UP5: Unbiased Foundation Model for Fairness-aware Recommendation

Wenyue Hua
Rutgers University
wenyue.hua@rutgers.edu

Yingqiang Ge
Rutgers University
yingqiang.ge@rutgers.edu

Shuyuan Xu
Rutgers University
shuyuan.xu@rutgers.edu

Jianchao Ji
Rutgers University
jianchao.ji@rutgers.edu

Yongfeng Zhang
Rutgers University
yongfeng.zhang@rutgers.edu

ABSTRACT

Recent advancements in foundation models such as large language models (LLM) have propelled them to the forefront of recommender systems (RS). Moreover, fairness in RS is critical since many users apply it for decision-making and demand fulfillment. However, at present, there is a lack of understanding regarding the level of fairness exhibited by recommendation foundation models and the appropriate methods for equitably treating different groups of users in foundation models. In this paper, we focus on user-side unfairness problem and show through a thorough examination that there is unfairness involved in LLMs that lead to unfair recommendation results. To eliminate bias from LLM for fairness-aware recommendation, we introduce a novel Unbiased P5 (UP5) foundation model based on Counterfactually-Fair-Prompting (CFP) techniques. CFP includes two sub-modules: a personalized prefix prompt that enhances fairness with respect to individual sensitive attributes, and a Prompt Mixture that integrates multiple counterfactually-fair prompts for a set of sensitive attributes. Experiments are conducted on two real-world datasets, MovieLens-1M and Insurance, and results are compared with both matching-based and sequential-based fairness-aware recommendation models. The results show that UP5 achieves better recommendation performance and meanwhile exhibits a high level of fairness.

KEYWORDS

Counterfactual Fairness; Individual Fairness; Recommender System; Large Language Model; Prompting

1 INTRODUCTION

Recommender Systems (RS) are algorithms designed to personalize contents or items for individual users based on their preferences. Fairness[1, 2, 13, 14, 36] in RS has drawn growing attention since it is a critical concern because these systems can significantly impact people's lives. This paper focuses on user-side[10, 22, 34, 37, 47] counterfactual fairness in large language models for RS (LLM for RS), a new type of RS adopting LLM as the backbone. User-side counterfactual fairness requires recommendations to be made independently of sensitive attributes that the user is unwilling to be discriminated against. For example, users may not want to be discriminated on their race or gender in an insurance product recommender. Ensuring counterfactual fairness is crucial in the development of recommender algorithms since neglecting to do so may result in utilizing sensitive user attributes against the user's will through the identification of behavioral similarities to make recommendations, ultimately leading to the amplification of existing unfairness and discrimination within society [21, 27]. While research on matching-based or sequential-based models has explored the issue of counterfactual fairness and the removal of sensitive attributes [27, 48], it remains an open question as to how to evaluate and mitigate these issues in the context of LLM for RS [8, 16, 51].

In most RS, each user is modeled either as a singular embedding [7, 20, 28, 32, 49] or a sequence of embeddings corresponding to the interacted items based on the user's interaction history [17–19, 39, 45, 50]. Fairness frameworks have been developed based on removing sensitive attributes from these embeddings via Pareto optimization [29], adversarial training [27, 44, 48], and graph-based models [46]. However, in the context of LLM for RS, the user's information is not consolidated into a singular user embedding, thus rendering traditional methods inapplicable. More specifically, there are three main challenges[27, 48] in addressing user-side fairness of LLM for recommendation: (1) minimizing the storage of multiple attribute-specific fairness-aware models for each attribute and their various combinations, (2) avoiding the training of separate models for each combination of attributes due to exponential growth in attribute combinations, and (3) minimizing performance decrease while producing fair recommendations, as user attributes could be important for recommendation performance.

In this work, we first develop three methods to probe the fairness of LLM for RS and detect unfairness issues in the models (more details in Section 4). We then present a novel approach to user-side fairness and propose a fairness-aware foundation model, wherein sensitive user attributes, such as gender, age, occupation, etc., can be removed or preserved based on each user's preference. Technically, our work proposes a counterfactually-fair-prompting (CFP) method for LLMs that addresses the three challenges above: (1) One prefix prompt is trained for each sensitive attribute to remove the sensitive information encoded in the model while preserving the original model parameters. Each attribute requires only the storage of a prefix prompt, significantly reducing the required storage space. (2) A Prompt Mixture module is developed to mix multiple prompts for any attribute sets specified by the user. (3) A Prompt Token Reweighter is proposed to operate on a trained prefix prompt to balance high recommendation performance and fair results. We experiment on two datasets, MovieLens-1M and Insurance for fairness research, showing the effectiveness of our model in eliminating unfairness while maintaining a high level of recommendation performance. The key contributions of the paper are as follows:

- We propose multiple probing methods to identify unfairness in LLM for RS and evaluate the degree of unfairness;
- We propose a CFP method to address unfairness issues in LLM for RS, which is both effective and parameter-efficient;
- Various experiments are conducted in single-attribute and multi-attribute scenarios to show the effectiveness of our proposed method: CFP has both better fairness and accuracy compared to other state-of-the-art fair RS models.

This paper proceeds as follows: Section 2 presents an overview of related literature on fairness in RS and prompt tuning for LLM. Section 3 briefly introduces recommendation foundation models as the backbone of the research. Section 4 examines the methods for detecting and measuring unfairness in LLMs for RS. Section 5 introduces the proposed CFP model, and Section 6 presents the experimental results for single-attribute fairness as well as combined-attribute fairness. Section 7 provides ablation studies and hyperparameter sensitivity analysis. Finally, section 8 concludes the paper.

2 RELATED WORK

2.1 Fairness in Recommender System

RS encompasses two algorithmic fairness frameworks[4, 41], namely group fairness and individual fairness [26, 38, 43]. Group fairness pertains to the equitable treatment of different groups, which is evaluated by assessing the disparities in recommendation performance, as quantified by metrics such as gaps in hits@k and NDCG across different groups [38, 40, 43]; individual fairness is concerned with whether recommendations for a user are made independently of the user's sensitive attributes, which is measured by determining whether the recommendation outcomes for a given user are equivalent in both the factual and counterfactual scenarios with respect to a specific attribute [12, 15, 27]. In the context of RS, a counterfactual world is an alternate scenario in which the user's sensitive attributes are manipulated while all other attributes independent of the sensitive attributes are held constant, defined as below [27]:

Definition 2.1 (Counterfactually fair recommendation)**.** RS is counterfactually fair if for any possible user u with features $X = x$ and $K = k$, K are the user's sensitive attributes and X are the attributes that are independent of K:

$$P(L_k|X = x, K = k) = P(L_{k'}|X = x, K = k) \tag{1}$$

for all L and for any value k attainable by K, where L denotes the Top-N recommendation list for user u.

