LangPTune: Optimizing Language-based User Profiles for Recommendation

Zhaolin Gao
Cornell University
Ithaca, USA
zg292@cornell.edu

Joyce Zhou
Cornell University
Ithaca, USA
jz549@cornell.edu

Yijia Dai
Cornell University
Ithaca, USA
yd73@cornell.edu

Thorsten Joachims
Cornell University
Ithaca, USA
tj36@cornell.edu

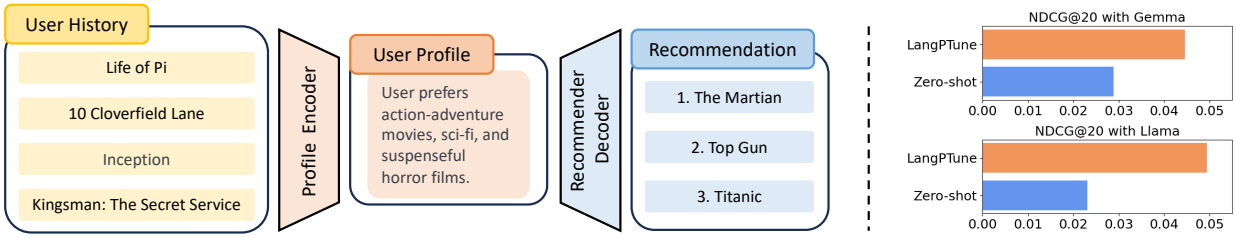


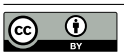
Figure 1: (Left) An illustration of LangPTune inference pipeline. The profile encoder first generates a language-based user profile, and the recommender decoder recommends based on the generated profile. (Right) Evaluation results on the Amazon-Movie-TV dataset show that LangPTune significantly outperforms zero-shot method on both Llama and Gemma models.

Abstract

Recent works have shown increasing interest in using natural language-based user profiles for recommender systems, as they offer greater transparency and interpretability compared to traditional embedding-based methods. Most existing approaches rely on zero-shot inference with large language models (LLMs) to generate these profiles, but the resulting quality remains insufficient, leading to suboptimal recommendation performance. In this paper, we present LangPTune, the first end-to-end training framework designed to directly optimize LLM-generated user profiles for recommendation tasks. By explicitly training the LLM for the recommendation objective, our approach significantly outperforms zero-shot baselines. Evaluations across training setups and benchmarks show that LangPTune not only exceeds the performance of zero-shot methods but also matches the performance of state-of-the-art embedding-based baselines. Additionally, we assess whether our training framework maintains the interpretability of user profiles, using both GPT-4 simulations and crowdworker studies.

CCS Concepts

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Reinforcement learning*.



CIKM '25, November 10–14, 2025, Seoul, Republic of Korea

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761369

Keywords

Large Language Models, Recommendation, Alignment

ACM Reference Format:
Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2025. LangPTune: Optimizing Language-based User Profiles for Recommendation. In *Proceedings of the 34th ACM Int'l Conference on Information and Knowledge Management (CIKM '25), Nov. 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3746252.3761369

1 Introduction

Accurate recommendations have become an essential part of web platforms such as multimedia streaming services, e-commerce, and social media. To power these systems, recommender system research has focused on learning embeddings of users and items to represent user preferences [15, 19, 23, 27, 32, 58, 63]. While these embeddings are computationally efficient, their inherently high dimensionality makes them difficult to interpret. This limitation poses a transparency barrier, making it difficult for the user to understand, modify, and steer the recommendations.

To overcome the fundamental transparency limitations of conventional recommender systems, previous studies have proposed and explored the use of language-based user profiles, where the user's preferences are represented as human-readable free-form text rather than dense vectors [45, 46, 48, 61, 65]. Such language-based user profiles are inherently more interpretable than vector-based profiles [20, 34], and they promise improved transparency, scrutability, and ultimately steerability of recommendation platforms [39, 51]. Furthermore, language-based profiles can provide additional features to traditional recommender systems, boosting

their performance and enabling more creative personalization affordances [41, 45, 49, 62].

Despite these advantages, prior work has not addressed how to train large language models (LLMs) to generate effective user profiles optimized for downstream recommendation. Existing approaches primarily rely on zero-shot or few-shot inference [46, 61, 65] or prompt tuning [48]. In this paper, we introduce **the first end-to-end training pipeline that optimizes LLMs to generate user profiles for the recommendation objective**. The inference process of our pipeline is illustrated in Fig. 1. Given a list of items that the user has interacted with, our goal is to produce a high-quality user profile via a profile encoder (i.e. LLM) that maximizes the downstream performance of the recommender decoder. Drawing on insights from Reinforcement Learning from Human Feedback (RLHF) [10, 68], we propose a novel approach called *Reinforcement Learning for System Optimization* (**RLSO**) that optimizes the LLM based on the feedback from the recommendation system. For each user, we sample multiple profiles from the LLM and use the performance of the downstream recommender as reward signals to optimize the LLM, as illustrated in Fig. 2. Furthermore, we jointly optimize the downstream ranking model of the recommender alongside the LLM through Contrastive Learning (CL) [8], which completes our end-to-end training framework that we call **LangPTune** (*Lang*uage-based *P*rofile *Tune*r).

We implement our framework using Mxbai [36] as the ranking model in the recommender decoder and evaluate the effectiveness of our algorithm on two different LLMs as profile encoder, Gemma-2B-it [54] and Llama-3-8B-it [55]. We compare LangPTune against a variety of baseline methods spanning different recommendation approaches on multiple public datasets. Our end-to-end training significantly outperforms other language-based profile methods and often exhibits performance comparable to the conventional embedding-based recommendation models[1]. The major contributions of this paper are summarized below:

- We introduce LangPTune, the first end-to-end training pipeline for effective generation of user profiles, with detailed theoretical derivations and intuitive explanations of our algorithm.
- We implement our framework using state-of-the-art LLMs and ranking models, demonstrating that it significantly outperforms other interpretable profile methods and exhibits performance comparable to conventional recommendation models.
- We study the effects of profile length and training dataset size, and provide a qualitative analysis of the generated user profiles.
- We conduct studies with both human participants and GPT-4 to compare the interpretability of the trained LLM against a zero-shot LLM, ensuring that interpretability is not compromised.

2 Related Work

LLM for Recommender Systems. Our research falls within the broad spectrum of using LLMs for recommendation, which can largely be categorized into two main approaches. The first approach involves directly training LLMs for item or rating prediction [6, 20, 29, 30, 38, 60, 64], or otherwise using them as system augmentation [37, 59]. With this approach, user interaction histories and item metadata are provided as inputs to the LLM, which is then trained to predict the items most relevant to the user. These works take advantage of LLM's ability to learn complex patterns for item recommendation, and generally do not prioritize transparency or explainability. The second approach leverages LLMs to generate additional user or item profiles/descriptions based on existing features in a zero-shot manner [22, 41, 49, 62]. These works demonstrate that incorporating additional features generated by LLMs could enhance the performance of conventional recommendation systems. Our work aligns with this second approach, but rather than relying solely on zero-shot inference, we aim to directly optimize the LLM to generate more effective profiles for recommendation.

