EasyRec: Simple yet Effective Language Models for Recommendation

Xubin Ren
The University of Hong Kong
xubinrencs@gmail.com

Chao Huang[*]
The University of Hong Kong
chaohuang75@gmail.com

Abstract

Deep neural networks have emerged as a powerful technique for learning representations from user-item interaction data in collaborative filtering (CF) for recommender systems. However, many existing methods heavily rely on unique user and item IDs, which restricts their performance in zero-shot learning scenarios. Inspired by the success of language models (LMs) and their robust generalization capabilities, we pose the question: How can we leverage language models to enhance recommender systems? We propose EasyRec, an effective approach that integrates text-based semantic understanding with collaborative signals. EasyRec employs a text-behavior alignment framework that combines contrastive learning with collaborative language model tuning. This ensures strong alignment between text-enhanced semantic representations and collaborative behavior information. Extensive evaluations across diverse datasets show EasyRec significantly outperforms state-of-the-art models, particularly in text-based zero-shot recommendation. EasyRec functions as a plug-and-play component that integrates seamlessly into collaborative filtering frameworks. This empowers existing systems with improved performance and adaptability to user preferences. Implementation codes are publicly available at: https://github.com/HKUDS/EasyRec.

1 Introduction

Deep learning has established itself as a highly promising solution for capturing user preferences in online recommender systems (Zhang et al., 2019; Yang et al., 2022; Zhang et al., 2022). This approach harnesses deep neural networks to learn rich user and item representations by analyzing complex user-item interaction patterns. Consequently, recommendation algorithms can accurately infer user preferences and deliver personalized recommendations (Xu et al., 2023; Sun et al., 2021).

[*]Chao Huang is the Corresponding Author.

Recent advancements in enhancing recommender systems through neural network-powered collaborative filtering frameworks, particularly graph neural networks (GNNs) (Wang et al., 2019; He et al., 2020; Xia et al., 2022), have leveraged the inherent graph structure in data to capture high-order relationships among users and items. Notable GNN-based approaches, such as NGCF (Wang et al., 2019) and LightGCN (He et al., 2020), demonstrate impressive performance via recursive message passing mechanisms. However, data scarcity remains a significant challenge, hindering deep collaborative filtering models from accurately learning user/item representations, especially with sparse interaction data (Lin et al., 2021; Wei et al., 2021; Hao et al., 2021). To address this, recent studies have explored self-supervised learning for effective data augmentation. For instance, contrastive methods like SGL (Wu et al., 2021) and NCL (Lin et al., 2022) utilize graph contrastive learning, while generative approaches like AutoCF (Xia et al., 2023) employ masked autoencoding to reconstruct interaction structures.

Recent advances in self-supervised learning show promise in alleviating data scarcity in collaborative filtering models. However, these methods encounter a significant limitation (Yuan et al., 2023). They rely heavily on unique identities (IDs) to represent users and items. This reliance presents major challenges for practical recommenders that must handle data from diverse domains or time periods effectively. Existing ID-based models struggle to adapt to changes in user and item identity tokens. This problem becomes particularly severe in **zero-shot recommendation** scenarios (Ding et al., 2021; Hou et al., 2022). In these cases, training data lacks overlap with deployment data regarding users and items. This limitation hinders their ability to generalize effectively towards foundational recommender systems. Cross-domain recommendation methods (Xie et al., 2022; Cao et al., 2022) attempt to leverage knowledge across multiple domains. However, they often make restrictive assumptions about user overlap. These approaches assume that users from different domains belong to the same set (Dacrema et al., 2012; Xie et al., 2022). This assumption significantly limits their flexibility and generalization capabilities. Consequently, existing methods struggle to provide accurate recommendations to diverse user populations. They face particular challenges when operating across varying domains and contexts.

Language Models as Zero-Shot Recommenders. The challenges discussed earlier highlight a critical need in recommendation systems. This study aims to introduce a recommender system that functions as a zero-shot learner. The system should possess robust generalization capabilities and adapt to new recommendation data seamlessly. To accomplish this objective, we propose integrating language models with collaborative relation modeling. This integration forms an effective text embedder-EasyRec that is both lightweight and effective. Our approach seamlessly combines text-based semantic encoding with high-order collaborative signals. This combination results in a recommender system with strong generalization ability. The system leverages rich semantic understanding while capturing collaborative patterns from user-item interactions.

Recent research has explored leveraging large language models (LLMs) to enhance recommender systems. Existing approaches broadly fall into two categories. The first category uses LLMs for data augmentation (*e.g.*, RLMRec (Ren et al., 2024), AlterRec (Li et al., 2024)). These methods encode textual information to complement collaborative filtering. While this combines LLM and collaborative strengths, these methods remain ID-based and struggle to generalize. The second approach utilizes LLMs to directly generate user-item interaction predictions (*e.g.*, LLaRA (Liao et al., 2024), CoLLM (Zhang et al., 2023)). However, such LLM-based recommenders suffer from poor efficiency, requiring approximately one second per prediction. This renders them impractical for large-scale recommendation tasks. These challenges highlight the need for efficient, scalable solutions that integrate semantic understanding with collaborative strengths for zero-shot recommendation.

Our model demonstrates superior performance compared to state-of-the-art language models, as illustrated in Figure 1. This performance advan-

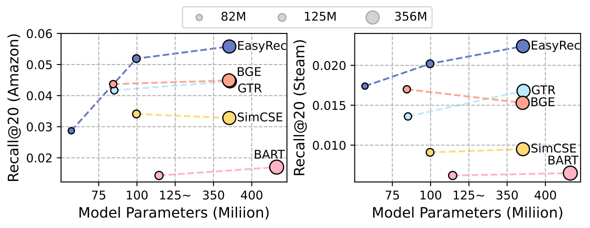
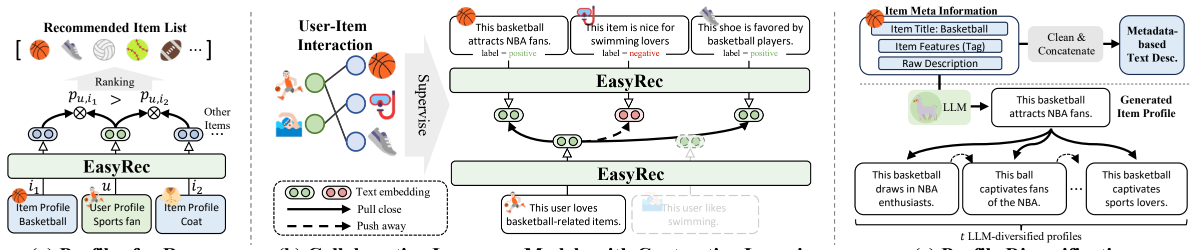


Figure 1: EasyRec outperforms state-of-the-art language models in text-based zero-shot recommendation.

tage is achieved within a cost-efficient parameter space of 100 to 400 million parameters. The computational cost is approximately 0.01 seconds per prediction. Notably, our model exhibits the scaling law phenomenon. Performance continually improves as parameter size increases. This contrasts with existing approaches that suffer from poor efficiency. Our model is designed to be highly scalable and practical for large-scale recommendation tasks. The computational efficiency represents significant advancement over current methods. While existing LLM-based recommenders require substantial inference time, our approach maintains both accuracy and speed for real-world deployment. In summary, this work makes the following contributions:

- **Motivation**. The primary objective of this study is to introduce a novel recommender system built upon language models. This system functions as a zero-shot learner with exceptional adaptability to diverse recommendation data. The new EasyRec exhibits robust generalization capabilities across different domains and contexts.
- **Methodology**. We propose a novel contrastive learning-powered collaborative language modeling approach. This method aligns text-based semantic encoding with collaborative signals from user behavior. The system captures both semantic representations of users and items. It also learns underlying behavioral patterns and interactions within the recommendation data.
- **Zero-Shot Recommendation Capacity**. The EasyRec is extensively evaluated through rigorous experiments as a text-based zero-shot recommender system. Performance comparisons reveal consistent and significant advantages over baseline methods. The model excels in both recommendation accuracy and generalization capabilities. Furthermore, the study demonstrates remarkable potential in adapting dynamic user profiles. These profiles are highly adaptive to time-evolving user preferences.
- **Existing Recommender Enhancement**. Our