A sufficient condition for an RS to be individually/counterfactually fair is to remove the user's sensitive information in generating recommendations so that the recommendation outcome remains unchanged across various counterfactual scenarios [27, 48]. Thus to measure individual fairness, AUC and F1 on attribute classification are commonly employed metrics [27, 48]. Li et al. [27] and Wu et al. [48] proposed two main models/frameworks for personalized individual fairness of RS. Li et al. [27] proposed a framework for matching-based models using filters to remove attribute-specific information implicitly encoded in user embeddings. To remove a set of user-sensitive attributes, each attribute filter can be averaged to produce a filter for all of them. However, this method requires updates on all model parameters. It needs to train one model for each feature, which is not parameter-efficient and thus unsuitable for large language models. Wu et al. [48] proposed to append a prefix prompt to the input sequence of items and insert an adapter in the model to improve fairness on sequential-based encoder-only RS. However, for each attribute combination, a new prefix prompt and a new adapter must be trained from scratch, thus the method cannot properly handle the exponential combination of attributes. Both methods are not directly applicable to LLMs on RS since Li et al. [27] works on matching-based models where a specific embedding is generated for each user while Wu et al. [48] works on encoder-only models such as RNN and BERT which are not trained to handle natural-language-based recommendation prompts.

2.2 Prompt Tuning

Recently, prompts [5, 25, 35] have been advanced as a lightweight methodology for downstream tasks to utilize pretrained LLM. Since discrete prompts meet the difficulty of discrete optimization, Li and Liang [25] show that soft tunable prompts are more convenient to work with despite their lack of explainability. In this work, instead of prompting the language model to generate answers for downstream tasks, our objective is to use prompts to conceal sensitive attributes of users and reduce unfair treatment during the recommendation process. As a result, we develop a light-weighted and effective method CFP for LLM, and experiments show that CFP creates a better-performing fair recommendation foundation model than baselines. Furthermore, Prompt Mixture in CFP can help to combine trained attribute-specific prompts to produce a prompt for multiple attributes while leaving the parameters of all attribute-specific prompts and the original pretrained model fixed [3].

3 PRELIMINARY OF RECOMMENDATION FOUNDATION MODELS

Foundation models such as large language models (LLMs), e.g., BERT [11], BART [24], T5 [33], and GPT-3 [5], have been shown to effectively learn rich semantics from web-scale data and transfer knowledge in pretrain data to various downstream natural language processing tasks. These models are often leveraged as backbone models as they have stored a large amount of language knowledge and their ability to capture informative representations. In the recommendation domain, P5 [16] as a recommendation foundation model increased the generalization ability of existing recommendation approaches by integrating different tasks to obtain more informative user and item representations. It is trained by input–target pairs generated from a collection of prompt templates that include personalized fields for different users and items.

In this work, to explore unfairness of recommendation foundation models, we leverage P5 as the backbone which is trained on two tasks: direct recommendation and sequential recommendation. The direct recommendation task involves prompts without user-item interactions while the sequential recommendation task includes user-item interactions in prompts. An example prompt for each task is provide in the following, where user and item IDs are represented by numeral indices.

> ***Direct Recommendation***
> **Input**: Which movie user_{{user_id}} would like to watch among the following candidates? {{movie indices with 1 positive index and 100

randomly sampled negative indices}}. **Output**: {{movie_index}}
Sequential Recommendation
Input: User_{{user_id}} has watched movies {{a sequence of movie indices this user watched}}. Which movie user_{{user_id}} would like to watch next? **Output**: {{movie_index}}

In the following section, we will conduct motivating experiments to show the unfairness issue of LLMs for RS, and then we develop methods to solve the unfairness issues.

4 PROBING UNFAIRNESS IN LLM FOR RS

Probing the user attributes out of LLM is a non-trivial task in LLM for RS because each user does not have one specific user embedding. In this section, we illustrate three methods to detect unfairness of LLM for RS. The results show that even if the training data does not explicitly use user-sensitive attributes, LLM for RS still implicitly infers user information and possibly leaks it.

In general, there are three distinct methodologies for probing user attributes in LLM: (1) eliciting attributes through in-context learning utilizing interpretable discrete prompts that are manually designed, (2) eliciting attributes through the training of tunable prompts, and in this paper, we adopt soft prompts which are more amenable to optimization compared with discrete prompts, (3) training a classifier on embeddings generated for user tokens that appear in the input prompts. The three subsections below show how much user attribute information is encoded and how they can be probed by the three methods above. We explicate which methods are useful and can be applied to LLM for RS. We also compare the results with other RS models: PMF, SimpleX, SASRec and BERT4Rec.

4.1 Manually-Designed Prompt

In the first method, we directly adopt manually-designed discrete prompts using in-context learning to probe user sensitive attributes out of the LLM. We use questions about users with (or without) their item interaction history and expect reasonable answers when multiple examples are appended in the input. More specifically, we test two types of manual prompts: direct prompt and in-context learning prompt. The direct prompt directly asks the LLM about a user's sensitive attribute, as shown by the following example, one without user-item interaction and one with user-item interaction:

Discrete Prompt without User-Item Interaction
Input: What is the {{attribute}} for user_{{user_id}}? **Output**: {{user attribute value}}
Discrete Prompt with User-Item Interaction
Input: User_{{user_id}} has watched movies (or bought insurance) {{sequence of movie (or insurance) IDs}}. What is the {{attribute}} of user_{{user_id}}? **Output**: {{user attribute value}}

The attribute can be gender, age, occupation or marital status provided by MovieLens and Insurance datasets. The answer template is simply the value of the questioned attribute, such as female/male, above/below 55 years old, or single/married. We constrain the output generated from the decoder based on constrained token generation over all possible values of the questioned attribute [9].

Table 1: Manually-Designed Prompt AUC (%)

MovieLens	gender	age	occupation	–
w/ interaction	50.33	50.09	50.00	–
w/o interaction	50.26	50.00	50.00	–
Insurance	gender	age	occupation	marital
w/ interaction	50.00	50.33	50.47	50.20
w/o interaction	50.00	50.00	50.00	50.00

For in-context learning prompts, contextual examples that are question-answer pairs of randomly sampled known users are appended before the question. We use as many contextual examples as the maximum input length allows. The following example presents in-context learning prompts for the MovieLens dataset with and without user-item interaction information. We use gray color to differentiate the context from the question.