Natural language-based profile methods. We focus on centering recommender systems around natural language-based user profiles, which has garnered increasing attention for enabling increased transparency, explainability, and scrutability of recommendation systems [46]. In comparison with previous work based on user interest concept sets [5, 43], natural language profiles offer increased nuance, flexibility, and affordances for user control [20, 34, 46]. There have been a number of works on generating initial profiles based on user interaction history, as well as using profiles to make recommendations. For example, Yang et al. [61] constructs user profiles based on observed user behaviors and uses them to perform personalized recommendations. Ramos et al. [48] explores prompting strategies aimed at improving the quality of LLM-generated user profiles, and examines the system's scrutability by observing how modifications to the user profile affect the resulting recommendations. Zhou et al. [65] demonstrates that generating a compact and human-readable summary often performs comparably with or better than direct LLM prediction. However, none of these approaches allow for end-to-end training to optimize profile generation.

Reinforcement Learning from Human Feedback (RLHF). Our method of training LLMs is closely related to the families of techniques for preference fine-tuning. One approach involves first training a reward model on a dataset of human preferences, which is then used to provide rewards to a downstream reinforcement learning algorithm (e.g., PPO [52, 68]). LLMs fine-tuned using this approach include GPT-N [44], Claude-N [2], and Llama-N [42]. Another approach directly optimizes the LLM using offline human preference data without fitting a reward model. This family of methods includes DPO [47], IPO [3], and KTO [17], which are simpler to implement due to the removal of the reward model. Our approach is an iterative variant of the latter approach, where we employ an offline algorithm on iteratively collected profiles, with rewards generated by the recommender decoder.

3 LangPTune: *Lang*uage-based *P*rofile *Tune*r

In this section, we formalize our encoder-decoder recommendation model and derive our end-to-end training framework.

3.1 LangPTune Model Formulation

The inference process of our pipeline is illustrated in Fig. 1. Given a list of items that the user has interacted with, our goal is to generate a high-quality user profile using a profile encoder that maximizes the downstream performance of the recommender decoder.

3.1.1 Recommender Decoder σ. The goal for the recommender decoder is to provide accurate recommendation based on the input

[1]Implementation available at https://anonymous.4open.science/r/LangPTune_a/

profile p. Specifically, given a set of I items, the recommender decoder σ generates a ranked list of I items based on the input profile p to maximize the ranking quality U. Let $i \in \{i_1, \ldots, i_I\}$ denote an item, and each item i has associated metadata $d_i \in \mathcal{D}$ such as title, description, and category. Given a user profile p and a complete set of item metadata $\mathcal{D}$, the decoder σ computes a similarity score $sim(p, d_i)$ for each item i and generates a ranked list r via sorting.

$$r = \sigma(p, \mathcal{D}) = \underset{d_i \in \mathcal{D}}{\arg\,\mathrm{sort}} \, [sim(p, d_i)] \tag{1}$$

The similarity model $sim(p, d_i)$ can take a wide range of forms, but for simplicity we focus on a model that learns a semantic embedding Φ of both profiles and items, and then computes similarity via the inner product

$$sim(p, d_i) = \Phi(p) \cdot \Phi(d_i). \tag{2}$$

However, there is a wide range of alternative similarity models (e.g. two-tower models [12, 16]) that can equally be trained with the method we will describe in Section 3.2.2. Let $f = [i_1, \ldots, i_{|f|}]$ denote the list of items the user will interact with in the future. The ranking quality is then evaluated based on the ranking r for test items in f, denoted as $U(r, f)$. This evaluation is typically performed using ranking metrics such as Mean Reciprocal Rank (MRR) or Normalized Discounted Cumulative Gain (NDCG).

A key advantage of our similarity-based decoder is its ability to scale to large collections of items, since the arg sort can be efficiently implemented with conventional ranking techniques. This overcomes the bottleneck of previous works [61, 65] that typically use an LLM as the recommender. While an LLM can directly operate on the textual profiles without further training, using zero-shot LLM-based recommenders typically implies long latency and it requires limiting to small subsets of all items due to the context length restrictions of LLMs. In contrast, our system is designed to scale across large item spaces while incorporating textual input and leveraging natural language user profiles.

3.1.2 Profile Encoder π. The profile encoder π computes a mapping from the user's interaction history to a natural-language profile. Let $h = [i_1, i_2, \ldots, i_{|h|}]$ represent a particular user's interaction history, consisting of a sequence of items, and denote the interaction history annotated with the metadata of the items as $D(h) = [d_{i_1}, \ldots, d_{i_{|h|}}]$. We model the profile encoder as a distribution over the profile space $\Delta\mathcal{P}$ from which we can efficiently generate a profile p by sampling.

$$p \sim \pi(\cdot|D(h)) \tag{3}$$

A natural question may rise regarding the necessity of utilizing a profile encoder rather than directly feeding the the metadata of items in the user interaction history, $D(h)$, to the decoder. Since the generated profile is a higher abstraction than a raw list of items, it can encapsulate more generalizable preferences and interests instead of merely reflecting specific past items. In addition, the profiles are more compact than the metadata, which enhances the interpretability of the profile and the steerability for users to modify or expand their profiles.

3.1.3 LangPTune. Combining the profile encoder π and the recommender decoder σ, we arrive at our proposed framework LangPTune. Given a user with history h, future interactions f, and item metadata $\mathcal{D}$, the encoder first generates a user profile p based on the metadata of items that are in the user interaction history, $p \sim \pi(\cdot|D(h))$. Then, the decoder generates a ranked list given the profile and item metadata, $r = \sigma(p, \mathcal{D})$. The framework's objective is to maximize recommendation performance $U(r, f)$ while ensuring that the user profile p is expressed in a well-formed natural language format.

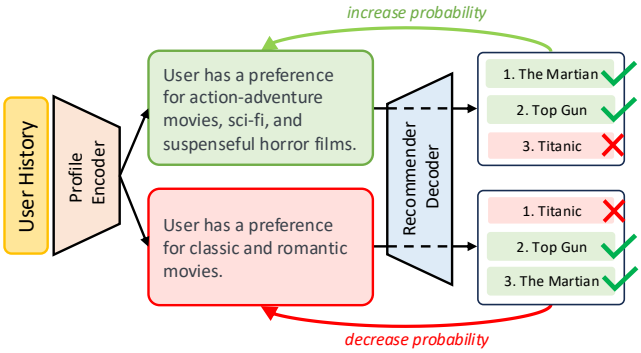


Figure 2: Reinforcement Learning for System Optimization Pipeline

3.2 LangPTune Training Algorithm

In the section, we first introduce our algorithms for training the profile encoder π and the recommender decoder σ separately. We then integrate both methods into a unified end-to-end training framework.

3.2.1 RLSO for Profile Encoder Optimization. In this section, we derive our training algorithm for profile encoder optimization which we refer to as *Reinforcement Learning for System Optimization* (**RLSO**) as illustrated in Fig. 2. Our approach is based on a contextual bandit formulation of reinforcement learning [33], which has been used to model the generation process in LLMs [18, 47]. Specifically, in the deterministic transition setting, explicit state representations are unnecessary, as they are isomorphic to the sequence of actions. In this view, an entire action sequence corresponds to a single "arm" in a bandit problem, with an action space that grows exponentially with the sequence length. We define the context as $D(h)$ and the action as $p \sim \pi(\cdot|D(h))$, sampled from the policy/encoder π.

To optimize the encoder π such that it maximizes the performance of the recommender decoder measured by U, we aim to solve the following Reinforcement Learning (RL) problem constrained by Kullback–Leibler (KL) divergence at each iteration of our algorithm:

$$\pi_{t+1} = \arg\max_{\pi} \mathbb{E}_{h,f,p\sim\pi(\cdot|D(h))} \Big[U(\sigma(p, \mathcal{D}), f) - \frac{1}{\eta} \mathrm{KL}\left(\pi(\cdot|D(h)) || \pi_t(\cdot|D(h))\right) \Big] \tag{4}$$

where η is a parameter that controls the degree of KL regularization, and π_t is the encoder from previous iteration. Intuitively, the objective is to maximize the metric of the recommender system, $U(\sigma(p, \mathcal{D}), f)$, while staying close to π_t by minimizing the KL divergence between π and π_t. The KL divergence is typically used in RL to prevent reward hacking. In this context, it serves to preserve

the readability of the user profiles by staying close to the encoder π_t of the previous iteration.