Recommended Item List
[...]
Ranking
p_{u,i_1} > p_{u,i_2}
Other Items ...
EasyRec
i_1
u
i_2
Item Profile Basketball
User Profile Sports fan
Item Profile Coat
User-Item Interaction
Supervise
Text embedding
Pull close
Push away
This basketball attracts NBA fans.
label = positive
This item is nice for swimming lovers
label = negative
This shoe is favored by basketball players.
label = positive
EasyRec
EasyRec
This user loves basketball-related items.
This user likes swimming.
Item Meta Information
Item Title: Basketball
Item Features (Tag)
Raw Description
Clean & Concatenate
Metadata-based Text Desc.
LLM
This basketball attracts NBA fans.
Generated Item Profile
This basketball draws in NBA enthusiasts.
This ball captivates fans of the NBA.
...
This basketball captivates sports lovers.
t LLM-diversified profiles

(a) Profiles for Rec. **(b) Collaborative Language Models with Contrastive Learning** **(c) Profile Diversification**

Figure 2: The overall framework of our proposed collaborative information-guided language model EasyRec.

proposed EasyRec integrates seamlessly as a lightweight, plug-and-play component with state-of-the-art collaborative filtering models. The lightweight and modular design represents a key strength of our approach. This design facilitates adoption of our novel recommendation paradigm across diverse use cases. The integration enhances existing recommendation methods without significant computational overhead.

2 Preliminaries

In recommender systems, we have a set of users $\mathcal{U}$ and items $\mathcal{I}$, along with their interactions (e.g., clicks, purchases). For each user $u \in \mathcal{U}$, let $\mathcal{N}_u$ be the set of items interacted with. For each item $i \in \mathcal{I}$, let $\mathcal{N}_i$ be the set of users who have interacted with it. These interactions can be represented by an interaction matrix $\mathcal{A}^{|\mathcal{U}|\times|\mathcal{I}|}$. Here, $\mathcal{A}^{u,i}$ is 1 if user u has interacted with item i, and 0 otherwise. The goal of recommendation is to predict the preference score $p_{u,i}$ of future interactions between user u and item i. This score can be used to generate recommendations based on individual preferences.

Text-based Zero-Shot Recommendation is crucial in recommender systems. It serves as a pathway toward foundation models. Textual descriptions, such as product titles and user profiles, provide valuable information. These descriptions enable effective recommendations across various datasets. This approach overcomes the limitations of traditional collaborative filtering methods. It offers significant advantage in generalization over the ID-based paradigm. This capability extends to scenarios where specific user-item interactions have not been previously encountered.

Formally, we define $\mathcal{P}_u$ and $\mathcal{P}_i$ as the generated text-based profiles of user u and item i, respectively. These profiles are then encoded into representations $\mathbf{e}_u$ and $\mathbf{e}_i$ using a language model (LM). This process is shown as:

$$\mathbf{e}_u = \mathrm{LM}(\mathcal{P}_u), \quad \mathbf{e}_i = \mathrm{LM}(\mathcal{P}_i). \tag{1}$$

The preference score $p_{u,i}$ between user u and item i is calculated as the cosine similarity between their text embeddings $\mathbf{e}_u$ and $\mathbf{e}i$. This is expressed as $p_{u,i} = \cos(\mathbf{e}_u, \mathbf{e}_i)$. We then recommend the top-k unclicked items with highest similarity scores, resulting in a recommendation set.

$$\mathcal{R}_u = \text{top-k}_{i\in\mathcal{I}\setminus\mathcal{N}_u} cos(\mathbf{e}_u, \mathbf{e}_i). \tag{2}$$

Text-enhanced Collaborative Filtering. Collaborative filtering (CF) is a widely used recommendation paradigm. It leverages the collaborative relationships among users and items. This existing CF paradigm can be enhanced by integrating encoded semantic representations. Typically, the value $p_{u,i}$ is calculated based on the interaction data. This is expressed as $p_{u,i} = f(u, i, \mathcal{A})$, where $\mathcal{A}$ represents the interaction data. Text-enhanced collaborative filtering builds upon this foundation. It incorporates textual features $\mathbf{e}$ encoded by language models as supplementary representations. This integration aims to improve the recommendation performance of traditional ID-based frameworks.

$$p_{u,i} = f(u, i, \mathcal{A}, \mathbf{e}_u, \mathbf{e}_i). \tag{3}$$

3 Methodology

In this section, we first discuss how we gather textual profiles for users and items. Next, we dive into the specifics of EasyRec and its training approach. Lastly, we introduce our method for diversifying user profiles. This method improves the model's ability to adapt to various situations.

3.1 Collaborative User and Item Profiling

In real-world recommenders, only raw text data may be available. This includes item titles and categories. Privacy concerns limit comprehensive user information. Directly using this textual data can overlook essential collaborative relationships. These relationships are needed for effective user behavior modeling. To address these issues, we propose generating textual profiles using large language models. Examples include GPT and LLaMA series (Ren et al., 2024). These models incorporate collaborative information.

3.1.1 Item Profiling

Given raw item information, we aim to generate a comprehensive item profile $\mathcal{P}_i$. This information includes title h_i, categories c_i, and description d_i (e.g., book summary). The profile captures both semantic and collaborative aspects. To reflect user-item interactions, we incorporate textual information. This includes user reviews $r_{u,i}$. The item profile generation process is:

$$\mathcal{P}_i = \text{LLM}(\mathcal{M}_i, h_i, c_i, \{r_{u,i}\}) \vee \text{LLM}(\mathcal{M}_i, h_i, d_i), \tag{4}$$

Here, $\mathcal{M}_i$ depicts the generation instruction. We consider two scenarios. One includes a description in the raw data (right-hand side). The other does not include a description (left-hand side). In the latter case, we utilize collaborative reviews to inform the profile. Using LLMs, we can generate an informative item profile $\mathcal{P}_i$.

3.1.2 User Profiling

In practical scenarios, privacy concerns limit generating user profiles from demographic information. Instead, we profile users by leveraging their collaborative relationships through interacted item profiles. This approach captures collaborative signals that reflect user preferences. The user profile generation process is defined as:

$$\mathcal{P}_u = \text{LLM}(\mathcal{M}_u, \{h_i, \mathcal{P}_i, r_{u,i} \mid i \in \mathcal{N}_u\}). \tag{5}$$

Here, $\mathcal{M}_u$ is the instruction for using a LLM to generate the user profile. We sample interacted items $\mathcal{N}_u$ from the user's purchase history. We combine their feedback $r_{u,i}$ with the pre-generated item profiles $\mathcal{P}_i$. This creates the user's text description $\mathcal{P}_u$, capturing their preferences. The incorporation of sampled data ensures that LLMs accurately infer profiles. This includes item metadata and user reviews. The profiles authentically reflect users' interaction preferences. To enhance profile quality, we adopt the principles of Chain of Thought (CoT)(Wei et al., 2022) and Self-Consistency(Wang et al., 2023). We require LLMs to generate explanatory rationales alongside the profiles. It improves inference reliability. It also establishes an interpretable connection between collaborative signals and profile generation.

3.1.3 Advantages of Collaborative Profiling

Our collaborative profiling framework offers two key advantages for real-world recommendation:

- **Preservation of Collaborative Information**. Our approach captures the original textual content. It also captures the semantics of user/item characteristics and their interaction patterns. We encode these profiles into a shared feature space. This uses a recommendation-oriented language model. The embeddings of interacted users and items are aligned. This allows recommenders to better identify relevant matches.
- **Rapid Adaptation to Dynamic Scenarios**. Our profiling enables the recommender system to adapt to evolving user preferences. It also adapts to changing interaction patterns. With robust language models, simple updates to textual user profiles can quickly reflect shifts in interests. They can also reflect shifts in behaviors. This flexibility makes our approach ideal for environments. User interests change over time in environments.

