMRec, collaborative information, user-item interactions

1. Introduction

Large Language Models (LLMs) have demonstrated impressive capabilities, thus sparking interest in applying them to recommender systems and leading to the creation of a new paradigm - LLM as Recommender (LLMRec) [2].

The existing approaches to the LLM-based recommendations can be grouped into three different categories, each of which has certain limitations. Direct LLM application [3, 4] is marred by reduced accuracy due to the inherent mismatch between the general-purpose nature of the LLMs and the special needs of recommendation systems. Fine-tuned LLMs [2, 5, 6] are often overly sensitive to semantic data but do not sufficiently utilise collaborative information. Collaborative filtering integrated LLMs [7-10] face challenges regarding accuracy and computational efficiency, in part because of the lack of synergies and the huge computational and memory requirements of full candidate scoring.

To enable the application of Large Language Models (LLMs) with collaborative signals in a lightweight and flexible architecture, this paper proposes CoLLM-FaissRet, which conceptualises collaborative information as an independent modality and includes it in language models through direct correspondence with an existing conventional recommendation system. The proposed solution takes advantage of a Multilayer Perceptron (MLP) as the mapping mechanism and incorporates the Faiss library [11] to support the efficient search of vector similarities, thus being able to select the most pertinent candidate items in the process of inference.

The suggested framework successfully combines the strengths of traditional collaborative filtering models with the reasoning abilities of LLMs. The traditional models that adopt the historical interaction of users are generally robust in warm start conditions; meanwhile, LLMs are robust in semantic understanding, which is especially beneficial in cold start conditions when there is little history. To achieve a balance between these two complementary assets, we incorporate Faiss into the mapping module, thereby making similarity matching more accurate and top-K retrieval much faster. Thus, the framework proves to be more accurate in recommendations and more efficient, which makes it especially applicable in large-scale deployments, where performance and computational cost are crucial factors to take into account.

Copyright: © 2025 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

The main contributions of this work are summarised as follows:

• Novel Framework: This paper presents CoLLM-FaissRet, a more powerful conceptualisation of collaborative information as a modality, which can be smoothly integrated into large language models through direct mapping. Incorporating Faiss to enhance efficiency through efficient retrieval, the framework can enhance performance in a large-scale recommendation setting, both in cold-start and warm-start settings.

• Parameter Optimisation: The optimisation of CoLLM-FaissRet through comprehensive parameter optimisation greatly improved recommendation accuracy. The AUC and UAUC indicators increased to 0.7479 and 0.6999, respectively, on the ML-1M dataset, as opposed to the nominal values of 0.7295 and 0.6875, which highlights the significance of parameter optimisation in multimodal recommendation systems.

• Datasets Evaluation: Ample experimental analyses performed on three real-life datasets support the high effectiveness and robustness of CoLLM-FaissRet. For example, on the BookCrossing dataset, the suggested method gives an AUC curve of 0.6947 and UAUC of 0.6362, outperforming a number of competing baseline methods and confirming its usefulness in practice.

2. Related works

We consider related works in this section in three key categories: (1) direct LLM application for recommendation [3, 4], which uses the natural language capabilities of the LLM without parameter updates; (2) fine-tuned LLMs for recommendation [2, 5, 6], which apply fine-tuning procedures together with external tools to improve recommendation performance; and (3) collaborative filtering integrated LLMs for recommendation [7-10].

2.1. Direct LLM application for recommendation

Studies have investigated the use of In-Context Learning (ICL) in recommender systems, where the natural language abilities of LLMs were used, and no parameter updates were performed. As an example, Chat-Rec framework [3] uses the conversational architecture of ChatGPT to guide the creation of the recommendations, and other studies generate them by providing carefully designed ICL prompts [4].

Drawback: The limited precision of these approaches is caused by the fact that general-purpose LLMs do not align with the specialised needs of recommendation systems. As the general-purpose LLMs are not trained to model user-item interaction, they often underperform compared with special-purpose recommendation models, thus providing suboptimal accuracy in practical applications.

2.2. Fine-tuned LLMs for recommendation

To overcome the drawbacks of the direct use of LLMs, investigators have used instruction tuning, where Supervised Fine-Tuning (SFT) is run on recommendation-specific corpora to enhance the recommendation capabilities of the LLMs [2, 6]. Similarly, more interactive conversational recommendation systems are enabled by alternative methods that add external tools, like databases and APIs, to the LLMs, either via fine-tuning or advanced prompting [5].