Now we start to derive the loss function for the above objective. From Ziebart et al. [67], there exists a closed-form solution to the above minimum relative entropy problem (Eq. 4) [21]:

$$\forall h, f, p : \pi_{t+1}(p|D(h)) = \frac{\pi_t(p|D(h)) \exp(\eta U(\sigma(p, \mathcal{D}), f))}{Z(h, f)}$$
$$Z(h, f) = \sum_p \pi_t(p|D(h)) \exp(\eta U(\sigma(p, \mathcal{D}), f)) \quad (5)$$

where $Z(h, f)$ is a partition function to ensure that $\pi_{t+1}(\cdot|D(h))$ remains as a valid probability distribution that sums to 1. Based on Rafailov et al. [47], we can invert Eq. 5 and write the ranking quality as a function of π:

$$\forall h, f, p : U(\sigma(p, \mathcal{D}), f) = \frac{1}{\eta}\left(\ln(Z(h, f)) + \ln\left(\frac{\pi_{t+1}(p|D(h))}{\pi_t(p|D(h))}\right)\right) \quad (6)$$

Note that the partition function $Z(h, f)$ does not depend on profile p. Therefore, we can sample another profile $p' \sim \pi(\cdot|D(h))$ and take the difference of the above expression across two profiles to eliminate the partition function:

$$\forall h, f, p, p' : U(\sigma(p, \mathcal{D}), f) - U(\sigma(p', \mathcal{D}), f) = \frac{1}{\eta}\left(\ln\left(\frac{\pi_{t+1}(p|D(h))}{\pi_t(p|D(h))}\right) - \ln\left(\frac{\pi_{t+1}(p'|D(h))}{\pi_t(p'|D(h))}\right)\right) \quad (7)$$

Eq. 7 represents the condition that the solution to the argmax problem in Eq. 4 should satisfy. Therefore, since we are trying to solve for π_{t+1}, following Gao et al. [18], we could directly regress the right part of Eq. 7 to the left part with squared regression which becomes our loss function:

$$\left(U(\sigma(p, \mathcal{D}), f) - U(\sigma(p', \mathcal{D}), f) - \frac{1}{\eta}\left(\ln\left(\frac{\pi_{t+1}(p|D(h))}{\pi_t(p|D(h))}\right) - \ln\left(\frac{\pi_{t+1}(p'|D(h))}{\pi_t(p'|D(h))}\right)\right)\right)^2 \quad (8)$$

To optimize the profile encoder with the above squared loss objective given an interaction history h, we begin by sampling two profiles, $p \sim \pi_t(\cdot|D(h))$ and $p' \sim \pi_t(\cdot|D(h))$, from π_t. Next, we obtain two ranked lists based on the sampled profiles, $r = \sigma(p, \mathcal{D})$ and $r' = \sigma(p', \mathcal{D})$, and calculate the corresponding ranking qualities, $U(r, f)$ and $U(r', f)$. The optimization of π_t is illustrated in Fig. 2 which is guided by the relative difference between these ranking qualities. Intuitively, if profile p is better than p', i.e. $U(\sigma(p, \mathcal{D}), f) > U(\sigma(p', \mathcal{D}), f)$, we aim to optimize the encoder to increase the likelihood of generating p over p', i.e. increasing $\ln\left(\frac{\pi_{t+1}(p|D(h))}{\pi_t(p|D(h))}\right) - \ln\left(\frac{\pi_{t+1}(p'|D(h)}{\pi_t(p'|D(h))}\right)$. We refer to this training algorithm as *Reinforcement Learning for System Optimization* (**RLSO**), since it trains the encoder to produce profiles that maximize the performance of the recommendation *system* (i.e. the decoder σ).

Intuitive Interpretation of RLSO. From the above derivations, solving Eq. 8 optimally would recover the optimal solution in Eq. 7. Aggregate over p' implies that there must exists a p-independent function $c(h, f)$ such that:

$$\forall h, f, p : \frac{1}{\eta} \ln \frac{\pi_{t+1}(p|D(h))}{\pi_t(p|D(h))} = U(\sigma(p, \mathcal{D}), f) + c(h, f) \quad (9)$$

Algorithm 1 LangPTune Optimization

Required: ranking quality function U, parameter η, similarity function $sim(\cdot, \cdot)$
Initialize encoder π and decoder σ which uses Φ as its embedding model.
for $k = 0$ to $K - 1$ **do**
  $\Phi_0 \leftarrow \Phi$
  for $j = 0$ to $J - 1$ **do**
    Collect data $\epsilon_j = \{h, p, B\}$ where $p \sim \pi(\cdot|D(h))$, B is a list of items with one positive item i_+.
    Optimize for Eq. 10 to obtain Φ_{j+1}.
  end for
  $\Phi \leftarrow \Phi_J$
  $\pi_0 \leftarrow \pi$
  for $t = 0$ to $T - 1$ **do**
    Collect data $\xi_t = \{h, f, p, p'\}$ where $p, p' \sim \pi_t(\cdot|D(h))$.
    Solve the squared regression in Eq. 8 to obtain π_{t+1}.
  end for
  $\pi \leftarrow \pi_T$
end for

Rearranging the terms, we have:

$$\forall h, f, p : \pi_{t+1}(p|D(h)) = \pi_t(p|D(h)) \exp(\eta U(\sigma(p, \mathcal{D}), f) + \eta c(h, f)) \propto \pi_t(p|D(h)) \exp(\eta U(\sigma(p, \mathcal{D}), f))$$

Intuitively, the probability of generating a profile at the next iteration, $\pi_{t+1}(p|D(h))$, is proportional to the performance of the profile on the recommender decoder, $\exp(\eta U(\sigma(p, \mathcal{D}), f))$. Better recommendation performance means the encoder is more likely to generate that profile at the next iteration. Note that the above update procedure also recovers the Natural Policy Gradient update with the softmax parametrization [1], which has a fast convergence rate of $O(1/T)$.

3.2.2 CL for Recommender Decoder Optimization. In our implementation, the text-based recommender decoder σ contains a text embedding model denoted by Φ. This embedding model is optimized using the standard Contrastive Learning (**CL**) framework [8], leveraging the InfoNCE loss [56] to enhance its performance. Given a user profile p, a batch of items B with one positive item $i_+ \in f$, the objective is to maximize the similarity between the user embedding $\Phi(p)$ and the positive item embedding $\Phi(d_{i_+})$, while minimizing the similarity between $\Phi(p)$ and all other items in the batch (other items are considered as negative items):

$$\Phi = \arg\min_{\Phi} \sum_{p,B} -\ln \frac{\exp(sim(\Phi(p), \Phi(d_{i_+})))}{\sum_{i \in B} \exp(sim(\Phi(p), \Phi(d_i)))} \quad (10)$$

By optimizing this contrastive loss, the model is trained to discriminate between relevant and irrelevant items, thereby improving the quality of recommendations generated for each user.