3.2 Profile Embedder with Collaborative LM

We have generated rich textual profiles for users and items, moving beyond conventional ID-based embeddings. However, directly encoding these textual profiles into latent embeddings for recommendations has two key limitations:

- **Capturing Recommendation-Specific Semantics**. While text embeddings are expressive, they may not optimize for the specific semantics relevant to recommendations. For example, consider two user profiles: (i) "This user is passionate about advanced AI techniques, focusing on deep learning and research." (ii) "With a passion for advanced AI development, this user enjoys science fiction and AI-themed novels." Although both profiles mention AI, their target audiences differ—one caters to AI scientists, while the other targets sci-fi readers. Directly encoding these profiles may overlook recommendation-specific semantics, necessitating refinement to align embeddings with the context of recommendation.
- **Overlooking High-Order Collaborative Signals**. While textual profiles provide rich semantic information, relying solely on them may cause us to miss valuable high-order collaborative patterns. These patterns arise from complex user-item interactions (Wang et al., 2019; Xia et al., 2023). Such signals include transitive associations and

community-level preferences. They offer additional insights that enhance preference learning.

To address these limitations, we propose a collaborative language modeling paradigm. It integrates the semantic richness of profiles with valuable collaborative signals from complex interactions.

3.2.1 Bidirectional Transformer Embedder

We use a multi-layer bidirectional Transformer encoder as the backbone for two key benefits: 1) **Efficient Encoding**: The encoder-only architecture generates effective text representations, allowing faster inference in recommendation. 2) **Flexible Adaptation**: Leveraging pre-trained Transformer models enables us to optimize the embedder for specific recommendation tasks effectively.

Let's consider a user's profile as a passage of n words: $\mathcal{P} = w_1, \ldots, w_n$. We start by adding a special token [CLS] at the beginning of the word sequence. The tokenization layer Tok$(\cdot)$ then encodes the input sequence into initial embeddings, which serve as the input for the Transformer layers:

$$\{\mathbf{x}^{(0)}_{\texttt{[CLS]}}, \ldots \mathbf{x}^{(0)}_n\} = \text{Tok}(\{w_{\texttt{[CLS]}}, \ldots, w_n\}). \quad (6)$$

Here, $x^{(0)} \in \mathbb{R}^d$ is the embedding from the embedding table for the tokens, with the (0) superscript indicating it is the input to the (0)-th layer of the language model. The tokenization process also adds positional embeddings. The language model then encodes a sequence of final embeddings:

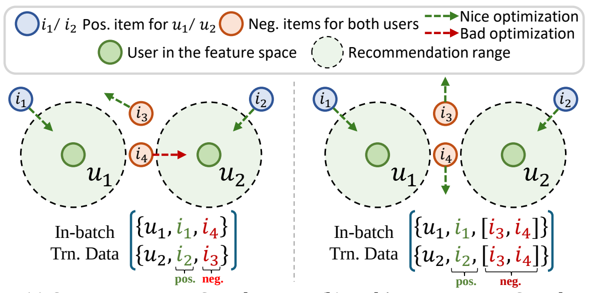
$$\{\mathbf{e}_{[CLS]}, \ldots, \mathbf{e}_n\} = \text{Enc}(\{\mathbf{x}^{(0)}_{\texttt{[CLS]}}, \ldots \mathbf{x}^{(0)}_n\}), \quad (7)$$

where Enc$(\cdot)$ refers to the Transformer-based encoder-only LM. The key operation in the encoding process is the self-attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d})V$$
$$\textit{w.r.t. } Q = XW^Q,\ K = XW^K,\ V = XW^V. \quad (8)$$

Here, $X \in \mathbb{R}^{n \times d}$ represents the stack of token embeddings, while $W^{Q/K/V}$ are parameter matrices that map these embeddings into queries, keys, and values. This self-attention mechanism enables each token to aggregate information from all others, ensuring awareness of the entire sequence. We then select the first embedding $\mathbf{e}_{\texttt{[CLS]}}$ as the representative embedding for the profile. This embedding is passed through a multi-layer perceptron to obtain the encoded representation e, as shown in Eq.(1):

$$\mathbf{e} = \text{MLP}(\mathbf{e}_{\texttt{[CLS]}}) = \text{LM}(\mathcal{P}). \quad (9)$$



(a) One-Neg. Item per Sample **(b) Multi-Neg. Items per Sample**

Figure 3: Contrastive tuning of the collaborative LM enables it to learn rich representations. It aligns the text-based semantic space with global collaborative signals.

With these encoded text embeddings $\mathbf{e}$ for each user and item, we can predict the score of interaction using cosine similarity and make recommendations as described in Eq.(2).

3.2.2 Contrastive Collaborative LM Learning

We fine-tune the collaborative LM using contrastive learning to effectively capture high-order collaborative signals. Traditional recommenders using Bayesian Personalized Ranking (BPR) (Rendle et al., 2012) optimize embeddings with only one negative item per sample. This limits their ability to capture complex global user-item relationships.

The supervised contrastive loss offers a strong alternative to traditional methods. By treating interacted user-item pairs as positives and non-interacted pairs as negatives, it brings related item embeddings closer in the feature space. As shown in Figure 3, contrastive learning uses a batch of negatives for a comprehensive adjustment of the encoded space, enabling the model to capture high-order collaborative relationships. We evaluate this in Appendix E to assess the impact of different learning objectives. The learning objective for tuning the language model is expressed as follows:

$$\mathcal{L}_{\text{con}} = -\sum \log \frac{\exp(s_{u,i^+}/\tau)}{\sum_{j \in \mathcal{N}^-} \exp(s_{u,j}/\tau)}, \quad (10)$$

where $s_{u,j} = \cos(\mathbf{e}_u, \mathbf{e}_j)$ denotes the cosine similarity between user u and item j, i^+ represents the positive item that user u has interacted with, τ is a temperature hyperparameter that controls the degree of learning, and $\mathcal{N}^-$ is the set of in-batch negative items. We also build on prior work (Li et al., 2023; Hou et al., 2024a) by incorporating an auxiliary masked language modeling (MLM) loss $\mathcal{L}_{\text{mlm}}$. This technique randomly masks input tokens, training the model to predict them, which stabilizes training and enhances generalization. The final training objective combines the contrastive loss and the MLM loss:

$$\mathcal{L} = \mathcal{L}_{\text{con}} + \lambda \mathcal{L}_{\text{mlm}}. \tag{11}$$

λ is a hyperparameter that balances contrastive learning with masked language modeling loss.

3.3 Augmentation with Profile Diversification

To enhance generalization to unseen users/items, we propose profile diversification. Single profiles per user or item limit representation diversity and training quality, hurting performance. Our augmentation creates multiple profiles per entity while preserving semantic meaning—capturing personalized preferences for users and varied characteristics.

Inspired by self-instruction mechanisms (Wang et al., 2023; Xu et al., 2024), LLMs can rephrase user or item profiles while preserving their underlying meaning. This generates multiple semantically similar yet distinctly worded profiles from single inputs. Iterative rephrasing creates diverse augmented profiles, substantially expanding available training data. This technique proves particularly valuable for limited datasets, as LLM-generated profiles improve model generalization and robustness. Through LLM-based diversification, we create diverse profile sets for each user and item.

$$\{\mathcal{P}\}_u = \{\mathcal{P}_u;\ \mathcal{P}_u^1,\ \mathcal{P}_u^2,\ \dots,\ \mathcal{P}_u^t\}, \tag{12}$$

$$\{\mathcal{P}\}_i = \{\mathcal{P}_i;\ \mathcal{P}_i^1,\ \mathcal{P}_i^2,\ \dots,\ \mathcal{P}_i^t\}. \tag{13}$$

Here, $\mathcal{P}_{u/i}$ represents the original profile, while $\mathcal{P}_{u/i}^{1-t}$ denotes the LLM-rephrased profiles, with t indicating the number of diversification steps. During training, we randomly select one profile from the user's or item's profile set for each batch data.

4 Evaluation

We evaluate the EasyRec in addressing the following research questions (RQs): **RQ1**: How effectively does the EasyRec perform in matching unseen users and items (zero-shot) within text-based recommendation? **RQ2:** How effectively does EasyRec integrate to enhance recommendations in text-based collaborative filtering scenarios? **RQ3**: How effective is our profile diversification mechanism for augmenting data and improving the recommendation model's performance? **RQ4**: How well can our proposed text-based EasyRec adapt to changes in users' dynamic preferences?