Drawback: In spite of these attempts, there are still problems with accuracy. LLMs are often biased towards semantic information, and empirical experiments have shown that even fine-tuned large language models sometimes perform worse than traditional models [12]. The key reasons behind these limitations are that they are based on the use of pre-trained semantic priors (e.g. text similarity) and ignore collaborative cues that play a key role in effective collaborative filtering [13].

2.3. Collaborative filtering integrated LLMs for recommendation

In an attempt to address the biases that LLMs exhibit towards semantic priors, researchers have come up with several strategies that seek to incorporate collaborative cues into recommendation systems that run on the basis of LLMs. Some of the methodologies use an ensemble of the standard collaborative filtering model with the LLM through ensemble techniques [7], or they learn the ID-based collaborative embeddings in the very representation space of the LLM [10]. Moreover, there are a few lines of work that use LLMs as feature extractors to provide semantic attributes to collaborative models [8], and other works also present a mapping mechanism that converts pre-trained collaborative embeddings into the LLM token space [9].

Drawback: These methods of integration face challenges in accuracy and efficiency when attempting to merge the collaborative and semantic information. Simply combining LLMs with traditional models through an ensemble may assign shallow integration, thus failing to achieve effective synergy between the two segments. The independent learning of collaborative embeddings in the representation space of the LLM can cause distortion of inherent structural characteristics that are important in modelling interaction between users and items. Using the LLMs as feature extractors only underutilises their strong semantic reasoning abilities in the final recommendation procedure. Moreover, algorithms that project collaborative model embeddings onto the token space have strong efficiency issues: they require scoring candidate items at inference time, which has computational complexity that scales with the size of the catalogue; the exhaustive candidate space produces a lot of noise that may compromise semantic alignment; and finally, storing the full set of item embeddings in memory takes a significant amount of GPU memory, which may limit the scale of the model that can be deployed.

3. Methodology

3.1. Overview

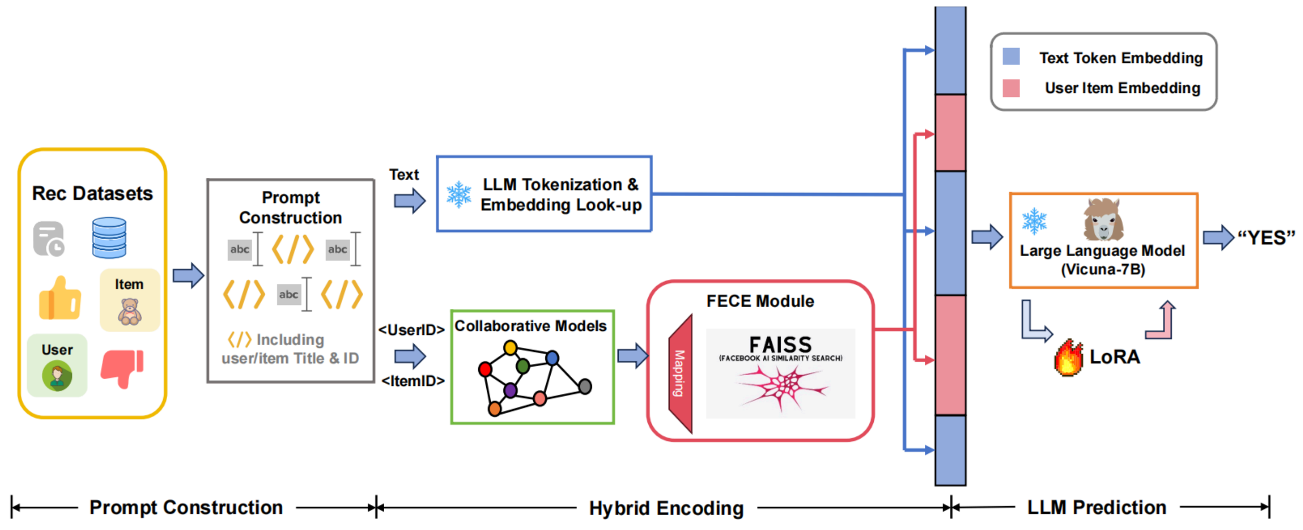


Figure 1. Overall architecture of the proposed method

The overall design of the proposed structure is composed of three sequentially structured elements, namely Prompt Construction (see Section 3.2), Hybrid Encoding (see Section 3.3), and LLM Prediction (see Section 3.4), depicted in Figure 1.