3.2.3 Joint Optimization. Combining the profile encoder optimization and recommender decoder optimization, we arrive at our joint optimization pipeline for LangPTune. The pseudocode is provided in Algorithm 1. Our approach alternates between optimizing the

Title: Outlander: Season 5 [DVD]
Description: (average rating: 4.9) The fifth season of Outlander sees a continuation of Claire and Jamie's fight to protect those they love, as they navigate the trials and tribulations of life in colonial America. The Frasers strive to flourish within a society which is unwittingly marching toward Revolution, and Jamie must now defend the home they have built together at Fraser's Ridge while Claire seeks to put her own skills and medical expertise to use in keeping her family together and safe from harm. Meanwhile, Brianna and Roger MacKenzie struggle to find their respective places in this world and chase away the shadow cast over their lives by Stephen Bonnet as they raise their son in this brave new world. For the Frasers and their family, "home" is more than simply a site on which they live, it is the place in which they are laying the foundations for the rest of their lives.
Category: Movies & TV, Studio Specials, Sony Pictures Home Entertainment, All Sony Pictures Titles
Price: 17.61

The user has rated (out of 5.0) and reviewed following movies and TV shows arranged chronologically from the oldest (top) to the newest (bottom). Please provide a high-level summary of the user preference in detail.

Figure 3: Examples of item metadata (top) and the LLM prompt (bottom).

Dataset	#Sessions Train/Val/Test	#Items	#Interactions
Amazon-Movie-TV	14,698/1,392/1,392	10,533	87,410
Amazon-Books	4,404/361/361	4,819	25,630

Table 1: Dataset statistics

two models over K iterations. In each iteration, we first optimize the recommender decoder using a contrastive learning (**CL**) objective across J batches, followed by optimizing the profile encoder via reinforcement learning for system optimization (**RLSO**) across T batches. This alternating process ensures that both components are continuously refined, leading to a more cohesive and effective model for personalized recommendations.

4 Experiments

In this section, we present a suite of experiments on various datasets and LLMs against multiple baselines to validate the effectiveness of LangPTune under different settings.

4.1 Dataset

We evaluate LangPTune on the Amazon-Movie-TV and Amazon-Books datasets [25] where the item metadata includes title, description, average rating, category, and price. In addition, we also include the user reviews of the items as additional inputs to the LLM. Examples of the item metadata and the LLM prompt are shown in Figure 3. We filter out sessions and items that have less than 5 interactions. To satisfy the context length of the LLM and embedding model, we additionally filter out items with metadata that are longer than 512 tokens and sessions with a history longer than 1024 tokens. The length of the interaction history is 4 and the number of items to predict in the future is 1 (i.e. $|h| = 4$ and $|f| = 1$), where it is guaranteed that all five items are distinct from each other. The statistics of the dataset are shown in Table 1. Our dataset size is comparable to popular reasoning datasets including MATH [24] and GSM8K [11] that are frequently used for RL training and evaluation of large language models. To evaluate top-k ranking performance for each model, we adopt three widely used metrics: Normalized Discounted Cumulative Gain (NDCG), Mean Reciprocal Rank (MRR), and Recall [19, 40, 53].

4.2 Baselines

4.2.1 Interpretable Profile Baselines. We compare LangPTune with variants that reflect interpretable user profile methods from prior works. **LangPTune-0** conducts zero-shot inference using only pre-trained embedding and language models without additional training. This method represents the pure zero-shot approach described by Zhou et al. [65]. **LangPTune-CL** focuses on training the recommender decoder using InfoNCE, while keeping the profile encoder fixed. This baseline aligns with previous profile-based approaches in Yang et al. [61] and Ramos et al. [48]. To test the effectiveness of the profile encoder optimization, we include **LangPTune-RLSO** which only trains the encoder with RLSO without optimizing the decoder.

4.2.2 Black-Box and No-Profile Baselines. We also compare against a variety of other baseline types to contextualize the performance.

No Profile methods do not produce any profile for the user and serve as a baseline to assess task difficulty. These include **Random** which generates a randomly shuffled ranked list of all items, and **Most Popular (MP)** which ranks all items in descending order of popularity, based on observations from the training set and user interaction history from the validation/test set. These baselines serve as benchmarks for the task's difficulty, offering a point of reference for evaluating the performance of other methods.

Black-Box Profile methods[2] rely on user and item interaction histories without leveraging additional item metadata. These methods apply collaborative filtering or other embedding-based approaches, producing user embeddings that are uninterpretable, hence referred to as black-box profiles. **NMF** [35] (Non-negative Matrix Factorization) decomposes user-item interaction data into latent factors and ranks the items based on the predictions of missing preferences. **LightGCN** [23] uses Graph Convolutional Networks (GCN) [31] with Bayesian Personalized Ranking (BPR) loss [50] to learn robust user/item embeddings. **MCL** [19] builds on top of LightGCN using sample mining and a mixed-centric loss. **BERT4REC** [53] uses masked language modeling (MLM) objective in BERT [14] to learn user/item embeddings. **LLM2BERT4REC** [22] initializes then trains a BERT4REC model with embeddings obtained from a text embedding model using item descriptions.

[2]For all Black-Box Profile baselines, we use both the training set and the user interaction history items of the validation/test set during training.

LLM	Method	Amazon-Movie-TV			Amazon-Books		
		Recall@20	MRR@20	NDCG@20	Recall@20	MRR@20	NDCG@20
Gemma	LangPTune-0	0.0643	0.0189	0.0289 (+54.3%*)	0.3397	0.1269	0.1743 (+57.0%*)
	LangPTune-CL	**0.0863**	0.0317	0.0436 (+2.29%)	0.3587	0.1516	0.1984 (+37.9%*)
	LangPTune-RLSO	0.0746	0.0253	0.0364 (+22.5%*)	0.4212	0.1872	0.2400 (+14.0%*)
	LangPTune	0.0859	**0.0327**	**0.0446**	**0.4375**	**0.2151**	**0.2736**
Llama	LangPTune-0	0.0535	0.0146	0.0231 (+113.9%*)	0.3370	0.1285	0.1753 (+36.5%*)
	LangPTune-CL	0.0938	0.0319	0.0456 (+8.33%*)	0.3995	0.1539	0.2099 (+14.0%*)
	LangPTune-RLSO	0.0763	0.0289	0.0377 (+31.0%*)	0.3913	0.1728	0.2224 (+7.60%*)
	LangPTune	**0.0970**	**0.0358**	**0.0494**	**0.4212**	**0.1860**	**0.2393**

Table 2: Results for Interpretable Profile Methods. The best-performing methods for each metric, LLM and dataset are highlighted in bold and the next best methods are underlined. The relative improvements of LangPTune on NDCG@20 w.r.t. the baseline methods are shown in brackets. Asterisks denote statistically significant improvements based on paired t-test ($\alpha = 0.05$).

4.3 Implementation Details

We tune each baseline individually on the dataset to achieve the best performance on the validation set, and evaluate on the test set. We experiment with two different LLMs for the profile encoder, Gemma-2B-it[3] [54] and Llama-3-8B-it[4] [55], to evaluate our framework across different LLM sizes and architectures. We set a maximum profile length of 512 tokens. Following Zhu et al. [66] and Gao et al. [18], we train the last four layers of the Llama-3-8B-it while keeping other layers frozen, and train Gemma-2B-it with Low-Rank Adaptors (LoRA) [26] with rank 1024 and alpha 2048. For CL, we use a learning rate of 3e-7, a batch size of 256, and a weight decay of 1e-6. For RLSO, we adopt a batch size of 64 and the same weight decay. We use a learning rate of 1e-7 for Gemma-2B-it and 3e-7 for Llama-3-8B-it with a linear decay. Training is performed for 4 epochs over the training dataset with $\eta = 1$ and $K = 5$. For Gemma-2B-it, we set $T = 200$ and $J = 800$, while for Llama-3-8B-it, we use $T = 500$ and $J = 2500$.