In-context Learning Example without User-Item Interaction
Input: What is the gender of user_1? Female. What is the gender of user_2? Male. What is the gender of user_3? Female. What is the gender of user_4? Female. What is the gender of user_5? Male. What is the gender of user_10? **Output**: Male
In-context Learning Example with User-Item Interaction
Input: User_1 has watched movies 17, 1991, 29, 3039, 890. What is the gender of user_1? Female. User_2 has watched movies 29, 1084, 27, 93, 781. What is the gender of user_2? Male. User_10 has watched movies 136, 798, 2778, 1894, 1. What is the gender of user_10? **Output**: Male.

We measure the performance of probing user sensitive attributes from LLM using AUC and results are presented in Table 1. We notice that the AUC is either 50.00 or slightly above 50.00, indicating that the prediction result is no better than random guessing. Thus even if there is user sensitive information encoded in LLM such as P5 (see the next two subsections), direct prompting cannot elicit it. The reason may be that the model is trained using numerical user and item identifiers rather than natural language labels or descriptions and does not include any additional user or item metadata. Therefore, prompts designed using natural language may not align with the numerical representations used in the model's training. Manual prompts' failure can be considered as an advantage of LLM for RS, as user attributes will not be leaked too easily.

4.2 Soft Probing Prompt Tuning

In the second method, we adopt tunable prompts proposed in Lester et al. [23] to explore soft prompt tuning with a frozen pretrained LLM for RS to elicit attributes. Each attribute has one soft probing prompt trained, which is tailored to act as a question, guiding the model to produce desired outcomes. Soft probing prompts can be optimized end-to-end over a training dataset and can condense information by learning from the training. The model structure is presented in Figure 1(a). The encoder input is a concatenation of an encoder attribute prompt and an untunable discrete prompt, where the discrete prompt part includes the target user and relevant user-item interaction history, as shown below:

User user_{{user_id}} has watched movies (or bought insurances) {{sequence of item IDs}}.

Table 2: Soft Probing Prompt Tuning AUC (%)

MovieLens	gender	age	occupation	–
	70.84	64.60	56.50	–
Insurance	gender	age	occupation	marital
	50.00	51.80	50.00	70.28

Table 3: Multi-class Classifier AUC (%)

MovieLens	gender	age	occupation	–
	74.71	67.40	53.47	–
Insurance	gender	age	occupation	marital
	50.13	56.92	57.87	76.37

The decoder attends to the decoder attribute prompt, the previously generated tokens, and the encoder hidden state to predict the probability distribution of future tokens. The encoder attribute prompt and decoder attribute prompt are generated respectively by a two-layer multi-layer perceptron (MLP) and a three-layer MLP as proposed in Li and Liang [25]. The prompts are tuned by minimizing the negative log-likelihood of the attribute value tokens y conditioned on the input text x and the soft probing prompts p in an end-to-end manner:

$$L = -\sum_{j=1}^{|y|} \log P(y_j | y_{<j}, x, p) \tag{2}$$

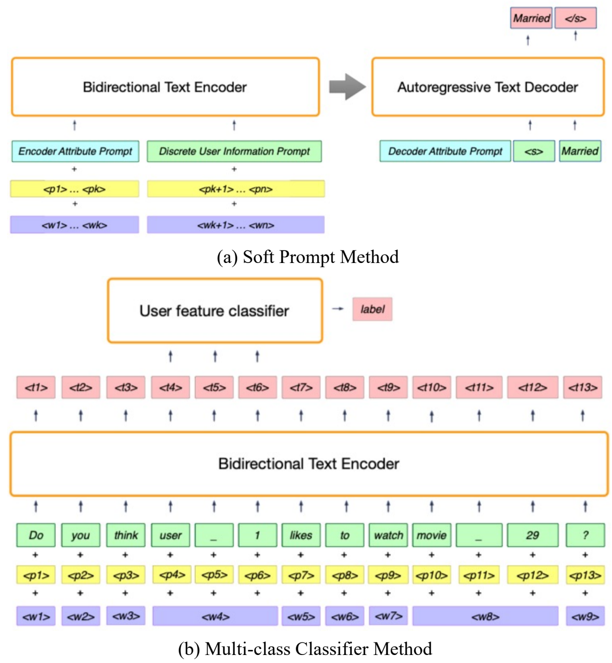
For answer generation, we also apply constraint generation as in manual prompting.

In experiments, we create separate training and test datasets by dividing all users into two groups in a 9:1 ratio, and generating a unique discrete attribute prompt for each user in the process. Experimental results on MovieLens and Insurance datasets are shown in Table 2. We notice that using soft probing prompt tuning does generate non-trivial predictions on user attributes, especially on MovieLens dataset, indicating that LLM for RS does encode user attributes and leaks personal information.

4.3 Multi-Class Classifier

The third probing method trains a multi-class classifier on the user token embeddings generated by the encoder for all input sentences in the training set. The model structure is presented in Figure 1(b), where the classifier is a seven-layer multi-layer perceptron (MLP) network trained by standard cross-entropy loss.

In experiments, the dataset utilized to train P5 is also utilized to train and test the classifiers, with a 9:1 split based on the user id included in each sentence. Tables 3 present the AUC results. The non-trivial AUC scores indicate that LLM for RS also suffers from user information leakage, similar to other RS models. We also observe that the AUC scores obtained from the trained classifier tend to be higher than those obtained through soft probing prompt tuning. This suggests that training a classifier is a more effective probing method of user attributes from LLMs than training soft probing prompts. This observation highlights that the cross-entropy loss over multiple classes is better suitable than the negative log-likelihood loss over the entire vocabulary. We will take advantage of this observation in our design of fairness-aware foundation model architecture in the following sections.



(a) Soft Prompt Method

(b) Multi-class Classifier Method

Figure 1: Model Structures of the Probing Methods

4.4 Summary of Probing the Unfairness of LLMs

This section demonstrates three possible methods to elicit user sensitive attributes from LLM for RS: manually-designed discrete prompts, soft probing prompts, and multi-class classifier. The latter two successfully generate non-trivial user attribute values among the three methods. Figure 2 illustrates the degree of unfairness on LLM models trained on MovieLens and Insurance datasets, measured by the AUC of label prediction. The model on MovieLens is unfair on gender, age, and slightly on occupation, while the model on Insurance is unfair on the marital status the most.