The working process starts with the conversion of the recommendation data into structured linguistic prompts through the module of prompt construction. These prompts are then sent to the hybrid encoding stage, where they are translated into latent representations that can be processed by the LLM. Finally, the encoded representations are passed to the LLM in order to produce recommendation predictions.

Our main invention is the successful merging of collaborative information to enhance the recommendation capabilities of LLMs and make them usable in both cold-start and warm-start situations. This innovation is achieved by certain design decisions in the first two elements. During the prompt construction stage, an item and user identification attribute are joined with textual descriptions to enumerate collaborative cues. During the hybrid encoding stage, the Faiss Enhanced Collaborative Encoding (FECE) module is used to decompose and project collaborative representations into the embedding space of the LLM. Faiss inference is also used to ensure scalability in computation when it comes to large-scale deployment.

3.2. Prompt construction

The recommendation dataset is taken as input in the prompt construction process, and structured prompts are created. Expanding the approach suggested in the methodology section of TALLRec [2], we use fixed templates to translate instances of recommendations into textual prompts. More specifically, the profiles of the users are coded in terms of titles of the items they have already engaged with, and candidate items are expressed in terms of their titles.

To incorporate the collaborative filtering signals, we define unique identifier fields which are semantically neutral placeholders to capture collaborative information. The template of the entire prompt is shown in Figure 2.

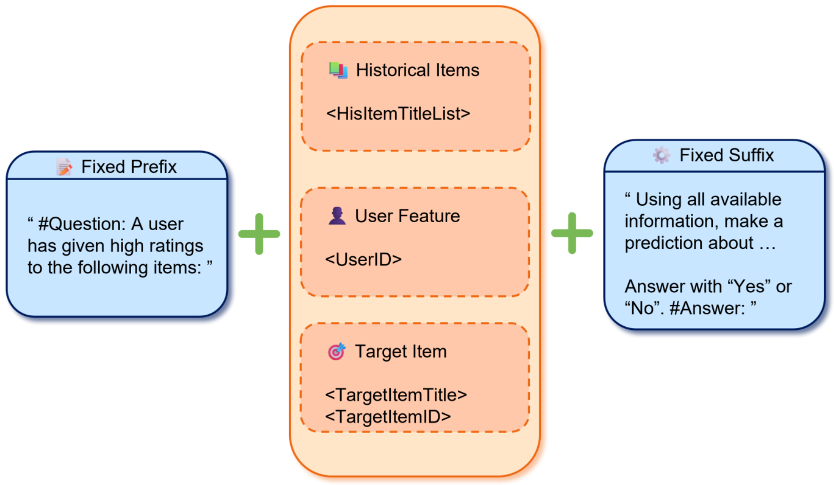

Fixed Prefix
" #Question: A user has given high ratings to the following items: "
Historical Items
<HisItemTitleList>
User Feature
<UserID>
Target Item
<TargetItemTitle>
<TargetItemID>
Fixed Suffix
" Using all available information, make a prediction about …
Answer with "Yes" or "No". #Answer: "

Figure 2. Prompt construction

In this template, the placeholder <HisItemTitleList> refers to an enumeration of the item titles in chronological order based on the interaction experience of the user and, therefore, provides clear indicators of preference. The name <TargetItemTitle> refers to the title of the currently being evaluated candidate item. The placeholders <UserID> and <TargetItemID> serve as collaborative information injectors where user and item identifiers are placed as unique traits, even though they lack semantic value. This design maintains textual coherence but enables the model to take advantage of identity-based collaborative cues through the characteristics denoted by the underlined descriptors.

For each recommendation instance, the four template fields are dynamically filled with the values of the dataset corresponding to each recommendation instance to create instance-specific prompts. It is also worth noting that at the pre-training stages, we use a different template configuration that does not use identity-related fields because preference modelling is based only on textual content.

3.3. Hybrid encoding

The hybrid encoding component converts input prompts into latent representations that can be processed by LLMs. As shown in Figure 1, the method uses two separate encoding paths to process different kinds of data. For text, the model relies on the LLM's built-in abilities to tokenise and embed the input. For structured identifiers like <UserID> and <TargetItemID>, it uses a dedicated FECE Module to extract and encode collaborative signals for the LLM.