Table 1: Dataset statisics: "Avg. n" is the average interactions per user, and "Inters." stands for interactions. Datasets with underlines are from different platforms.

Datasets	#Users	#Items	#Inters.	Avg. n
Train Data	124,732	67,455	802,869	6.44
-Arts	14,470	8,537	96,328	6.66
-Games	17,397	8,330	120,255	6.91
-Movies	16,994	9,370	134,649	7.92
-Home	22,893	13,070	131,556	5.75
-Electronics	26,837	14,033	165,628	6.17
-Tools	26,141	14,155	154,453	5.91
Test Data	55,877	28,988	615,210	11.01
-Sports	21,476	12,741	132,400	6.17
-Steam	23,310	5,237	316,190	13.56
-Yelp	11,091	11,010	166,620	15.02

Table 2: Model variants of EasyRec differ by parameter size. "HS" stands for hidden size.

Model	Layers	HS	Heads	Params
EasyRec-Small	6	768	12	82M
EasyRec-Base	12	768	12	125M
EasyRec-Large	24	1024	16	355M

4.1 Experimental Settings

4.1.1 Datasets

To assess our proposed model's capability in encoding user/item textual profiles into embeddings for recommendation, we curated diverse datasets across various domains and platforms. A portion was used for training, while the remainder served as test sets for zero-shot evaluation. The dataset statistics are shown in Table 1. Due to the page limit, we place the detail of data resources are described in Appendix F.1

4.1.2 Evaluation Protocols

We employ two ranking-based evaluation metrics, Recall@N and NDCG@N, to assess performance in both text-based recommendation and collaborative filtering scenarios. Specifically, we compute these metrics for N values of 10 and 20 (He et al., 2017, 2020). The evaluation is conducted using the all-rank protocol (He et al., 2020) with the predicted preference scores. In the context of text-based recommendation, particularly for datasets containing multiple LLM-diversified profiles (as discussed in Section 3.3), we calculate the metrics separately t times based on different profile pairs $(\mathcal{P}_u^{1,\dots,t}, \mathcal{P}_i^{1,\dots,t})$. Subsequently, we compute the mean value for each metric to obtain a comprehensive assessment. For the training datasets, we utilize the validation split for evaluation, while for the test datasets, we employ the test split.

Table 3: Text-based recommendation performance of various LMs across different datasets. The best performance is indicated in **bold**, while the second-best is highlighted with underline. The script $*$ denotes significance ($p < 0.05$).

Data	Sports				Steam				Yelp			
Methods	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20
Proprietary Models												
OpenAIv3-Small	0.0324	0.0444	0.0198	0.0230	0.0066	0.0119	0.0049	0.0068	0.0028	0.0056	0.0021	0.0031
OpenAIv3-Large	0.0300	0.0436	0.0180	0.0217	0.0070	0.0137	0.0049	0.0073	0.0029	0.0055	0.0023	0.0032
Base Size Models												
BERT-Base	0.0015	0.0032	0.0008	0.0013	0.0015	0.0031	0.0011	0.0017	0.0009	0.0018	0.0006	0.0010
RoBERTa-Base	0.0121	0.0206	0.0065	0.0087	0.0041	0.0078	0.0031	0.0043	0.0018	0.0034	0.0014	0.0020
BART-Base	0.0088	0.0143	0.0048	0.0063	0.0033	0.0062	0.0024	0.0035	0.0014	0.0027	0.0012	0.0016
SimCSE-Base	0.0240	0.0341	0.0143	0.0170	0.0048	0.0091	0.0035	0.0050	0.0020	0.0039	0.0015	0.0022
BLaIR-Base	0.0251	0.0352	0.0145	0.0173	0.0041	0.0081	0.0030	0.0044	0.0025	0.0046	0.0019	0.0026
GTR-Base	0.0290	0.0417	0.0172	0.0206	0.0075	0.0136	0.0056	0.0077	0.0025	0.0052	0.0019	0.0029
BGE-Base	0.0315	0.0437	0.0194	0.0227	0.0094	0.0170	0.0069	0.0095	0.0029	0.0053	0.0022	0.0031
Large Size Models												
BERT-Large	0.0011	0.0021	0.0006	0.0008	0.0019	0.0040	0.0014	0.0022	0.0008	0.0019	0.0007	0.0010
RoBERTa-Large	0.0039	0.0065	0.0021	0.0028	0.0027	0.0052	0.0021	0.0029	0.0012	0.0023	0.0010	0.0014
BART-Large	0.0114	0.0170	0.0064	0.0080	0.0036	0.0065	0.0026	0.0037	0.0017	0.0033	0.0014	0.0019
SimCSE-Large	0.0232	0.0328	0.0134	0.0160	0.0051	0.0095	0.0037	0.0052	0.0023	0.0044	0.0017	0.0025
BLaIR-Large	0.0227	0.0322	0.0133	0.0159	0.0057	0.0108	0.0041	0.0059	0.0027	0.0047	0.0019	0.0027
GTR-Large	0.0329	0.0446	0.0198	0.0229	0.0095	0.0168	0.0069	0.0094	0.0021	0.0042	0.0018	0.0025
BGE-Large	0.0324	0.0449	0.0196	0.0230	0.0089	0.0153	0.0066	0.0088	0.0025	0.0052	0.0020	0.0029
EasyRec Series												
EasyRec-Small	0.0186	0.0286	0.0108	0.0135	0.0097	0.0174	0.0070	0.0097	0.0022	0.0046	0.0017	0.0026
EasyRec-Base	0.0360	0.0518	0.0210	0.0253	0.0114	0.0203	0.0081	0.0112	0.0034	0.0063	0.0026	0.0037
EasyRec-Large	**0.0396***	**0.0557***	**0.0236***	**0.0279***	**0.0129***	**0.0225***	**0.0093***	**0.0127***	**0.0034***	**0.0065***	**0.0026***	**0.0037***
Improve	↑ 20.36%	↑ 24.05%	↑ 19.19%	↑ 21.30%	↑ 35.79%	↑ 32.35%	↑ 39.13%	↑ 33.68%	↑ 17.24%	↑ 16.07%	↑ 13.04%	↑ 15.63%

4.2 Performance Comparision for Text-based Recommendation (RQ1)

We evaluate the performance of various language models (LMs) for zero-shot text-based recommendation on the unseen Sports, Steam, and Yelp datasets. This approach directly leverages the encoded embeddings derived from user/item profiles to make recommendations, without any additional training on the target datasets.

4.2.1 Baseline Methods and Settings

We have included the following state-of-the-art language models as text embedders for comparative evaluation: (i) General Language Models: **BERT** (Devlin et al., 2018), **RoBERTa** (Liu et al., 2019) and **BART** (Lewis et al., 2019); (ii) Language Models for Dense Retrieval: **SimCSE** (Gao et al., 2021), **GTR** (Ni et al., 2021) and **BGE** (Xiao et al., 2023); (iii) Pre-trained Language Models for Recommendation: **BLaIR** (Hou et al., 2024a). We also compare with SoTA text embedding models provided by **OpenAI**. Details of the baseline models are described in Appendix G.

4.2.2 Result Analysis

The overall comparison of different models is presented in Table 3. This evaluation reveals several observations, which are outlined below:

- **Superiority across Diverse Datasets**. Our evaluation consistently shows that the EasyRec outperforms all other models across the datasets spanning different platforms. This provides strong evidence for the effectiveness of the EasyRec. We attribute these improvements to two key factors: i) By injecting collaborative signals into the LMs, we effectively optimized our EasyRec using supervised contrastive learning within the recommendation context. This approach allows the model to inherently encode user and item text embeddings that are well-suited for recommendation tasks. ii) By integrating a diverse array of datasets across multiple categories and utilizing data augmentation techniques to enrich the text descriptions for training, our EasyRec exhibits impressive generalization capabilities, enabling it to effectively handle unseen data.
- **Scaling Law Investigation of EasyRec Model**. Our experiments reveal that as the size of the EasyRec increases, its performance consistently improves across all the datasets. This observation reflects a scaling law, where the model's performance growth is directly correlated with its size. Furthermore, this finding effectively reinforces the validity of text-based recommendation systems. It also validates our approach to training the LMs, which enables it to learn collaborative signals from a new perspective.