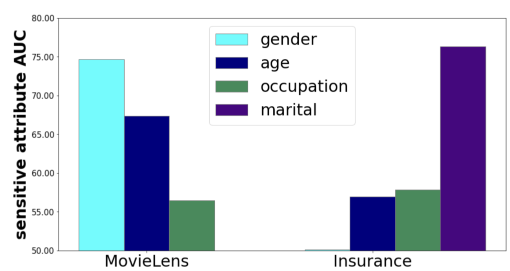


Figure 2: P5 sensitive attribute unfairness

5 COUNTERFACTUALLY-FAIR PROMPTING

We propose a counterfactually-fair prompting approach to mitigate unfairness of LLMs for RS, resulting in the development of both fair and accurate CFP model. Our approach is (1) personalized, since different users can select which attributes they wish to be treated fairly, and (2) space and time efficient, since the model does not require retraining the entire foundation model but only training the prefix prompts. The key idea of CFP is to train a counterfactually-fair encoder prompt p_{enc} for the sensitive attributes. The encoder prompt p_{enc} is concatenated to the model's plain input x to prevent

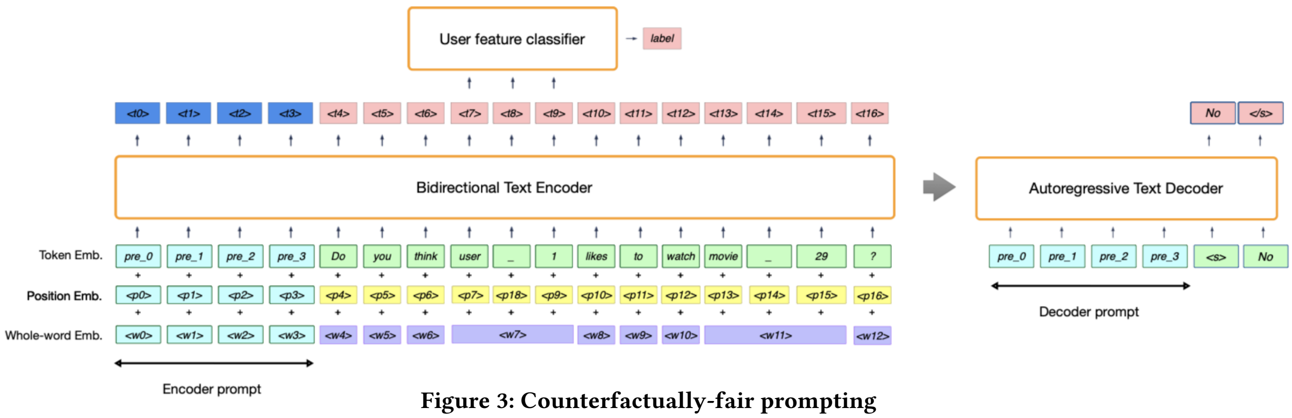


Figure 3: Counterfactually-fair prompting

the detection of sensitive attributes from the LLM. Additionally, we also train a decoder prompt p_{dec} appended to the decoder, which aids at generating the recommended item y:

$$\begin{aligned} \text{hidden\_state} &= \text{Encoder}(p_{enc} \circ x), \\ y &= \text{Decoder}(p_{dec}, \text{hidden\_state}) \end{aligned} \tag{3}$$

where $\circ$ is token concatenation. In the decoder, each token is generated based on the probability distribution:

$$P_{\theta_{dec}}(y_j | p_{dec}, y_{0:j-1}, \text{hidden\_state}) \tag{4}$$

The counterfactually-fair prompts are trained by the widely-adopted adversarial learning technique [6, 30, 52] to remove the sensitive information from the user tokens in the encoder. For parameter-efficient training, we only optimize the parameters in the prefix prompt and leave the pretrained LLM for RS untainted, making the proposed technique applicable to already pretrained LLMs. Adversarial learning requires a discriminator module [42]. According to our probing experiments in Section 4, the multi-class classifier approach demonstrates better performance in predicting user attributes than other approaches. As a result, we proceed with the classifier approach for implementing the discriminator of adversarial learning. The objective of the discriminator is to accurately predict attribute values, while the objective of the counterfactually-fair prompts is to make it difficult for the discriminator to make such accurate predictions. Figure 3 shows the model architecture.

The model training involves an iterative process in which the counterfactually-fair prompts and the classifier are optimized in succession. We denote the recommendation loss as L_{rec} and the discriminator loss as L_{dis}. L_{rec} is a negative log-likelihood loss that encourages generating the correct item index:

$$L_{rec} = -\sum_{j=1}^{|y|} \log P(y_j | p_{dec}, y_{0:j-1}, \text{hidden\_state}) \tag{5}$$

L_{dis} is a Cross-Entropy Loss (CEL) for classification. The encoder generates token embeddings E for each input token depending on p_{enc}, and L_{dis} computes the user attribute value based on the mean pooling $mean(\cdot)$ of the user relevant tokens from position i to j (e.g., the tokens "user," "_," and "1" in Figure 3). For each attribute k, let C denotes the classifier, u denotes the user, and c_u is the correct attribute value for the user, then the discriminator loss is:

$$L_{dis}^{k} = \text{CEL}(c_u | C(mean(\text{hidden\_state}[i:j]))) \tag{6}$$

The adversarial loss L_k for each attribute k is defined as below, where λ_k denotes the discriminator weight for attribute k:

$$L_k = \sum_u L_{rec} - \lambda_k \cdot L_{dis}^{k} \tag{7}$$

Algorithm 1 outlines the training process, balancing between the recommendation performance and fairness. The algorithm requires a pretrained LLM for RS $\mathcal{M}$, a randomly initialized prefix prompt $\mathcal{P}$, and a randomly initialized classifier C. The training process includes two parts: part 1 (lines 5 - 10) updates $\mathcal{P}$ based on Eq.(7) to confuse the classifier; part 2 (lines 11 - 22) updates C based on Eq.(6) to enhance the classifier's ability, and updates $\mathcal{P}$ based on Eq.(5) to maintain high recommendation performance.

Algorithm 1 Single Attribute Adversarial Training Algorithm

Require: pretrained LLM for RS $\mathcal{M}$, Randomly initialized prefix prompt $\mathcal{P}$, Randomly initialized classifier C, discriminator loss weight λ, number of epochs $Epoch\_num$, number of steps T to update C on L_{dis} or prefix prompt $\mathcal{P}$ on L_{rec}, number of batches R to update prefix prompt $\mathcal{P}$ on adversarial loss L

```
1:  for epoch ← 1 to Epoch_num do
2:      for batch_num, batch do
3:          for i ∈ [1, T] do
4:              rec_loss, u_emb ← P(M,batch)
5:              dis_loss ← C(u_emb, label_u)
6:              L ← rec_loss - λ · dis_loss
7:              Optimize P based on L with M, C fixed
8:          end for
9:          if batch_num % R == 0 then
10:             for i ∈ [1, T] do
11:                 rec_loss ← P(M,batch)
12:                 Optimize P based on rec_loss with M, C fixed
13:             end for
14:             for i ∈ [1, T] do
15:                 rec_loss, u_emb ← P(M,batch)
16:                 dis_loss ← C(u_emb, label_u)
17:                 Optimize C based on dis_loss with M, P fixed
18:             end for
19:         end if
20:     end for
21: end for
```

5.1 Prompt Token Reweighter

To generate the encoder and decoder prompts, we introduce a Prompt Token Reweighter module on top of the prompt generated by the feed-forward network (FFN) layer (Figure 4), which allows for attention among the tokens within the prompt, similar to how natural language tokens interact in a language model. This helps to improve the model expressiveness and enhance the performance, which we will show in the experiments. Techincially, as shown in Figure 4, we randomly initializes a query Q, while taking linearly projected prompt tokens as key K and value V. The query and key are used to learn a set of weights for the value, which evaluates the usefulness of each token learned by the FFN module to generate a more effective prefix prompt. The final prompt is generated by selections and linear combinations of the value tokens.