Formally, given a prompt corresponding to sample $(u, i, y) \in D$, we first tokenise its textual content using the LLM Tokeniser, resulting in a token sequence $P = [t_1, t_2, ..., t_k, u, t_{k+1}, ..., i, ..., t_K]$, where t_k represents a text token and $\$u/i$ denotes user/item identifiers within their respective fields. This sequence is subsequently encoded into embedding representations:

$$E = [e_{t_1}, ..., e_{t_k}, e_u, e_{t_{k+1}}, ..., e_i, ..., e_{t_K}] \tag{1}$$

where $e_{t_k} \in R^{1 \times d_2}$ signifies the token embedding for t_k obtained via the LLM's embedding layer, i.e., $e_{t_k} = Embedding_{LLM}(t_k)$; whereas $e_u/e_i \in R^{1 \times d_2}$ represents collaborative embeddings for user u and item i, generated through our collaborative encoding module.

FECE Module: This module comprises a conventional collaborative model $f_\psi(\cdot)$ and a mapping layer $g_\phi(\cdot)$ parameterised by ϕ. Given user u and item i, the collaborative model generates preliminary representations $u = f_\psi(u; D)$ and $i = f_\psi(i; D)$ encapsulating collaborative information. The mapping layer then projects these representations into the LLM's embedding space:

$$e_u = g_\phi(u), u = f_\psi(u; D) \tag{2}$$

$$e_i = g_\phi(i), i = f_\psi(i; D) \tag{3}$$

where $u, i \in R^{1 \times d_1}$ denote user and item representations from f_ψ, and the mapping layer g_ϕ is implemented as a MLP with input dimension d_1 and output dimension d_2 (typically $d_1 < d_2$).

During training, the module does not use Faiss integration to ensure the best accuracy and to allow proper gradient propagation for learning good collaborative representations. However, during inference, we use Faiss for fast approximate nearest neighbour search, greatly speeding up retrieval in large-scale situations where handling the entire item catalogue is too demanding. This approach strikes a good balance between training accuracy and inference efficiency.

3.4. LLM prediction

After converting the input prompt into an embedding sequence E (as shown in Equation 1), the LLM uses these representations to produce predictions. Because general-purpose LLMs are not specifically trained for recommendation tasks, we enhance the base model using a Low-Rank Adaptation (LoRA) component [14] to support recommendation-oriented predictions, as depicted in Figure 1. This enhancement works by adding pairs of rank-decomposition matrices to the original LLM parameters via a plugged-in mechanism, allowing the model to adapt effectively to the new tasks (recommendation) with only a minimal increase in parameters. The prediction process is formally defined as:

$$\hat{y} = h_{\hat{\Theta}+\Theta'}(E) \tag{4}$$

As Θ denotes the fixed weights of the pre-trained language model $h(\cdot)$, and Θ' indicates the parameters of Low-Rank Adaptation (LoRA) that are to be trained explicitly as part of the recommendation task. The model variable, $\hat{y}$, represents the expected probability of a positive interaction, i.e. the probability that the language model will respond to a recommendation prompt with a "Yes". The LoRA implementation ensures parameter-efficient learning by changing only the adapter weights, as the task-specific optimisation is performed.

The two-stage tuning strategy that we follow in this research is: 1) optimisation of the FECE module to collaborate with recommendation tasks and 2) fine-tuning the LoRA module to collaborate with information extraction. This sequential method guarantees stable incorporation of knowledge in both cold-start and warm-start settings.

4. Experiments

4.1. Experimental settings

Experiments were conducted on a Linux-based server with Ubuntu 20.04, with Miniconda3 as environment management. The hardware setup included a 24GB VRAM (per GPU) dual-GPU configuration and a 32-vCPU processor, and CUDA 11.6 was used to speed up computations in the GPUs. An experimental protocol similar to the configurations of CoLLM was used [9].

4.1.1. Datasets

Our methodology was empirically evaluated using three publicly available real-world datasets: ML-1M, Amazon-Book, and BookCrossing.

ML-1M [15] is a widely-used movie recommendation dataset containing approximately 1 million ratings collected between 2000 and 2003. Amazon-Book [16] is derived from the Amazon Product Review dataset's book category, containing user reviews from 1996 to 2018. BookCrossing [17] is a book recommendation dataset containing both numerical ratings (1-10 scale) and textual metadata, including book authors and titles. We follow the settings proposed by Zhang et al. [9] for ML-1M and Amazon-Book, and those by Bao et al. [2] for BookCrossing.