4.3 Performance of Text-enhanced CF (RQ2)

In addition to our investigation of zero-shot recommendation, we explore the potential of EasyRec as an enhancement when integrated with CF models. To assess the effectiveness of various LMs in CF, we employ two widely used ID-based methods as backbone models: **GCCF** (Chen et al., 2020) and **LightGCN** (He et al., 2020), which were chosen for their proven effectiveness and ef-

Table 4: Recommendation performance in text-enhanced CF. The experiment was conducted on the Steam dataset with 5-runs to obtain the mean results.

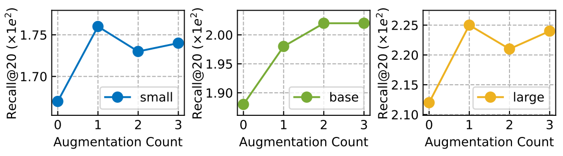
Metric	Recall @10	Recall @20	NDCG @10	NDCG @20
ID-based Methods				
GCCF	0.0826	0.1314	0.0665	0.0830
LightGCN	0.0851	0.1349	0.0686	0.0854
Text-enhanced GCCF				
BERT	0.0822	0.1313	0.0663	0.0829
RoBERTa	0.0848	0.1351	0.0684	0.0854
BART	0.0874	0.1383	0.0701	0.0874
SimCSE	0.0877	0.1395	0.0706	0.0881
BLaIR	0.0880	0.1392	0.0708	0.0882
GTR	0.0873	0.1387	0.0706	0.0880
BGE	0.0875	0.1393	0.0705	0.0881
EasyRec	**0.0881**	**0.1402**	**0.0712**	**0.0888**
Text-enhanced LightGCN				
BERT	0.0849	0.1347	0.0684	0.0852
RoBERTa	0.0867	0.1374	0.0699	0.0870
BART	0.0887	0.1407	0.0715	0.0891
SimCSE	0.0891	0.1417	0.0719	0.0898
BLaIR	0.0897	0.1418	0.0724	0.0901
GTR	0.0894	0.1417	0.0719	0.0896
BGE	0.0891	0.1407	0.0718	0.0893
EasyRec	**0.0908**	**0.1430**	**0.0732**	**0.0908**

ficiency. Furthermore, we utilize the advanced model-agnostic text-enhanced framework RLM-Rec (Ren et al., 2024) with contrastive alignment to conduct our investigation. We compare the large versions of both EasyRec and other open-source LMs. The findings from the results in Table 4 are:

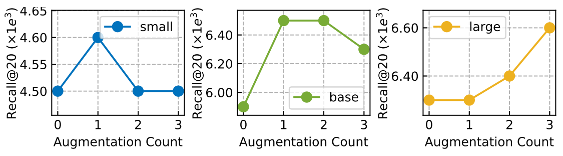
- Compared to backbones, the integration of the text-enhanced framework generally improves the performance for both GCCF and LightGCN. This observation highlights the significance of incorporating text modality (*i.e.*, user/item profiles) into the recommendation paradigm.
- Among the various LMs, EasyRec consistently achieves the highest performance in the text-enhanced recommenders. This outcome not only illustrates the efficacy of EasyRec for recommendation, but also emphasizes the advantages of incorporating collaborative information into LMs.

4.4 Efficacy of Profile Diversification (RQ3)

In this section, we examine the impact of diversifying user and item profiles with large language models (LLMs) on model performance. As mentioned in Section 3.3, we perform LLM-based diversification three times on the original generated profiles. This process continuously increases the number of profiles in the training set. To investigate whether data augmentation positively affects model performance, we conduct experiments with three variants of the EasyRec under different numbers of diversified profiles. The results are shown in Figure 4, leading to the following key observations:

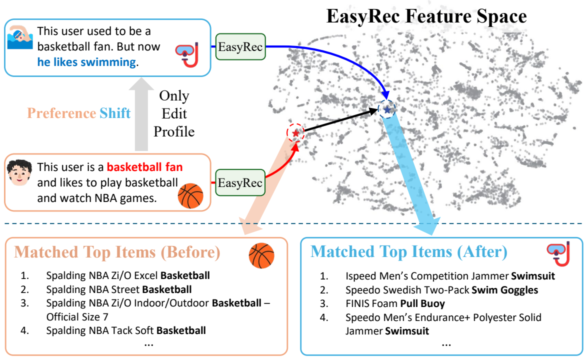


(a) Performance on Steam data



(b) Performance on Yelp data

Figure 4: Performance *w.r.t.* data size. "Augmentation Count" indicates the number t of diversified profiles.



This user used to be a basketball fan. But now he likes swimming.

This user is a basketball fan and likes to play basketball and watch NBA games.

Matched Top Items (Before)

1. Spalding NBA Zi/O Excel **Basketball**
2. Spalding NBA Street **Basketball**
3. Spalding NBA Zi/O Indoor/Outdoor **Basketball** – Official Size 7
4. Spalding NBA Tack Soft **Basketball**

...

Matched Top Items (After)

1. Ispeed Men's Competition Jammer **Swimsuit**
2. Speedo Swedish Two-Pack **Swim Goggles**
3. FINIS Foam **Pull Buoy**
4. Speedo Men's Endurance+ Polyester Solid Jammer **Swimsuit**

...

Figure 5: Case study on handling user preference shift.

- **Effectiveness of Profile Diversification**. The increase in the number of diversified profiles enhances model performance, particularly for larger models. This finding underscores the effectiveness of our augmentation approach using LLMs for profile diversification, and emphasizes the significance of increasing training corpus for improved recommendation outcomes.
- **Scaling Relationship**: The scaling experiments on both model size and data size reveal a crucial relationship that influences model performance. This demonstrates that our approach of training the LM with collaborative signals follows a scaling law, indicating that model performance benefits from both increased capacity and data volume. Such findings are vital as they provide insights into how model capacity and data availability interact, guiding future research and development.

Moreover, we find that profile diversification can improve recommendation diversity. We conduct experiments and discussions in Appendix F.5.

4.5 Fast Adaptation Case Study (RQ4)

As mentioned in Section 3.1.3, a key advantage of EasyRec is its ability to empower recommender systems to efficiently adapt to shifts in user preferences and behavior dynamics over time. To evaluate this capability, we create two user profiles reflecting shifted preferences on the *Amazon-Sport* dataset and examine the recommended items from the EasyRec. As shown in Figure 5, the original user profile indicates that the user enjoys playing basketball. However, the user's preference later transitions to a preference for swimming.

We visualize all the encoded embeddings using t-SNE (Van der Maaten and Hinton, 2008), as illustrated in Figure 5, which reveals a significant shift in the user embedding within the feature space. Correspondingly, recommended items transition from basketball-related products to swimming gear, reflecting the user's changing preferences. Notably, this adjustment is accomplished solely by modifying the user's profile, without further training of the model. This underscores the efficiency and flexibility in adapting to evolving user preferences.

5 Related Work

LM-Powered Recommender Systems. Recent recommendation methods increasingly incorporate text modalities (Yuan et al., 2023; Wu et al., 2024; Zhang et al., 2023), using LM-derived semantic embeddings to boost CTR prediction and sequence recommendation (Xi et al., 2023; Geng et al., 2024; Hou et al., 2022; Sheng et al., 2024). Some approaches use text-based agents to boost performance (Zhang et al., 2024a,b), with notable examples like ZESRec (Ding et al., 2021) and MoRec (Yuan et al., 2023) leveraging text embeddings for inductive performance. RLMRec applies information-theoretic text augmentation to ID-based recommenders (Ren et al., 2024). However, most systems depend on general-purpose encoders (*e.g.*, BERT) rather than recommendation-tailored LMs. Inspired by BLaIR's use of feedback and metadata (Hou et al., 2024a), we train a recommendation-specific LM on user profiles and collaborative signals to improve zero-shot and text-augmented performance.