Llama-3-8B-it experiments are trained on 8 A6000 GPUs for two days, and Gemma-2B-it experiments are trained on the same hardware for one day. For all methods that require a text embedding model, we use the same Mxbai[5] [36] embedding model, and perform full-parameter training on 8 A6000 GPUs with each iteration taking one hour.

In our implementation, we use NDCG as the ranking quality function U as it is bounded between 0 and 1. In addition, we standardize the values of U within each batch to zero-mean and unit-variance to further stabilize the learning of the policy. To ensure that π remains close to π_0, we apply an additional KL penalty to the U:

$$U(\sigma(p, \mathcal{D}), f) - \gamma \left(\ln \pi(p|D(h)) - \ln \pi_0(p|D(h))\right) \qquad (11)$$

where γ is a hyperparameter that controls the penalty [18, 28]. Furthermore, to ensure that the generations terminate within the maximum generation length, following Huang et al. [28], we penalize any generation that exceeds this length by setting $U(\sigma(p, \mathcal{D}), f)$ to a small fixed constant, Γ. We set $\Gamma = 0$ for all experiments. For Gemma-2B-it, we use $\gamma = 0.2$, while for Llama-3-8B-it, we set $\gamma = 0$.

[3]HuggingFace Model Card: google/gemma-2b-it
[4]HuggingFace Model Card: meta-llama/Meta-Llama-3-8B-Instruct
[5]HuggingFace Model Card: mixedbread-ai/mxbai-embed-large-v1

4.4 Performance Analysis

4.4.1 Performance Comparison among Interpretable Profile Methods. The results for the interpretable profile methods are shown in Table 2 where the best-performing methods for each metric, base model and dataset are highlighted in bold and the second best methods are underlined. The relative improvements of LangPTune on NDCG@20 w.r.t the baselines are shown in bracket with asterisk denotes statistically significant improvement based on paired t-test ($\alpha = 0.05$). While both LangPTune-CL and LangPTune-RLSO show substantial improvements over the non-learning approach LangPTune-0, **LangPTune achieves the highest performance compared to other interpretable profile methods in terms of NDCG and MRR**. This demonstrates LangPTune's effectiveness: CL ensures user profiles and item metadata align in the embedding space, while RLSO refines the profile encoder, generating user profiles that better match target items, thereby improving recommendation accuracy. An example of generated user profiles is shown in Figure 4, demonstrating qualitatively that LangPTune is able to generate additional details compared to the base model while remaining interpretable. Notably, the model's ability to autonomously enrich the user profiles with relevant details—without explicit supervision or ground-truth profiles—highlights a significant advancement. This suggests that the LLM effectively explores the profile space and learns to incorporate valuable information independently without any human guidance.

4.4.2 Performance Comparison with Black-Box Baselines. We compare LangPTune with other state-of-the-art black-box recommendation methods in Table 3. For LangPTune, we select either the Llama or Gemma variant, depending on which yields the higher performance metric on the validation set. Even though interpretability adds another objective and constraint, LangPTune outperforms all black-box methods on the Amazon-Books dataset, and only LightGCN and MCL manage to show better performance on Amazon-Movie-TV dataset. We attribute the excellent performance of LangPTune on Amazon-Books to the rich and informative textual descriptions in this dataset, which allows LangPTune to better capture user preferences. Additionally, it is possible that the LLMs

User Interation History: Life of Pi, 10 Cloverfield Lane, Inception, Kingsman: The Secret Service

Zero-shot Generation:
story is important: the user appreciates a good story, as evident from their reviews of "" life of pi "" and "" 10 cloverfield lane "". they mention the importance of a great story, message, and script.
action and suspense: the user enjoys action - packed and suspenseful movies, as seen in their reviews of "" 10 cloverfield lane "", "" inception "", and "" kingsman : the secret service "". they appreciate films that keep them engaged and guessing until the end.
visually impressive: the user is impressed by visually stunning movies, as evident from their review of "" life of pi "", which they praised for its great visuals.

LangPTune Generation:
Storytelling: The user enjoys complex storylines with twists and surprises. They appreciate movies that explore themes of adventure, discovery, and suspense. The user particularly liked the unexpected connections and surprises in "Life of Pi" and the twists in "10 Cloverfield Lane".
Suspense and Thrills: The user enjoys suspenseful movies that keep them guessing until the end. They highly rated "10 Cloverfield Lane" for its ability to maintain tension and uncertainty throughout the film.
Visuals: The user appreciates visually stunning movies with groundbreaking effects. They praised the visuals in "Life of Pi" and might have appreciated the action sequences and spy-thriller elements in "Kingsman: The Secret Service".

Figure 4: Example of the first three points from the profiles generated by Llama-3-8B-it before and after trained by LangPTune. The highlighted text represents the additional details generated after training.

Type	Method	Amazon-Movie-TV			Amazon-Books		
		Recall@20	MRR@20	NDCG@20	Recall@20	MRR@20	NDCG@20
No Profile	Random	0.0019	0.0003	0.0006 (+8133%*)	0.0042	0.0007	0.0014 (+19443%*)
	MP	0.0129	0.0015	0.0039 (+1167%*)	0.0776	0.0714	0.0729 (+275.3%*)
Black-Box Profile	NMF	0.0625	0.0269	0.0350 (+41.1%*)	0.4238	0.1766	0.2323 (+17.8%*)
	LightGCN	0.1009	0.0387	0.0525 (-5.90%*)	0.4072	0.2145	0.2586 (+5.80%*)
	MCL	0.1078	0.0417	0.0563 (-12.3%*)	0.3906	0.1616	0.2134 (+28.2%*)
	BERT4REC	0.0758	0.0280	0.0384 (+28.6%*)	0.3006	0.1662	0.2235 (+22.4%*)
	LLM2BERT4REC	0.0783	0.0312	0.0423 (+16.8%*)	0.3572	0.2014	0.2343 (+16.8%*)
Interpretable Profile	LangPTune	0.0970	0.0358	0.0494	0.4375	0.2151	0.2736

Table 3: Results using Different Recommender Types. The relative improvements of LangPTune on NDCG@20 w.r.t. the baseline methods are shown in brackets. Asterisks denote statistically significant differences based on paired t-test ($\alpha = 0.05$).

have a deeper understanding of books compared to movies, resulting in more accurate profiles on Amazon-Books. It is important to note that we do not necessarily expect interpretable, profile-based methods to outperform all embedding-based approaches, as embedding spaces inherently offer greater representational capacity than token-based profiles.

Overall, these results shows that LangPTune can be highly competitive even with methods that lack the interpretability and steerability of language-based profiles. These qualities are crucial for enhancing user experience and system control, making LangPTune more advantageous for applications that prioritize understanding and tailoring user preferences.

4.5 Ablation Analysis

4.5.1 Profile Length. The length of the profile plays an important role as it controls the width of the information bottleneck in the pipeline, where a shorter length requires stronger (and probably lossy) compression. We ablate the maximum generation length from 512 to 64 using Llama-3-8B-it on Amazon-Movie-TV. In addition, to quantify the amount of information loss through the bottleneck, we incorporate two additional baselines that have no bottleneck. **Zero-shot Embedding Model without Profile Encoder (NoProfile-0)** uses the same pre-trained Mxbai embedding model to generate user and item embeddings directly from the user interaction history with a list of item metadata and individual item metadata respectively. This approach is analogous to our pipeline illustrated in Fig. 1, but without the profile encoder π and the user profile p. **CL Embedding Model (NoProfile-CL)** is analogous to **NoProfile-0**, but the embedding model is trained with InfoNCE loss in Eq. 10.