5.2 Prompt Mixture

Users may require that their recommendation not to be discriminated on multiple attributes at the same time. As a result, the encoder and decoder prompts should be able to handle the removal of multiple attributes at the same time. However, learning a prompt for each possible attribution combination can be prohibitive due to the explosive number of combinations. To solve the problem, we propose to train a Prompt Mixture layer that mixes the parameters of each trained prefix prompt by taking the concatenation of these prompts as input. The model structure is identical to that of Prompt Token Reweighter, both of which utilize an attention module which allows for flexibility of input length and thus any number of prompts can be taken as input. The computation flow is a standard attention mechanism based on the concatenation of single-attribute prefix prompts to generate mixed prompt mp:

$$\begin{aligned} k &= K(p_{k1} \circ p_{k2} \circ \cdots \circ p_{kn}) \\ v &= V(p_{k1} \circ p_{k2} \circ \cdots \circ p_{kn}) \\ \alpha_{q,k_i} &= \text{softmax}(q \cdot k_i) \\ mp &= \sum\nolimits_i \alpha_{q,k_i} \cdot v_i \end{aligned} \tag{8}$$

Same as single attribute prompt learning introduced above, the Prompt Mixture is also trained based on adversarial learning, where each step takes a random combination of sensitive attributes to be removed. The module takes a concatenation of multiple single-attribute prefix prompts as input and generates a new prompt, which is optimized to simultaneously decrease the recommendation loss and increase the sum of discriminator loss of multiple classifiers. Only the Prompt Mixture and classifiers are optimized during the training process. The loss function for one step with randomly selected set of attributes $\mathbf{K}$ in the training process is:

$$\begin{aligned} &\text{hidden\_state} = \text{Encoder}(mp_{enc}, x), \\ &L_{dis}^k = \text{CEL}(k_u | C(mean(\text{hidden\_state}[i:j]))) \\ &L_{rec} = -\sum\nolimits_{j=1}^{|y|} \log P(y_j | mp_{dec}, y_{0:j-1}, \text{hidden\_state}) \\ &L_{\mathbf{K}} = \sum\nolimits_u (L_{rec} - \sum\nolimits_{k \in \mathbf{K}} \lambda_k \cdot L_{dis}^k) \end{aligned} \tag{9}$$

The learning algorithm is similar to the single attribute adversarial training (Algorithm 1), where the only difference is to replace the adversarial loss with the multiple attribute version (Eq.(9)).

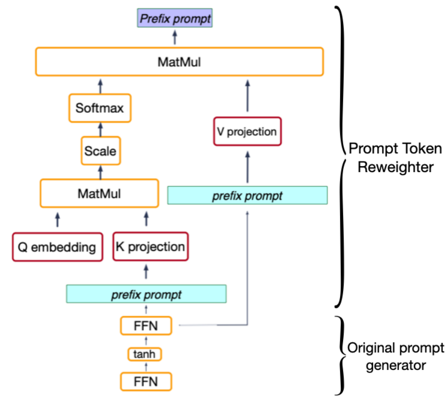


Figure 4: The Prompt Generator Model

6 EXPERIMENT

This section presents the experimental results of the CFP model on a variety of metrics, including both recommendation performance and fairness. The results show the model's ability to achieve fairness in both single-attribute and multi-attribute scenarios.

6.1 Experimental Setup

6.1.1 **Datasets.** Experiments are conducted on the MovieLens-1M dataset and Insurance dataset:

MovieLens-1M[1]: The dataset contains user-movie interactions and user profile information: gender, age, and occupation. Gender is a binary feature, occupation is a twenty-one-class feature, and age is a seven-class feature.

Insurance[2]: The dataset recommends insurance products to a target user. The user profile contains four features: gender, marital status, age, and occupation. Gender is a binary feature; marital status is a seven-class feature, and occupation is a six-class feature; for age, we group the users based on their birth year into five classes.

6.1.2 **Evaluation Metrics.** To evaluate direct and sequential recommendation tasks, one correct item is predicted among 100 randomly selected negative samples for both tasks. The metrics are hit@k for k in $\{1, 3, 5, 10\}$. We adopt the leave-one-out strategy to create the training, validation, and test datasets to train the P5 language model as the backbone. We adopt AUC for user attribute classification to evaluate whether sensitive attributes are involved in recommendation outcomes.

6.1.3 **Baselines.** We adopt four fairness-aware models as baselines: Li et al. [27]'s counterfactual-filter method applied on PMF (C-PMF) and SimpleX (C-SX) and Wu et al. [48]'s selective-prompt-adapter method on SASRec (S-SAS) and BERT4Rec (S-BERT).

PMF [32] is the Probabilistic Matrix Factorization model that adds Gaussian prior into the user and item latent factor distributions for matrix factorization. SimpleX [31] is a contrastive learning model based on cosine contrastive loss which has achieved state-of-the-art performance on recommendation performance. Li et al. [27]'s unfairness-removing filters are applied right after the user embedding computed by PMF and SimpleX, which creates C-PMF and C-SX. SASRec [19] is a sequential recommendation model

[1] https://grouplens.org/datasets/movielens/1m/
[2] https://www.kaggle.com/datasets/mrmorj/insurance-recommendation