Cross-Domain Recommendation. Cross-domain recommendation enhances recommendations in one domain by leveraging data from another domain to combat data sparsity and improve personalization (Zang et al., 2022). Techniques like graph collaborative filtering (Liu et al., 2020) use Graph Neural Networks (GNNs) to aggregate common and domain-specific user features. Recent studies have integrated self-supervised learning, such as C2DSR (Cao et al., 2022) with contrastive learning for both single and cross-domain representations, and CCDR (Xie et al., 2022) with intra- and inter-domain contrastive learning. SITN (Sun et al., 2023) utilizes self-attention to represent user sequences from source and target domains, followed by contrastive learning. However, current cross-domain approaches often rely on correlations between source and target data, limiting their application to zero-shot tasks. In contrast, our proposed EasyRec employs text-based zero-shot learning, removing these constraints and enabling effective transfer across different domains.

Graph Collaborative Filtering for Recommendation. Graph Neural Networks (GNNs) are effective for recommendation by modeling user–item interactions and high-order dependencies (Gao et al., 2023). Representative models include PinSage (Ying et al., 2018), NGCF (Wang et al., 2019) and LightGCN (He et al., 2020), which learn node representations via neighborhood aggregation. To address data sparsity, recent studies combine self-supervised learning with collaborative filtering using graph augmentations; notable examples are SGL (Wu et al., 2021), SimGCL (Yu et al., 2022) and HCCF (Xia et al., 2022). These works demonstrate the promise of self-supervised graph learning for recommender systems.

6 Conclusion

This paper presents EasyRec which integrates LMs to enhance recommendation. Our approach is simple yet effective, excelling in scenarios like text-based zero-shot recommendations and text-enhanced CF. Central to EasyRec's success is its combination of collaborative LM tuning and contrastive learning, which captures nuanced semantics and high-order collaborative signals, leading to improved recommendations. Extensive experiments demonstrate EasyRec's superiority over existing models, showcasing its adaptability to changing user preferences and real-world applications.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants 624B2122.

Limitation

While EasyRec shows promising advancements in generalization for zero-shot recommendation, it faces challenges related to data modality diversity. Currently, our approach relies primarily on textual data, whereas items can encompass a richer variety of modalities. The absence of visual inputs, such as images and videos, limits the contextual information we can leverage. These visual elements have the potential to convey aesthetic preferences, cultural trends, and emotional responses, capturing nuances that textual data might overlook. They can also reveal visual patterns linked to user behavior and preferences, including color schemes, styles, and settings, which are essential for effective personalization. Acknowledging these aspects, future work could explore the integration of multimodal data processing techniques, potentially enhancing predictive accuracy and improving the system's ability to generalize recommendations.

References

Jiangxia Cao, Xin Cong, Jiawei Sheng, Tingwen Liu, and Bin Wang. 2022. Contrastive cross-domain sequential recommendation. In *CIKM*, pages 138–147.

Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *AAAI*, volume 34, pages 27–34.

Maurizio Ferrari Dacrema, Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. 2012. Design and evaluation of cross-domain recommender systems. In *Recommender Systems Handbook*, pages 485–516. Springer.

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

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

Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. 2023. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM TORS*, 1(1):1–51.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Binzong Geng, Zhaoxin Huan, Xiaolu Zhang, Yong He, Liang Zhang, Fajie Yuan, Jun Zhou, and Linjian Mo. 2024. Breaking the length barrier: Llm-enhanced ctr prediction in long textual user behaviors. In *SIGIR*, pages 2311–2315.

Bowen Hao, Jing Zhang, Hongzhi Yin, Cuiping Li, and Hong Chen. 2021. Pre-training graph neural networks for cold-start users and items representation. In *WSDM*, pages 265–273.

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

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*, pages 173–182.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024a. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*.

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *KDD*, pages 585–593.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024b. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*, pages 364–381. Springer.

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

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text is all you need: Learning language representations for sequential recommendation. In *KDD*, pages 1258–1267.

Juanhui Li, Haoyu Han, Zhikai Chen, Harry Shomer, Wei Jin, Amin Javari, and Jiliang Tang. 2024. Enhancing id and text fusion via alternative training in session-based recommendation. *arXiv preprint arXiv:2402.08921*.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *SIGIR*, pages 1785–1795.

Xixun Lin, Jia Wu, Chuan Zhou, Shirui Pan, Yanan Cao, and Bin Wang. 2021. Task-adaptive neural process for user cold-start recommendation. In *WWW*, pages 1306–1316.

Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *WWW*, pages 2320–2329.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Meng Liu, Jianjun Li, Guohui Li, and Peng Pan. 2020. Cross domain recommendation via bi-directional transfer graph collaborative filtering networks. In *CIKM*, pages 885–894.

Yinhan Liu et al. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, pages 188–197.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32.

Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *WWW*, pages 3464–3475.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*.

Leheng Sheng, An Zhang, Yi Zhang, Yuxin Chen, Xiang Wang, and Tat-Seng Chua. 2024. Language models encode collaborative signals in recommendation. *arXiv preprint arXiv:2407.05441*.

Guoqiang Sun, Yibin Shen, Sijin Zhou, Xiang Chen, Hongyan Liu, Chunming Wu, Chenyi Lei, Xianhui Wei, and Fei Fang. 2023. Self-supervised interest transfer network via prototypical contrastive learning for recommendation. *arXiv preprint arXiv:2302.14438*.

Jianing Sun, Zhaoyue Cheng, Saba Zuberi, Felipe Pérez, and Maksims Volkovs. 2021. Hgcf: Hyperbolic graph convolution networks for collaborative filtering. In *WWW*, pages 593–601.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *JMLR*, 9(11).

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*, pages 165–174.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *ACL*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 35:24824–24837.

Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive learning for cold-start recommendation. In *MM*, pages 5382–5390.

Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *SIGIR*, pages 726–735.

Xuansheng Wu, Huachi Zhou, Yucheng Shi, Wenlin Yao, Xiao Huang, and Ninghao Liu. 2024. Could small language models serve as recommenders? towards data-centric cold-start recommendation. In *WWW*, pages 3566–3575.

Yunjia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, Rui Zhang, et al. 2023. Towards open-world recommendation with knowledge augmentation from large language models. *arXiv preprint arXiv:2306.10933*.

Lianghao Xia, Chao Huang, Chunzhen Huang, Kangyi Lin, Tao Yu, and Ben Kao. 2023. Automated self-supervised learning for recommendation. In *WWW*, pages 992–1002.

Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy Huang. 2022. Hypergraph contrastive collaborative filtering. In *SIGIR*, pages 70–79.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*.

Ruobing Xie, Qi Liu, Liangdong Wang, Shukai Liu, Bo Zhang, and Leyu Lin. 2022. Contrastive cross-domain recommendation in matching. In *KDD*, pages 4226–4236.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2024. Wizardlm: Empowering large language models to follow complex instructions. In *ICLR*.

Shuyuan Xu, Yingqiang Ge, Yunqi Li, Zuohui Fu, Xu Chen, and Yongfeng Zhang. 2023. Causal collaborative filtering. In *SIGIR*, pages 235–245.

Menglin Yang, Min Zhou, Jiahong Liu, Defu Lian, and Irwin King. 2022. Hrcf: Enhancing collaborative filtering via hyperbolic geometric regularization. In *WWW*, pages 2462–2471.

Xihong Yang, Heming Jing, Zixing Zhang, Jindong Wang, Huakang Niu, Shuaiqiang Wang, Yu Lu, Junfeng Wang, Dawei Yin, Xinwang Liu, et al. 2024. Darec: A disentangled alignment framework for large language model and recommender system. *arXiv preprint arXiv:2408.08231*.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *KDD*, pages 974–983.

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *SIGIR*, pages 1294–1303.

Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. In *SIGIR*, pages 2639–2649.

Tianzi Zang, Yanmin Zhu, Haobing Liu, Ruohan Zhang, and Jiadi Yu. 2022. A survey on cross-domain recommendation: taxonomies, methods, and future directions. *ACM TOIS*, 41(2):1–39.

An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024a. On generative agents in recommendation. In *SIGIR*, pages 1807–1817.

Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024b. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *WWW*, pages 3679–3689.

Mi Zhang and Neil Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 123–130.

Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM CSUR*, 52(1):1–38.