The results are shown in Table 4. We observe an almost monotonic increase in performance as the profile length increases, with the rate of improvement accelerating at shorter profile lengths. Profiles of length 256 and 512 show comparable performance, indicating that excessively long profiles are not necessary for optimal results. Beyond a certain length, increasing the profile size does not yield further improvements, as performance tends to plateau.

Method	Profile Length	Recall@20	MRR@20	NDCG@20
LangPTune	64	0.0604	0.0201	0.0319
	128	0.0901	0.0313	0.0444
	256	0.0929	0.0352	0.0482
	512	0.0970	0.0358	0.0494
NoProfile-0	N/A	0.0760	0.0261	0.0372
NoProfile-CL	N/A	0.1012	0.0408	0.0543

Table 4: Ablation of Profile Length on Amazon-Movie-TV.

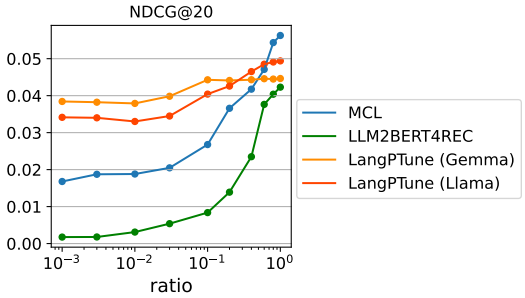


Figure 5: Ablation of Train Data Ratio on Amazon-Movie-TV.

Profile Type	Human	GPT-4
NoProfile-0	57.11%	75.56%
LangPTune-0	54.67%	73.33%
LangPTune	**58.89%**	**76.67%**

Table 5: Interpretability task average accuracy rates with human participants and GPT-4 responses. Highest accuracy per respondent type highlighted in bold.

Compared to NoProfile-CL, we expected that the requirement to produce a language-based profile is a bottleneck that leads to marginally lower performance. NoProfile-CL avoids the bottleneck in our pipeline by directly utilizing the list of item metadata without incurring information loss. It is important to note that interpretable profile methods inherently experience some degree of information loss due to the generally shorter and more concise nature of profiles. Despite the slight reduction in performance, our approach offers significant advantages in terms of readability and interpretability, which we will demonstrate in Section 4.6, as the generated profiles are far more understandable than a raw list of item metadata.

4.5.2 Training Dataset Size. To assess the sensitivity of the methods to the amount of available training data, we conducted an ablation study by gradually reducing the training data from 100% down to 0.1% of the full training set size for Amazon-Movie-TV. The results are shown in Fig. 5. MCL and LLM2BERT4REC experience significant performance degradation with reduced training data, as these methods require an explicit embedding for each item. In contrast, LangPTune is less affected, as it operates directly on text-based user profiles and item metadata. Their performance remains more consistent across the different dataset sizes. This suggests that interpretable profile methods may be particularly suitable for cold-start scenarios where there is little training data.

Profile/Text Type	Avg Char #	Avg Word #	Avg Duration (sec)
(Answer Options)	701	108	-
NoProfile-0	2983	465	34.76
LangPTune-0	1598	292	28.77
LangPTune	1730	256	29.15

Table 6: Interpretability task duration for human participants with average text lengths ($|z| > 3$ outliers filtered out)

4.6 Interpretability Evaluation

Ideally, LangPTune should preserve the interpretability of the base model, ensuring that its generated profiles remain human-readable rather than generating a "secret code" that only the recommender decoder σ can understand, as observed in the training of DeepSeek-R1-Zero [13]. To verify that LangPTune maintains interpretability of the generated profiles, we conduct a GPT-4 and human crowdworker study.

4.6.1 Dataset and Baselines. We compare the profiles generated by **LangPTune** against two baselines: **LangPTune-0**, where profiles are generated through zero-shot inference using an LLM, and **NoProfile-0**, where the profile is a raw list of item metadata drawn from the user's interaction history. We utilize 450 items from the Amazon-Movie-TV test set and employ Llama-3-8B-it for LangPTune and LangPTune-0. Either a human or GPT-4 (`gpt-4-0613`) is presented with a profile text and asked to predict one of the two movies that the user would watch next. The two-movie pool for each user consists of the ground truth movie randomly paired with another movie from the test set. The negative movie option for each user remains consistent across profile types. This experiment setup allows us to quantitatively assess the interpretability and usefulness of the generated profiles, since better profiles should lead to more accurate predictions of the next movie by humans and general-purpose language models (e.g. GPT-4) other than the recommender decoder σ.

4.6.2 GPT-4 and Human Study Details. To mitigate the possible position bias and hallucination in GPT-4 evaluations [4, 9], we randomly shuffle the order of two candidate movies and prompt GPT-4 to first give an explanation of the choice before outputting the final choice. The randomization ensures that any potential preference for the first or second option is distributed evenly across the dataset, effectively neutralizing systematic position-based influences. The explanation encourages the model to engage in a more thorough analysis of the given information before making a decision.

Human participants ($n = 150$, recruited via Amazon Mechanical Turk [7]) are first presented with an informed consent form with a brief task description. To maintain experimental integrity and balance, we implement a systematic approach to profile assignment. Each participant receives three profiles of each format in a randomized sequence. We ensure that no participant receives profiles from the same user, and all three profile formats generated for each user interaction history are distributed exactly once across participants. We give participants incentive to answer correctly by giving bonus payment for each correct answer.

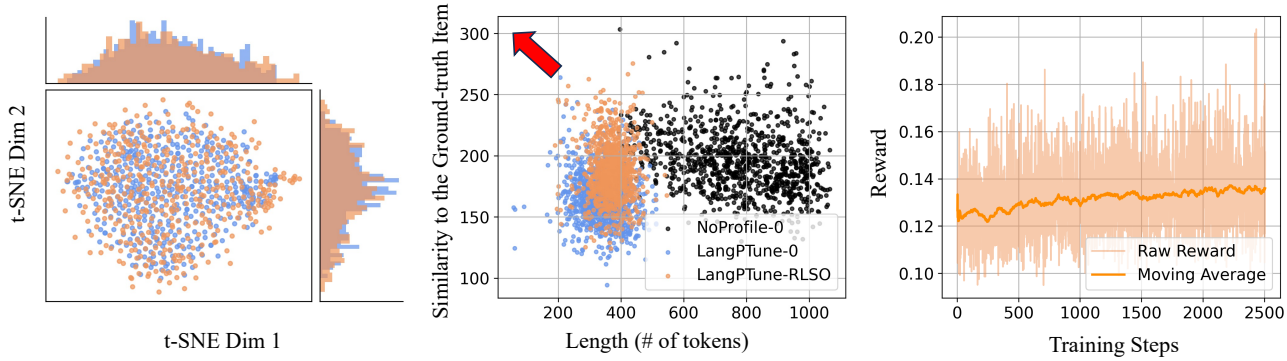


Figure 6: Visualizations with Llama-3-8B-it on Amazon-Movie-TV dataset. (Left) Visualization of the embeddings of profiles from LangPTune-0 and LangPTune-RLSO using Mxbai and t-SNE. (Middle) Length vs. Similarity to the ground-truth item for the profiles from NoProfile-0, LangPTune-0, and LangPTune-RLSO. Red arrow indicates the optimal direction. (Right) Reward for LangPTune-RLSO during training.