Dataset	MovieLens									Insurance								
attribute	gender			age			occupation			age			marital			occupation		
model	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP
↑ hit@1	**16.73**	13.96	16.38	17.42	13.87	**21.22**	15.60	14.06	**21.00**	67.61	71.14	**82.53**	66.68	71.50	**81.03**	68.51	71.09	**82.53**
↑ hit@3	34.03	29.56	**35.04**	34.20	29.61	**39.22**	34.36	29.56	**38.50**	73.25	83.23	**92.68**	74.23	83.00	**90.58**	74.09	82.23	**92.68**
↑ hit@5	46.72	40.05	**47.33**	46.72	39.25	**48.85**	46.80	39.82	**49.35**	78.86	86.50	**96.44**	76.57	86.12	**94.76**	76.48	88.00	**96.44**
↑ hit@10	65.32	56.02	**65.82**	65.18	55.42	**67.30**	65.33	56.02	**69.49**	85.98	92.65	**98.89**	85.99	96.50	**97.66**	85.95	93.27	**98.89**
↓ AUC	56.62	70.80	**54.19**	62.55	79.26	**52.91**	56.01	57.02	**50.00**	50.81	51.26	**50.09**	**52.10**	56.23	52.19	54.40	**52.09**	53.28

Table 4: Results of single-attribute fairness-aware prompting on matching-based models (%)

Dataset	MovieLens									Insurance								
attribute	gender			age			occupation			age			marital			occupation		
model	S-SAS	S-BERT	CFP	S-SAS	S-BERT	CFP	S-SAS	S-BERT	CFP	S-SAS	S-BERT	CFP	S-SAS	S-BERT	CFP	S-SAS	S-BERT	CFP
↑ hit@1	20.87	23.48	**26.82**	22.95	27.98	**31.23**	18.90	24.33	**31.66**	69.40	81.20	**82.08**	70.10	75.33	**80.63**	70.09	81.20	**82.62**
↑ hit@3	41.64	42.09	**45.18**	44.10	49.32	**51.18**	20.84	43.29	**50.73**	80.05	**93.33**	92.62	80.38	84.54	**90.16**	80.38	**93.33**	92.65
↑ hit@5	49.65	**55.77**	53.46	54.99	56.56	**58.91**	29.57	51.02	**58.26**	84.48	**97.50**	96.12	85.02	90.02	**94.33**	84.39	**97.50**	95.81
↑ hit@10	60.82	62.43	**64.38**	66.00	69.38	**67.70**	43.87	59.74	**67.45**	88.34	**98.78**	98.37	88.49	94.34	**98.38**	88.91	**98.78**	98.54
↓ AUC	59.72	58.33	**54.19**	60.20	67.33	**52.91**	67.27	60.36	**50.00**	57.48	53.34	**51.23**	66.51	69.11	**50.03**	86.66	54.30	**50.82**

Table 5: Results of single-attribute fairness-aware prompting on sequential models (%)

Dataset	MovieLens			Insurance		
Model	PMF	SimpleX	P5	PMF	SimpleX	P5
↑ hit@1	19.91	17.94	**20.57**	70.20	76.50	**82.53**
↑ hit@3	38.66	**38.79**	38.38	75.23	80.12	**92.68**
↑ hit@5	**50.28**	49.84	49.60	83.12	87.34	**96.44**
↑ hit@10	65.69	65.69	**67.31**	90.04	91.41	**98.89**
↓ AUC (G)	80.22	75.52	**74.71**	52.04	53.34	**50.11**
↓ AUC (A)	82.37	79.39	**67.40**	57.94	56.87	**50.09**
↓ AUC (O)	61.32	59.40	**56.50**	58.25	57.12	**53.28**
↓ AUC (M)	–	–	–	71.30	**68.85**	69.25

Table 6: Results of matching-based recommendation, G is Gender, A is Age, O is Occupation, M is Marital Status (%).

Dataset	MovieLens			Insurance		
Model	SASRec	BERT4rec	P5	SASRec	BERT4rec	P5
↑ hit@1	28.39	29.30	**30.34**	77.26	81.20	**84.56**
↑ hit@3	**53.89**	49.06	49.26	85.15	93.33	**93.99**
↑ hit@5	**64.44**	58.90	56.47	92.30	**97.50**	97.08
↑ hit@10	**76.32**	70.06	67.40	95.76	98.78	**98.98**
↓ AUC (G)	91.90	78.52	**74.71**	73.23	61.20	**50.13**
↓ AUC (A)	92.06	73.35	**67.40**	57.93	**54.34**	56.92
↓ AUC (O)	76.57	64.79	**56.50**	88.04	**54.30**	57.87
↓ AUC (M)	–	–	–	76.61	**76.11**	76.37

Table 7: Results of sequential recommendation, G is Gender, A is Age, O is Occupation, and M is Marital Status (%).

based on left-to-right self-attention mechanism. BERT4Rec [39] is a bidirectional sequential recommendation model based on BERT. Wu et al. [48]'s prompts are appended to item sequences and adaptors are inserted into each Transformer encoder block in SASRec and BERT4Rec, which creates S-SAS and S-BERT.

Table 6 and Table 7 present the recommendation performance and unfairness of the baseline models, serving as a reference for the results of non-fairness-aware models.

6.2 Main Results of the CFP Model

This subsection presents the main experimental results. The model hyper-parameters are selected within the following range: discriminator weight $\lambda \in \{1, 5, 10, 100\}$, prefix length $\in \{5, 15, 30\}$, batch size = 16, number of steps $T \in \{10, 20\}$ to update C on L_{dis} or prefix prompt $\mathcal{P}$ on L_{rec}, number of batches $R \in \{20\}$ to update prefix prompt $\mathcal{P}$ on adversarial loss L, We train all models up to 10k steps.

6.2.1 **Sinlge-Attribute Scenario.** This subsection compares the CFP model with fair matching-based models C-PMF and C-SX in Table 4 and fair sequential-based models S-SASRec and S-BERT4Rec in Table 5, since both frameworks provide solutions in single-attribute scenarios. CFP outperforms both fair matching-based and sequential-based models in terms of both AUC and recommendation accuracy. The AUC of CFP is close to 50.00, indicating a high level of fairness, and the negative impact on recommendation performance is minimal compared with other models.

6.2.2 **Multi-Attribute Scenario.** This subsection provides experiment results on multi-attribute fairness treatment, as shown in Table 8 and Table 9. The attribute row denotes the set of attributes to be removed, where "G" represents "gender," "A" represents "age," "O" represent "occupation," and "M" represents "marital status". Two or more attributes together such as "GA" means that the sensitive attributes need to be removed at the same time. We compare our CFP model with the two matching-based fairness baselines C-PMF and C-SX from Li et al. [27], since the sequential fairness baselines from Wu et al. [48] are unable to handle mutiple attributes. We report the recommendation performance and the average AUC for the targeted user attributes in Table 8 (MovieLens) and Table 9 (Insurance). We can see that the Prompt Mixture is an effective method to combine the trained single-attribute prefix prompts, achieving fairness in models while at the same time maintaining high recommendation performance.