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

Yiding Zhang, Chaozhuo Li, Xing Xie, Xiao Wang, Chuan Shi, Yuming Liu, Hao Sun, Liangjie Zhang, Weiwei Deng, and Qi Zhang. 2022. Geometric disentangled collaborative filtering. In *SIGIR*, pages 80–90.

Appendices

A Discussion on EasyRec

A.1 Motivation and Technical Positioning

The motivation behind our proposed EasyRec stems from the limitations of existing recommendation algorithms that primarily rely on ID-based representations. While ID-based representations can enhance algorithm performance, they fundamentally restrict the generalization capabilities of recommendation systems. Specifically, models trained in this manner cannot be directly transferred to new datasets in zero-shot recommendation scenarios, where there is no overlap between users and items. Recent studies (Ren et al., 2024; Sheng et al., 2024) have shown a growing interest in incorporating language representations (i.e., text embeddings) to replace or complement ID-based representations. This approach aims to improve the generalization and overall performance of these algorithms. However, the predominant reliance on general-purpose language models as text encoders has created a gap in the development of language models specifically tailored for recommendation scenarios, which could yield higher-quality user and item embeddings. To address this gap, we propose utilizing user and item profiles as fundamental textual representations. By training a language model specifically designed for recommendation algorithms, based on collaborative filtering signals and profile diversification, we can generate embeddings that are more suitable for recommendation contexts. This approach promises to deliver improved performance compared to general LMs.

A.2 Relationship between EasyRec and LLM-CF Methods

The proposed EasyRec constitutes a specialized class of language models meticulously trained for recommendation scenarios. Functioning as a textual encoder, its primary objective lies in transforming descriptive textual profiles of users and items into discriminative feature representations compatible with collaborative filtering architectures. The training process employs collaborative filtering signals as self-supervised signals to guide model optimization. The motivation of EasyRec stands distinct from approaches that combine LLMs with CF models as end-to-end frameworks (Ren et al., 2024; Yang et al., 2024). By contrast, EasyRec adopts a

Table 5: Performance comparison (Steam dataset) between text-based and ID-based recommenders

Model	R@10	R@20	N@10	N@20
LightGCN (ID-based)	0.0851	0.1349	0.0686	0.0854
AlphaRec + EasyRec	**0.0863**	**0.1358**	**0.0689**	**0.0857**

foundational modeling perspective, aiming to enhance existing text-enhanced algorithms through improved embeddings.

Recently, LLM-CF integration methods predominantly utilize text embeddings as auxiliary features. As empirically validated in our experiments on Text-enhanced Collaborative Filtering (Section 4.3), EasyRec shows superior performance compared to open-source text embedders.

A.3 Can Text-Only Frameworks Outperform ID-Based Recommendation?

ID-based recommendation systems have long served as the predominant paradigm in both industrial deployments and academic research. However, their inherent limitations in cross-domain generalization have recently propelled text-only recommendation approaches – the core focus of EasyRec – to the forefront of research attention. While text-based methods demonstrate enhanced generalization capabilities, the performance hierarchy between text-based and ID-based approaches remains empirically underexplored.

To investigate this research question, we integrate EasyRec with the state-of-the-art text-based framework AlphaRec (Sheng et al., 2024), conducting comparative analyses against the ID-based LightGCN backbone. To ensure robustness, we perform multiple runs and report averaged results. The experimental results detailed in Table 5 indicate that the text-only framework provides superior performance compared to ID-based methods.

B Implementation and Training Details

We implemented our EasyRec and conducted all experiments using PyTorch (Paszke et al., 2019). For the transformer-based encoder backbone, we adopted the architecture of RoBERTa (Liu et al., 2019) and utilized its pre-trained parameters as initialization. We trained three versions of EasyRec with varying parameter sizes (**small**, **base**, and **large**), as detailed in Table 2. For the loss function, we set the hyperparameters τ to 0.05 and λ to 0.1. The token masking ratio for masked language modeling is 0.15, and the learning rate is set to 5×10^{-5}. We train the model for 25 epochs.

Table 6: Performance comparison with other zero-shot frameworks on sports dataset.

Model	R@5	R@10	Inference Time
LLMRank	0.6155	0.7822	$\sim$1 hours
EasyRec	**0.6332**	**0.8078**	**3-4 minutes**

For profile augmentation, we set the diversification time t for LLM-based methods to 3. During training, we evaluate the model every 1000 steps and use the validation interactions from each training dataset to select the optimal model parameters, employing the Recall@20 metric. Detailed implementation of our model is provided in the code.

C Comparison with Zero-shot Frameworks

The exploration of zero-shot recommendation has garnered significant attention, particularly in sequential recommendation scenarios (Ding et al., 2021). In our work, we propose a novel zero-shot research line for collaborative filtering (CF) based on semantic profile matching. To demonstrate its effectiveness, we compare it with another fundamental zero-shot framework, LLMRank (Hou et al., 2024b), which leverages LLMs for item ranking. Specifically, we implement LLMRank with `GPT-4o-mini` to rank candidate items based on historical interactions and item titles, while our method employs user/item profile embeddings for feature-based ranking. Following the same experimental protocol, for each user in the test set we construct a candidate set containing 19 negative items and 1 ground-truth item. As shown in Table 6, our approach demonstrates better recall performance while achieving higher efficiency.

D Profile Generation with Different Large Language Models

To explore whether the trained EasyRec can generalize to different LLMs that under different tokenizers, we regenerate alternative user/item profiles using DeepSeek-V3 (Liu et al., 2024) on the Sports dataset and performed a performance validation based on EasyRec-Large. The results on Sports dataset are shown in Table 7, from which we can observed that the profiles generated based on GPT-3.5 and those generated by DeepSeek have a minimal impact on performance. This is because the final profiles are presented in text form. During the training of EasyRec, we exposed it to a large diver-

```
User Profile Diversification
```

Instruction

```
You will assist me in revising a user's profile while maintaining its original meaning. I will
present you with the user's initial profile.

Instructions:
USER PROFILE: The original user profile.

Requirements:
1. Please provide the revised profile directly begin with "REVISED PROFILE: ".
2. The rephrased profile should minimize duplication with the original text while preserving
its intended meaning.
3. The revised profile should exhibit varied sentence structures while faithfully conveying the
original profile's essence.
```

Input Prompt

```
USER PROFILE: This user is likely to enjoy items related to baking, entertaining, and colorful
table settings. They appreciate convenience, efficiency, and practicality in kitchen appliances.
```

Response

```
REVISED PROFILE: An individual who finds pleasure in baking, hosting gatherings, and vibrant table
arrangements. They value kitchen devices that offer convenience, efficiency, and practicality.
```

Figure 6: An example of large language models-based user profile diversification.

Table 7: Performance comparison with various LLM-generated profiles on sports dataset.

Profile Generator	R@10	R@20	N@10	N@20
GPT-3.5	0.0423	0.0586	0.0250	0.0294
DeepSeek-V3	0.0417	0.0555	0.0253	0.0290

Table 8: Comparison of EasyRec with different learning objectives (where "Contrast" stands for contrastive).

Objective	Sports		Yelp	
	R@10	N@10	R@10	N@10
BPR Loss	0.0381	0.0226	0.0028	0.0021
Contrast Loss	**0.0396**	**0.0236**	**0.0034**	**0.0026**

sity of recommendation-related texts and utilized multiple datasets for joint optimization, effectively avoiding overfitting on specific profile patterns.

E Impact of Training Objectives

To evaluate the impact of different training objectives on the LM's learning, we also implemented EasyRec-Large training with BPR loss (i.e., one negative item per training sample) for comparison with the contrastive learning results. As shown in Table 8, the model performance generally outperforms that trained with BPR loss, demonstrating the effectiveness of using contrastive learning to incorporate collaborative information into the LMs.

F Datasets and User/Item Profiles

In this section, we provide detailed information on the processes for profile generation and diversification, including instructions, examples, and associated costs.