4.1.2. Algorithms

We evaluate the following methods in our experiments:

• CoLLM-FaissRet (Ours): Our proposed approach treats collaborative information as a separate modality and integrates it into LLMs through direct mapping, incorporating Faiss for efficient retrieval to better handle large-scale cold-start and warm-start recommendation scenarios.

• MF [18]: A widely-used latent factor-based collaborative filtering technique based on Matrix Factorisation.

• CTRL (DIN) [8]: An approach that combines language and collaborative models based on knowledge distillation, with DIN [19] as the collaborative model.

• TALLRec [2]: An LLM-based recommendation method that aligns large language models with recommendation tasks via instruction tuning, implemented using the Vicuna-7B model.

4.1.3. Experimental factors

The following key factors are the focus of the experimental design:

• Embedding dimensions: $\{64, 128, 256\}$.

• LoRA fine-tuning learning rate: $\{1 \times 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}, 7 \times 10^{-4}, 1 \times 10^{-3}\}$.

• FECE module learning rate: $\{1 \times 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}, 7 \times 10^{-4}, 1 \times 10^{-3}\}$.

• Robustness validation: additional experiments on the BookCrossing dataset.

4.1.4. Measurement

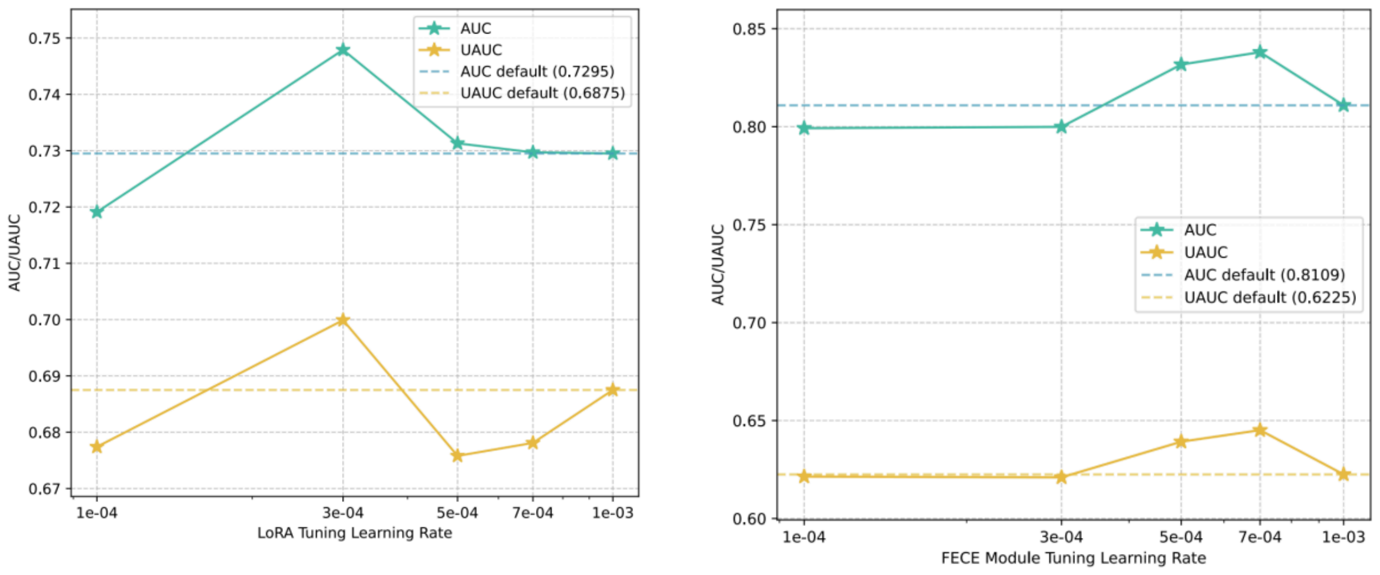
To evaluate the performance of the studied methods, we employ two widely used metrics for explicit recommendation:

• Area Under the ROC Curve (AUC): This is a measure of overall ranking accuracy that determines the area under the receiver operating characteristic curve.