4.6.3 Results. We compare human and GPT-4 accuracy rates across the three different profile types in Table 5. We find that both human participants and GPT-4 achieve higher accuracy on this inference task with LangPTune profiles than with LangPTune-0 profiles. Although the difference between LangPTune and LangPTune-0 on human survey responses is not significant via two-sided McNemar's test with exact Bernoulli p-value ($p = 0.201$), the ordinal alignment of the human and GPT-4 results suggests that the profiles generated by LangPTune are at least as interpretable as those generated by LangPTune-0. It is important to highlight that we only compare LangPTune profiles against full item metadata with length 5. We speculate that if LangPTune were adapted to generate profiles for longer item interaction histories, the difference in interpretability would become much larger - the length and readability of a generated profile may stay relatively steady, while a full item history listing continues to grow longer and more difficult to read.

We also compare the average profile lengths and response durations for human participants in Table 6. Although the performance difference between LangPTune and NoProfile-0 on human surveys is smaller ($p = 0.624$), NoProfile-0 has much longer texts and requires more time to complete, indicating that its raw format imposes a higher cognitive load on participants. In contrast, LangPTune profiles are more concise while preserving the essential information needed for accurate inference.

4.7 Additional Analysis

To gain deeper insights into our pipeline, we present three visualizations based on experiments with Llama-3-8B-it using the Amazon-Movie-TV dataset, as shown in Fig. 6. These visualizations focus on LangPTune-0 and LangPTune-RLSO, as neither involves training the embedding model. This allows us to use the same zero-shot Mxbai embedding model to extract their profile embeddings, ensuring a consistent basis for comparison. We first visualize the profile embeddings generated by LangPTune-0 and LangPTune-RLSO, with t-SNE[57] for dimensionality reduction. The results indicate that LangPTune-RLSO produces more detailed profiles, as reflected in its broader distribution. We also compare the number of tokens in profiles from NoProfile-0 (i.e. a raw list of item metadata), LangPTune-0, and LangPTune-RLSO against their similarity to ground-truth items. Similarity is measured via the dot product of embeddings generated by Mxbai between the profile and the item. While LangPTune-RLSO slightly increases profile length, it significantly enhances similarity to the ground-truth items, reaching levels comparable to NoProfile-0. Finally, we examine the reward trajectory during LangPTune-RLSO training. The results show a steady and consistent enhancement in recommendation performance without any performance rollbacks.

5 Conclusions and Discussions

We introduce LangPTune, the first end-to-end pipeline that optimizes LLMs to generate user profiles for recommendation. By combining Reinforcement Learning for System Optimization (RLSO) with contrastive learning (CL), our approach outperforms prior language-based profile methods and matches black-box recommendation models while maintaining interpretability. Human and GPT-4 evaluations confirm that LangPTune generates readable and informative profiles. Our model is designed for practical scalability, low latency, and cost efficiency where our choice to use a similarity-based decoder has great practical advantages compared to using an LLM as the decoder. This design requires only a single inference call per user—through both the LLM and the embedding model. The inference call does not lie on the latency path of serving recommendation results, making our method at least as scalable as previous LLM-based recommendation systems. For the future works, there are a number of immediate research questions. First, while we evaluated interpretability, it would also be interesting to design user studies that evaluate steerability through manual changes to the user profiles. Second, on the technical side, instead of only focusing on user profile generation with clear context lengths, it would be interesting to adapt our pipeline to accommodate longer interaction histories or to obtain better natural-language item descriptions.

Acknowledgments

This research was supported in part by the Graduate Fellowships for STEM Diversity (GFSD) and LinkedIn-Cornell Grant, as well as NSF Awards IIS-2312865 and OAC-2311521.

GenAI Usage Disclosure

The profile encoder in our proposed LangPTune framework is implemented using publicly available large language models (LLMs), including Gemma-2B-it and Llama-3-8B-it. These models are fine-tuned using our custom reinforcement learning and contrastive learning procedures for the recommendation task. The recommender decoder uses a pretrained text embedding model (Mxbai) without GenAI generation capabilities. GPT-4 was used to simulate user understanding and assess profile interpretability as part of our evaluation methodology, in accordance with our interpretability user study.

References

[1] Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. 2020. On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift. arXiv:1908.00261 [cs.LG] https://arxiv.org/abs/1908.00261

[2] Anthropic. 2024. Introducing the next generation of Claude. https://www.anthropic.com/news/claude-3-family

[3] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A General Theoretical Paradigm to Understand Learning from Human Preferences. arXiv:2310.12036 [cs.AI] https://arxiv.org/abs/2310.12036

[4] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022. Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073 [cs.CL] https://arxiv.org/abs/2212.08073

[5] Krisztian Balog, Filip Radlinski, and Shushan Arakelyan. 2019. Transparent, Scrutable and Explainable User Models for Personalized Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 265–274. https://doi.org/10.1145/3331184.3331211

[6] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys '23)*. ACM. https://doi.org/10.1145/3604915.3608857

[7] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling. 2016. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality data? In *Methodological Issues and Strategies in Clinical Research* (4th ed.), Alan E. Kazdin (Ed.). American Psychological Association, 133–139. https://doi.org/10.1037/14805-009

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. arXiv:2002.05709 [cs.LG] https://arxiv.org/abs/2002.05709

[9] Cheng-Han Chiang and Hung yi Lee. 2023. Can Large Language Models Be an Alternative to Human Evaluations? arXiv:2305.01937 [cs.CL] https://arxiv.org/abs/2305.01937

[10] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 4302–4310.

[11] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168 [cs.LG] https://arxiv.org/abs/2110.14168

[12] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) *(RecSys '16)*. Association for Computing Machinery, New York, NY, USA, 191–198. https://doi.org/10.1145/2959100.2959190

[13] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL] https://arxiv.org/abs/1810.04805

[15] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. 2020. Simplify and Robustify Negative Sampling for Implicit Collaborative Filtering. In *Advances in Neural Information Processing Systems*.

[16] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *Proceedings of the 24th International Conference on World Wide Web* (Florence, Italy) *(WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 278–288. https://doi.org/10.1145/2736277.2741667

[17] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. KTO: Model Alignment as Prospect Theoretic Optimization. arXiv:2402.01306 [cs.LG] https://arxiv.org/abs/2402.01306

[18] Zhaolin Gao, Jonathan D. Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, J. Andrew Bagnell, Jason D. Lee, and Wen Sun. 2024. REBEL: Reinforcement Learning via Regressing Relative Rewards. arXiv:2404.16767 [cs.LG] https://arxiv.org/abs/2404.16767

[19] Zhaolin Gao, Zhaoyue Cheng, Felipe Perez, Jianing Sun, and Maksims Volkovs. 2022. MCL: Mixed-Centric Loss for Collaborative Filtering. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2339–2347. https://doi.org/10.1145/3485447.3512106

[20] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2023. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). https://doi.org/10.48550/arXiv.2203.13366 arXiv:2203.13366 [cs].

[21] Peter D. Grünwald and A. Philip Dawid. 2004. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *The Annals of Statistics* 32, 4 (Aug. 2004). https://doi.org/10.1214/009053604000000553

[22] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging Large Language Models for Sequential Recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys '23)*. ACM. https://doi.org/10.1145/3604915.3610639

[23] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. arXiv:2002.02126 [cs.IR] https://arxiv.org/abs/2002.02126

[24] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. arXiv:2103.03874 [cs.LG] https://arxiv.org/abs/2103.03874

[25] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024. Bridging Language and Items for Retrieval and Recommendation. *arXiv preprint arXiv:2403.03952* (2024).