F.1 Details of Dataset

We utilize Amazon review data (Ni et al., 2019) across six categories to form the training data: *Arts, Crafts and Sewing* (**Arts**), *Movies and TV* (**Movies**), *Video Games* (**Games**), *Home and Kitchen* (**Home**), *Electronics* (**Electronics**), and *Tools and Home Improvement* (**Tools**). For the test datasets, we use one domain, *Sports and Outdoors* (**Sports**), from the Amazon review data, along with two cross-platform datasets: **Steam** and **Yelp**, for comprehensive evaluation. For the datasets from the Amazon platform, we first filter the data to include only those with a rating score ≥ 3 and apply a 10-core filtering to densify the dataset. Subsequently, for each category, we split the interactions into training, validation, and test splits in a ratio of 8:1:1. In contrast, for the Steam and Yelp datasets, we directly use the data processed in RLMRec (Ren et al., 2024), which maintains a split ratio of 3:1:1.

F.2 Details of Profile Generation

After the data processing described in Section 4.1.1, each dataset contains a split of training interactions. We use these interactions to generate user

Item Profile Diversification

Instruction

```
You will assist me in revising a item's profile while maintaining its original meaning. I will
present you with the item's initial profile.

Instructions:
ITEM PROFILE: The original item profile.

Requirements:
1. Please provide the revised profile directly begin with "REVISED PROFILE: ".
2. The rephrased profile should minimize duplication with the original text while preserving
its intended meaning.
3. The revised profile should exhibit varied sentence structures while faithfully conveying the
original profile's essence.
```

Input Prompt

```
ITEM PROFILE: The Innovee Lemon Squeezer is a high-quality stainless steel manual citrus press
that comes with a lemon recipes ebook. Ideal for those who enjoy fresh lemon juice and recipes.
```

Response

```
REVISED PROFILE: The Innovee Lemon Squeezer is a stainless steel manual citrus press that
includes a lemon recipes ebook, perfect for individuals who appreciate the taste of freshly
squeezed lemon juice and love trying out new recipes.
```

Figure 7: An example of large language models-based item profile diversification.

and item profiles according to the paradigm outlined in Section 3.1, as this requires user-item interaction information. For the Steam and Yelp datasets, we directly utilize the profiles generated by RLMRec (Ren et al., 2024), which follow the protocol that leverages review information for user and item profile generation. It is important to note that the profiles for each dataset are generated exclusively based on the training interactions. This approach ensures that validation and test interactions are reserved for evaluation purposes, preventing data leakage and allowing for a more accurate assessment of the model's generalization performance on unseen recommendation data.

The profiling process adopts an item-to-user paradigm, where we first generate item profiles in parallel using multi-thread processing, followed by the parallel generation of user profiles that incorporate collaborative information. The large language model employed is `GPT-3.5-Turbo` from OpenAI. Specifically, in this work, we process our own datasets from Amazon review data (Ni et al., 2019), which include the categories Arts, Movies, Games, Home, Electronics, Tools, and Sports. For **item profile generation**, each item i includes a title h_i and an original description d_i. We leverage these two pieces of information to generate the profile, as described on the right-hand side of Eq. 4. Next, for **user profile generation**, we uniformly sample a number of interacted items from each user's behavior history as collaborative information. We then arrange the input prompt for the large language model according to Eq. 5. All prompts and instructions for user and item profile generation are provided in the code.

F.3 Details of Profile Diversification

As described in Section 3.3, we also conduct profile diversification using large language models (LLMs) to enhance the diversity of the training and test datasets, thereby improving and better evaluating the model's generalization ability across different user and item profiles. For each user or item, we perform t iterations of diversification starting from the initially generated profile. This means that we obtain the first diversified profile based on the original profile and then use this diversified profile for further diversification with the LLMs.

For reference, examples of user and item profile diversification are provided in Figure 6 and Figure 7, respectively. As illustrated in the case of user profile diversification, the profiles for the same user differ at the word level while still representing the same preferences. Such diversification enhances the quality and diversity of textual data while increasing the overall dataset size.

F.4 Cost of Generation and Diversification

We summarize the total number of tokens and the costs for utilizing LLMs for profile generation and

Table 9: Costs for profile generation and diversification, including three iterations of diversification.

Operation	#Data	#Tokens	#Cost ($)
Generation	7	189M	$\sim$114
Diversification	9	132M	$\sim$97

Table 10: Diversity improvement with profile diversification (Higher ILS indicates better diversity)

Model	Steam			Yelp		
	ILS@5	ILS@10	ILS@20	ILS@5	ILS@10	ILS@20
w/o Diversification	0.7128	0.7336	0.7573	0.5954	0.6090	0.6248
w/ Diversification	0.7268	0.7487	0.7727	0.6012	0.6136	0.6285

diversification in Table 9. The profiles for the Steam and Yelp datasets have already been generated in previous work (Ren et al., 2024); therefore, the number of profiled datasets and diversified datasets differs. The total number of tokens required to process both profile generation and diversification is approximately 322 million, costing approximately 200 dollars with the `GPT-3.5-Turbo` API, making it a cost-effective choice.

F.5 Improved Recommendation Diversity with Profile Diversification

To further demonstrate the effecitveness of profile diversification, We also conduct experiments to explore whether diversification can enhance the diversity of recommendation results. We calculated Intra-List Similarity (ILS) (Zhang and Hurley, 2008) for EasyRecLarge with and without profile diversification on the Steam and Yelp datasets. As shown in Table 10, our findings indicate that profile diversification effectively increases the diversity of the model's recommendation results.

G Details of Text-based Recommendation

G.1 Baseline Models

In this section, we provide a detailed description of the language models compared in this work.

(i) General Language Models.

- **BERT** (Devlin et al., 2018): A prominent transformer model known for its strong language understanding via bidirectional training. We use the pooled BERT output for text embedding.
- **RoBERTa** (Liu et al., 2019): An optimized BERT with dynamic masking and larger datasets. We use the final [CLS] token embedding.
- **BART** (Lewis et al., 2019): A denoising autoencoder transformer trained on corrupted text reconstruction. We apply mean pooling on the last hidden state for the text embedding.

Table 11: Details of compared language models.

Model	Pre-trained Weights (From Hugging Face)
BERT-Base	`google-bert/bert-base-uncased`
BERT-Large	`google-bert/bert-large-uncased`
BART-Base	`facebook/bart-base`
BART-Base	`facebook/bart-large`
RoBERTa-Base	`FacebookAI/roberta-base`
RoBERTa-Large	`FacebookAI/roberta-large`
SimCSE-Base	`princeton-nlp/sup-simcse-roberta-base`
SimCSE-Large	`princeton-nlp/sup-simcse-roberta-large`
BLaIR-Base	`hyp1231/blair-roberta-base`
BLaIR-Large	`hyp1231/blair-roberta-large`
GTR-Base	`sentence-transformers/gtr-t5-base`
GTR-Large	`sentence-transformers/gtr-t5-large`
BGE-Base	`BAAI/bge-base-en-v1.5`
BGE-Large	`BAAI/bge-large-en-v1.5`

(ii) Language Models for Dense Retrieval.

- **SimCSE** (Gao et al., 2021): A framework that leverages contrastive learning to generate high-quality sentence embeddings, enhancing the model's ability to discern semantic similarity between sentences.
- **GTR** (Ni et al., 2021): GTR is a generalizable T5-based dense retriever that improves retrieval tasks across various domains by overcoming limitations of traditional dual encoders.
- **BGE** (Xiao et al., 2023): A state-of-the-art family of well-trained models for general text embeddings, utilizing the English version of BGE.

(iii) Langauge Models for Recommendation.

- **BLaIR** (Hou et al., 2024a): A series of embedding models for recommendation. BLaIR learns correlations between item metadata and user feedback, improving item retrieval and recommendation.

The pre-trained weights utilized for each baseline language models are listed in Table 11. For proprietary embedding models **OpenAIv3**, we use the latest `text-embedding-3-small` and `text-embedding-3-large` from OpenAI.

G.2 Detail of Baseline Settings

Given that the experiment was conducted in a zero-shot setting, where the test data remained unseen during training for both our EasyRec and the other baselines, we directly utilized the original released parameters of these baselines from Hugging Face for comparison. Besides, for BGE, we added the recommended retrieval instruction in front of the user profiles, following the provided guidelines from the open-source code, as we found it offered better performance in zero-shot recommendations. We normalized model outputs and computed cosine similarity between user and item embeddings.