• User-based AUC (UAUC): This metric is calculated by first computing AUC scores individually for each user based on their exposed items, then averaging these values across all users.

4.2. Results

4.2.1. Parameter optimisation



(a) AUC and UAUC on ML-1M as the LoRA tuning rate is swept from 1×10^{-4} to 1×10^{-3}

(b) AUC and UAUC on Amazon-Book as the FECE module learning rate is varied over the same range

Figure 3. Impact of learning rate on CoLLM-FaissRet performance, all other parameters were fixed

Our parameter optimisation methodology employs a two-stage strategy: initially optimising the LoRA module for recommendation task adaptation, followed by FECE module fine-tuning for collaborative information extraction. We perform extensive parameter searches on the ML-1M and Amazon-Book datasets, examining embedding dimensions of $\{64, 128, 256\}$. For learning rates, we conduct fine-grained adjustments. For LoRA fine-tuning, we explore rates in $\{1 \times 10^{-4}$, $3 \times 10^{-4}, 5 \times 10^{-4}, 7 \times 10^{-4}, 1 \times 10^{-3}\}$; similarly, for the FECE module, we vary the learning rate over the same range. Because of limited space, Figure 3 focuses on two key learning rates: the LoRA fine-tuning rate on ML-1M (a) and the FECE module learning rate on Amazon-Book (b). Full results from other parameter optimisation tests will be provided in the technical report [1]. We did not test larger learning rates beyond 1×10^{-3} because initial experiments showed that higher values, such as 5×10^{-3}, led to unstable training and a clear drop in performance. Similarly, the embedding size was kept to a maximum of 256, limited by our GPU memory, since larger dimensions would have required much more computing power than our hardware can support.

The experimental findings demonstrate that the CoLLM-FaissRet framework has strong performance across diverse datasets. Learning rates can be improved by careful optimisation, which can significantly enhance model accuracy, as shown in Figure 3. Specifically, the AUC for the LoRA tuning rate of 3×10^{-4} on the ML-1M data is 0.7479, and the UAUC is 0.6999. Similarly, an optimised FECE module learning rate of 7×10^{-4} leads to an improved performance with an AUC of 0.8380 and UAUC of 0.6451. These results validate that the learning rates of individual components of the framework can be configured to significantly improve recommendation accuracy.

4.2.2. Datasets evaluation

To determine the soundness of the model, we used the BookCrossing dataset in further experiments. The BookCrossing dataset, as cited in [17], consists of book ratings on a 1-10 scale as well as textual information, including author and title. In order to undergo binary classification, the ratings were binarised with a threshold of 5, thus transforming them into binary categories. In each user case, we randomly chose one item that the user had interacted with as the prediction target and ten historical interactions to act as contextual data. The data were randomly divided into training, validation and testing sets in a ratio of 8:1:1 to make the model performance evaluation fair.

A detailed comparison against other algorithms is given in Table 1, where CoLLM-FaissRet is seen to have competitive performance in all cases. It is important to note that its strength on the BookCrossing data is clearly demonstrated, as the proposed approach achieves an AUC of 0.6947 and UAUC of 0.6362, thereby outperforming several robust baselines. These findings support the generalisation of the approach to different recommendation contexts.

Table 1. Performance comparison between algorithms on three real-world datasets (best in bold, second best underlined)

Method	ML-1M		Amazon-Book		BookCrossing	
	AUC	UAUC	AUC	UAUC	AUC	UAUC
MF	0.6482	0.6361	0.7134	0.5565	0.6173	0.5478
CTRL	0.7159	0.6492	0.8202	0.5996	0.7129	0.5923
TallRec	0.7097	0.6818	0.7375	0.5983	0.6347	0.6017
CoLLM-FaissRet	0.7295	0.6875	0.8109	0.6225	0.6947	0.6362

4.2.3. Summary

The CoLLM-FaissRet framework achieves significant improvements in accuracy and robustness, with parameter optimisation showing AUC gains of 2.52% for LoRA and 3.34% for FECE learning rate, and UAUC gains of 1.80 and 3.63%, respectively. On the BookCrossing dataset, CoLLM-FaissRet's AUC is competitive with the best method, while its UAUC surpasses the second-best by 5.73%.