[26] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685 [cs.CL] https://arxiv.org/abs/2106.09685

[27] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*. 263–272. https://doi.org/10.1109/ICDM.2008.22

[28] Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. 2022. The 37 Implementation Details of Proximal Policy Optimization. In *ICLR Blog Track*. https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/ https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/.

[29] Jianchao Ji, Zelong Li, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Juntao Tan, and Yongfeng Zhang. 2023. GenRec: Large Language Model for Generative Recommendation. arXiv:2307.00457 [cs.IR] https://arxiv.org/abs/2307.00457

[30] Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. arXiv:2305.06474 [cs.IR] https://arxiv.org/abs/2305.06474

[31] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907 [cs.LG] https://arxiv.org/abs/1609.02907

[32] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37. https://doi.org/10.1109/MC.2009.263

[33] John Langford and Tong Zhang. 2007. The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information. In *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis (Eds.), Vol. 20. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2007/file/4b04a686b0ad13dce35fa99fa4161c65-Paper.pdf

[34] Seth Lazar, Luke Thorburn, Tian Jin, and Luca Belli. 2024. The Moral Case for Using Language Model Agents for Recommendation. *arXiv preprint arXiv:2410.12123* (Oct. 2024). https://doi.org/10.48550/arXiv.2410.12123

[35] Daniel Lee and H. Sebastian Seung. 2000. Algorithms for Non-negative Matrix Factorization. In *Advances in Neural Information Processing Systems*, T. Leen, T. Dietterich, and V. Tresp (Eds.), Vol. 13. MIT Press. https://proceedings.neurips.cc/paper_files/paper/2000/file/f9d1152547c0bde01830b7e8bd60024c-Paper.pdf

[36] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. 2024. *Open Source Strikes Bread - New Fluffy Embeddings Model.* https://www.mixedbread.ai/blog/mxbai-embed-large-v1
[37] Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. 2023. CTRL: Connect Collaborative and Language Model for CTR Prediction. https://doi.org/10.48550/arXiv.2306.02841 arXiv:2306.02841 [cs].
[38] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. LLaRA: Large Language-Recommendation Assistant. arXiv:2312.02445 [cs.IR] https://arxiv.org/abs/2312.02445
[39] Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is ChatGPT a Good Recommender? A Preliminary Study. arXiv:2304.10149 [cs.IR] https://arxiv.org/abs/2304.10149
[40] Yichao Lu, Zhaolin Gao, Zhaoyue Cheng, Jianing Sun, Bradley Brown, Guangwei Yu, Anson Wong, Felipe Pérez, and Maksims Volkovs. 2022. Session-based Recommendation with Transformers. In *Proceedings of the Recommender Systems Challenge 2022* (Seattle, WA, USA) *(RecSysChallenge '22)*. Association for Computing Machinery, New York, NY, USA, 29–33. https://doi.org/10.1145/3556702.3556844
[41] Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Christopher Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-Rec: Personalized Recommendation via Prompting Large Language Models. arXiv:2307.15780 [cs.CL] https://arxiv.org/abs/2307.15780
[42] Meta. 2024. Introducing Meta Llama 3: The most capable openly available LLM to date. https://ai.meta.com/blog/meta-llama-3/
[43] Sheshera Mysore, Mahmood Jasim, Andrew Mccallum, and Hamed Zamani. 2023. Editable User Profiles for Controllable Text Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (Sigir '23)*. Association for Computing Machinery, New York, NY, USA, 993–1003. https://doi.org/10.1145/3539618.3591677
[44] OpenAI. 2023. Gpt-4 technical report.
[45] Emiliano Penaloza, Olivier Gouvert, Haolun Wu, and Laurent Charlin. 2024. TEARS: Textual Representations for Scrutable Recommendations. *arXiv preprint arXiv:2410.19302* (Oct. 2024).
[46] Filip Radlinski, Krisztian Balog, Fernando Diaz, Lucas Dixon, and Ben Wedin. 2022. On Natural Language User Profiles for Transparent and Scrutable Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. ACM. https://doi.org/10.1145/3477495.3531873
[47] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290 [cs.LG]
[48] Jerome Ramos, Hossen A. Rahmani, Xi Wang, Xiao Fu, and Aldo Lipani. 2024. Transparent and Scrutable Recommendations Using Natural Language User Profiles. arXiv:2402.05810 [cs.IR] https://arxiv.org/abs/2402.05810
[49] 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 *Proceedings of the ACM Web Conference 2024 (WWW '24, Vol. 33)*. ACM, 3464–3475. https://doi.org/10.1145/3589334.3645458
[50] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. arXiv:1205.2618 [cs.IR] https://arxiv.org/abs/1205.2618
[51] Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. 2023. Large Language Models are Competitive Near Cold-start Recommenders for Language- and Item-based Preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 890–896. https://doi.org/10.1145/3604915.3608845
[52] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347
[53] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. arXiv:1904.06690 [cs.IR] https://arxiv.org/abs/1904.06690
[54] Gemma Team. 2024. Gemma 2: Improving Open Language Models at a Practical Size. arXiv:2408.00118 [cs.CL] https://arxiv.org/abs/2408.00118
[55] Llama Team. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783
[56] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748 [cs.LG] https://arxiv.org/abs/1807.03748
[57] Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605. https://api.semanticscholar.org/CorpusID:5855042
[58] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the ACM SIGIR International conference on Research and development in Information Retrieval*.
[59] Yunjia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, Rui Zhang, and Yong Yu. 2023. Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models. https://doi.org/10.48550/arXiv.2306.10933 arXiv:2306.10933 [cs].
[60] Lanling Xu, Junjie Zhang, Bingqian Li, Jinpeng Wang, Mingchen Cai, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Prompting Large Language Models for Recommender Systems: A Comprehensive Framework and Empirical Analysis. https://doi.org/10.48550/arXiv.2401.04997 arXiv:2401.04997 [cs].
[61] Fan Yang, Zheng Chen, Ziyan Jiang, Eunah Cho, Xiaojiang Huang, and Yanbin Lu. 2023. PALR: Personalization Aware LLMs for Recommendation. arXiv:2305.07622 [cs.IR] https://arxiv.org/abs/2305.07622
[62] Shenghao Yang, Weizhi Ma, Peijie Sun, Qingyao Ai, Yiqun Liu, Mingchen Cai, and Min Zhang. 2024. Sequential Recommendation with Latent Relations based on Large Language Model. arXiv:2403.18348 [cs.IR] https://arxiv.org/abs/2403.18348
[63] Wenhao Yang, Yingchun Jian, Yibo Wang, Shiyin Lu, Lei Shen, Bing Wang, Haihong Tang, and Lijun Zhang. 2024. Not All Embeddings are Created Equal: Towards Robust Cross-domain Recommendation via Contrastive Learning. In *Proceedings of the ACM Web Conference 2024* (Singapore, Singapore) *(WWW '24)*. Association for Computing Machinery, New York, NY, USA, 3195–3206. https://doi.org/10.1145/3589334.3645357
[64] 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:2310.19488 [cs.IR] https://arxiv.org/abs/2310.19488
[65] Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2024. Language-Based User Profiles for Recommendation. arXiv:2402.15623 [cs.CL] https://arxiv.org/abs/2402.15623
[66] Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7B: Improving LLM Helpfulness & Harmlessness with RLAIF.
[67] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. 2008. Maximum entropy inverse reinforcement learning.. In *Aaai*, Vol. 8. Chicago, IL, USA, 1433–1438.
[68] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-Tuning Language Models from Human Preferences. arXiv:1909.08593 [cs.CL] https://arxiv.org/abs/1909.08593