6.2.3 **Counterfactually-fair prompts for soft probing prompts.** Though the counterfactually-fair prompts are trained using the

model	**GA**			**GO**			**AO**			**GAO**		
attribute	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP
↑ hit@1	14.93	15.61	**16.33**	15.25	15.53	**18.67**	14.84	15.43	**21.37**	15.09	15.67	**20.18**
↑ hit@3	32.11	31.79	**37.48**	32.70	31.84	**39.02**	31.83	31.87	**39.83**	32.58	31.85	**38.79**
↑ hit@5	43.28	42.33	**47.86**	43.39	42.41	**48.94**	42.36	42.47	**49.53**	43.58	42.54	**48.50**
↑ hit@10	60.51	58.82	**66.89**	60.58	58.78	**66.39**	59.51	58.71	**68.40**	60.75	58.87	**66.78**
↓ avg. AUC	58.03	70.25	**54.22**	56.57	60.90	**52.10**	56.57	64.41	**50.00**	56.54	65.19	**53.21**

Table 8: Results of multi-attribute fairness-aware prompting on MovieLens dataset (%)

model	**AO**			**AM**			**MO**			**AMO**		
attribute	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP	C-PMF	C-SX	CFP
↑ hit@1	63.68	71.58	**79.00**	62.27	71.23	**80.91**	62.44	71.11	**78.30**	64.38	72.30	**81.63**
↑ hit@3	70.55	80.50	**89.22**	69.78	79.18	**90.97**	69.39	81.22	**88.45**	70.11	81.78	**91.52**
↑ hit@5	75.00	85.14	**93.65**	74.33	84.50	**95.23**	74.58	85.43	**93.44**	74.84	84.58	**95.37**
↑ hit@10	84.88	93.61	**97.66**	83.85	93.22	**98.73**	84.88	93.52	**97.33**	85.90	93.35	**97.37**
↓ avg. AUC	58.38	55.98	**50.80**	55.60	59.97	**50.79**	57.86	59.79	**50.64**	57.44	58.43	**50.74**

Table 9: Results of multi-attribute fairness-aware prompting on Insurance dataset (%)

MovieLens	gender	age	occupation
	51.78	50.00	50.00
Insurance	age	occupation	marital
sequential	50.00	50.00	50.00
direct	50.00	50.00	50.00

Table 10: AUC of the soft probing prompt method (%)

	Prompt (5)	Prompt (15)	Prompt (30)	PM
Parameters	0.08%	0.2%	0.5%	3.3%

Table 11: Relative number of parameters of counterfactually-fair prompting compared to the backbone foundation model

multi-class classifiers as the discriminator module in adversarial training, they can also prevent the soft probing prompt method from inferring the user-sensitive attributes. Table 10 presents the AUC results on the soft probing prompt method when appending the trained single-attribute counterfactually-fair prompts before the inputs. The results show that soft probing prompts cannot extract any user attribute information from the input when the counterfactually-fair prompts are added to the input.

6.2.4 **Number of parameters.** This section provides information on the number of parameters needed for the counterfactually-fair prompts and the Prompt Mixture module. A single-attribute or multi-attribute prefix prompt of length 5 contains approximately 92k parameters, which is roughly 0.08% of the parameters in the P5 backbone foundation model (about 110 million parameters). The Prompt Mixture has about 3 million parameters, accounting for 3.3% of the backbone parameters. Table 11 presents the number of parameters in prefix prompts of lengths 5, 15, and 30, and the Prompt Mixture (PM), as compared to the backbone. We can see that the parameters for the fairness-aware modules are minimal, making counterfactually-fair prompting a efficient solution compared to fine-tuning the whole foundation model.

7 FURTHER ANALYSIS

This section discusses the effect of different model structure designs on CFP. We first experiment on the prefix prompt model structure: (1) whether the attentional module in Prompt Token Reweighter is useful, (2) does longer prefix length affect the model performance, and (3) how does the discriminator affect the model performance. Then, we discuss whether we can utilize soft probing prompt as the discriminator module to train counterfactually-fair prompts.

7.1 Model Structure of Prefix Prompt

7.1.1 **Prompt Token Reweighter.** We explore the effectiveness of Prompt Token Reweighter by comparing it with (1) a prompt generator model without the reweighter (2) a prompt generator model with reweighter but replacing the attentional layer of the Prompt Token Reweighter with a plain feedfoward layer to make sure that it is not the extra parameters that boost the model performance. We make the feed-forward layer 1.4x larger than the original attentional layer so that the total number of parameters is comparable to the original attentional layer. We conduct experiments on the Insurance dataset since mitigating sensitive attributes in the Insurance dataset leads to a more significant decline in recommendation performance when using simple FFN structure introduced in Li and Liang [25]. We test the sequential recommendation model on age (S-age) and marital (S-marital) attributes as well as the direct recommendation model on the marital (D-marital) attribute. Results are shown in Table 12. In this table, rows names with "-" after the attribute indicates no reweighter module is used (i.e., neither attentional layer nor feed-forward layer is used), "+A" indicates using the attentional-layer version of the reweighter, and "+F" indicates using the feed-forward layer version of the reweighter. We can see that the attentional Prompt Token Reweighter improves the recommendation performance without making the model more unfair; it does not further drive the AUC lower since models without it already obtain very low AUC scores. In addition, the results also shows that the feed-forward Prompt Token Reweighter does not help the model performance, thus it is not simply the extra parameters that improve the performance.

Model	hit@1	hit@3	hit@5	hit@10	AUC
S-age -	77.64	90.73	95.77	97.78	51.26
S-age +A	**82.08**	**92.62**	**96.12**	**98.37**	**51.23**
S-age +F	76.87	89.39	94.38	96.19	51.44
S-marital -	78.93	87.71	92.65	95.23	51.76
S-marital +A	**80.63**	**90.16**	**94.33**	**98.39**	**50.03**
S-marital +F	77.42	86.33	90.08	95.12	52.32
D-marital -	76.48	84.22	87.27	94.30	51.89
D-marital +A	**81.03**	**90.58**	**94.76**	**97.66**	52.19
D-marital +F	77.40	87.54	91.32	95.67	**51.65**

Table 12: Ablation study results for prompt token reweighter

7.1.2 **Hyperparameter Sensitivity.** In this section, we study the effect of prompt length (5, 10, 15, 30) and discriminator weight (0.1, 1, 10, and 100) on both recommendation performance (measured by hit@1 on sequential recommendation) and attribute inference performance. Figure 5 and 6 present the effects of prefix prompt length on MovieLens and Insurance, respectively. In general, longer prefix length hurts fairness but improves the recommendation performance. Figure 7 and 8 present the results, from which we can see that larger weights bring better fairness but hurts the recommendation performance. Results indicate that we need to choose the prompt length and discriminator weight carefully to balance the fairness-recommendation trade-off.