5. Conclusion

This paper presents CoLLM-FaissRet, a novel framework that theorises collaborative information as an independent modality within the LLM-based recommendation framework. The proposed methodology combines direct mapping techniques with Faiss-based vector retrieval, which enables efficient information retrieval and, therefore, deals with cold-start and warm-start recommendation states with a greater degree of efficiency.

Future studies will focus on alternative LLMs and determine the improvement in inferential efficiency of CoLLM-FaissRet to larger real-world datasets. These studies will give additional support for the feasibility of the method in production settings.

References

[1] Zheng, C. (2025). *CoLLM-FaissRet: Integrating collaborative information as a modality with efficient Faiss retrieval for recommendation* [Technical report]. GitHub. Retrieved May 27, 2024, from https: //github.com/ChuhongZheng/CoLLM-FaissRet/blob/main/CoLLM-FaissRet%20(technical%20report).pdf

[2] Bao, K., Zhang, J., Zhang, Y., Wang, W., Feng, F., & He, X. (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* (pp. 1007–1014). https: //doi.org/10.1145/3604915.3608847

[3] Gao, Y., Sheng, T., Xiang, Y., Xiong, Y., Wang, H., & Zhang, J. (2023). *Chat-rec: Towards interactive and explainable LLMs-augmented recommender system*. arXiv. https: //arxiv.org/abs/2303.14524

[4] Liu, J., Liu, C., Zhou, P., Lv, R., Zhou, K., & Zhang, Y. (2023). *Is ChatGPT a good recommender? A preliminary study*. arXiv. https: //arxiv.org/abs/2304.10149

[5] Huang, X., Lian, J., Lei, Y., Yao, J., Lian, D., & Xie, X. (2025). Recommender ai agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems, 43*(4), Article 103. https: //doi.org/10.1145/3702593

[6] Zhang, J., Xie, R., Hou, Y., Zhao, X., Lin, L., & Wen, J.-R. (2025). Recommendation as instruction following: A large language model empowered recommendation approach. *ACM Transactions on Information Systems, 43*(5), Article 115. https: //doi.org/10.1145/3707204

[7] Bao, K., Zhang, J., Wang, W., Zhang, Y., Yang, Z., Luo, Y., Chen, C., Feng, F., & Tian, Q. (2025). A bi-step grounding paradigm for large language models in recommendation systems. *ACM Transactions on Recommender Systems, 3*(4), Article 45. https: //doi.org/10.1145/3713192

[8] Li, X., Chen, B., Hou, L., & Tang, R. (2023). Ctrl: Connect collaborative and language model for ctr prediction. *ACM Transactions on Recommender Systems, 1*(3), Article 24. https: //doi.org/10.1145/3632412

[9] Zhang, Y., Feng, F., Zhang, J., Bao, K., Wang, Q., & He, X. (2025). CoLLM: Integrating collaborative embeddings into large language models for recommendation. *IEEE Transactions on Knowledge and Data Engineering*. Advance online publication. https: //doi.org/10.1109/TKDE.2025.3511865

[10] Zheng, B., Hou, Y., Lu, H., Chen, Y., Zhao, W. X., Chen, M., & Wen, J.-R. (2024). Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering* (ICDE) (pp. 1435–1448). IEEE. https: //doi.org/10.1109/ICDE60146.2024.00110

[11] Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., & Jégou, H. (2024). *The Faiss library*. arXiv. https: //arxiv.org/abs/2401.08281

[12] Lin, J., Shan, R., Zhu, C., Du, K., Chen, B., Quan, S., Tang, R., Yu, Y., & Zhang, W. (2024). Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024* (pp. 3497–3508). https: //doi.org/10.1145/3589334.3645399

[13] Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., Weng, T., Savarese, R., & Bosma, M. (2023). *Larger language models do in-context learning differently*. arXiv. https: //arxiv.org/abs/2303.03846

[14] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations* (ICLR 2022). https: //openreview.net/forum?id=nZeVKeeFYf9

[15] Harper, F. M., & Konstan, J. A. (2015). The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems, 5*(4), Article 19. https: //doi.org/10.1145/2827872

[16] He, R., & McAuley, J. (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* (pp. 507–517). https: //doi.org/10.1145/2872427.2883037

[17] Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web* (pp. 22–32). https: //doi.org/10.1145/1060745.1060754

[18] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30–37. https: //doi.org/10.1109/MC.2009.263

[19] Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., & Gai, K. (2018). Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1059–1068). https: //doi.org/10.1145/3219819.3219823