7.2 Soft Probing Prompt as Discriminator

This section discusses whether we can use soft probing prompt as the discriminator in adversarial training to improve fairness. According to the motivating experiments on probing fairness of LLMs (Section 4), soft probing prompt is a weaker tool to extract user attribute information compared with multi-class classier. To further validate this, We train the counterfactually-fair prompts using soft probing prompt as the discriminator. To test the effectiveness of the trained prompts, we append the trained counterfactually-fair prompts in front of the model inputs and then use 1) soft probing prompt and 2) multi-class classifier to extract user attribute information. We conduct experiments on the Insurance dataset targeting the marital status attribute trying different lengths of the counterfactually-fair prompt, and the results are shown in Figure 9. We see that after training a counterfactually-fair prompt using soft probing prompt as the discriminator, the probing prompts cannot extract any user attribute since its AUC is close to 50%, but the classifier can still extract non-trivial sensitive attribute information from the LLM.

We also investigate the length of counterfactually-fair prompts. According to Figure 9, we notice that longer counterfactually fair prompts are more effective in removing sensitive attributes, since the classifier can extract less information, while AUCs for probing prompts are always around 50.00. This result shows that to train counterfactually-fair prompts, it is better to use the classifier instead of the probing prompt as the discriminator module, since the classifier is a stronger indicator of the degree of unfairness.

8 CONCLUSION AND FUTURE WORK

In this paper, we explore the fairness of LLMs for RS. We first probe the unfairness problems of LLMs for RS based on three approaches to show that unfairness in indeed a valid concern for LLM-based recommendation models. To solve the problem, we further propose a novel counterfactually-fair prompting (CFP) method to mitigate the unfairness of LLMs for recommendation, enabling an unbiased P5 framework (UP5). Through experiments, the proposed CFP method shows its effectiveness in (1) learning counterfactually-fair prompts for each sensitive attribute while keeping the pretrained foundation model fixed, which reduces the number of trainable parameters compared with fine-tuning the whole model, (2) utilizing a Prompt Mixture module to effectively mix multiple single-attribute prompts to generate a prompt that addresses unfairness across multiple attributes, and (3) effectively reducing unfairness in recommendations while maintaining high recommendation performance. In the future, we will explore unfairness problems in other LLM for RS tasks such as explanation generation and conversational recommendation, since our proposed counterfactually fair prompting method is a very general framework that can be applied to various tasks.

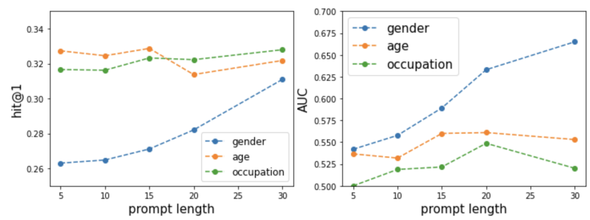


Figure 5: Different prompt length on MovieLens

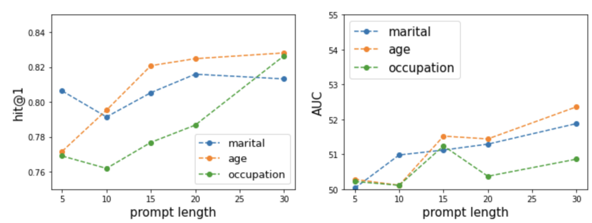


Figure 6: Different prompt length on Insurance

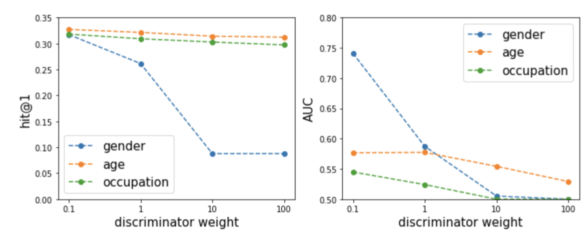


Figure 7: Different discriminator weight on MovieLens

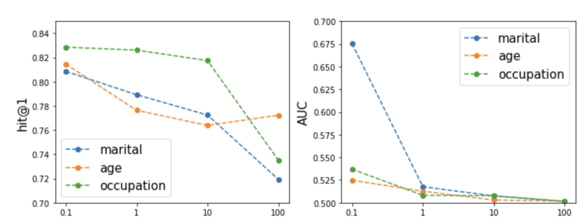


Figure 8: Different discriminator weight on Insurance

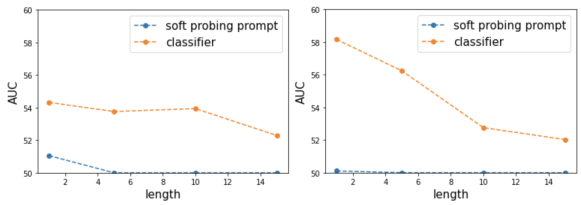


Figure 9: Effect of different CFP lengths on AUC using soft probing prompt method and classifier method for probing

REFERENCES

[1] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2020. The connection between popularity bias, calibration, and fairness in recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems.* 726–731.

[2] Enrique Amigó, Yashar Deldjoo, Stefano Mizzaro, and Alejandro Bellogín. 2023. A unifying and general account of fairness measurement in recommender systems. *Information Processing & Management* 60, 1 (2023), 103115.

[3] Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. Attentional Mixtures of Soft Prompt Tuning for Parameter-efficient Multi-task Knowledge Sharing. *EMNLP* (2022).

[4] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. 2019. Fairness in recommendation ranking through pairwise comparisons. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining.* 2212–2220.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[6] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. 2018. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069* (2018).

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems.* 7–10.

[8] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. *arXiv preprint arXiv:2205.08084* (2022).

[9] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. *International Conference on Learning Representations (ICLR) 2021* (2021).

[10] Yashar Deldjoo, Vito Walter Anelli, Hamed Zamani, Alejandro Bellogin, and Tommaso Di Noia. 2021. A flexible framework for evaluating user and item fairness in recommender systems. *User Modeling and User-Adapted Interaction* (2021), 1–55.

[11] 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* (2018).

[12] Zhenhua Dong, Hong Zhu, Pengxiang Cheng, Xinhua Feng, Guohao Cai, Xiuqiang He, Jun Xu, and Jirong Wen. 2020. Counterfactual learning for recommender system. In *Fourteenth ACM Conference on Recommender Systems.* 568–569.

[13] Michael D Ekstrand, Robin Burke, and Fernando Diaz. 2019. Fairness and discrimination in recommendation and retrieval. In *Proceedings of the 13th ACM Conference on Recommender Systems.* 576–577.

[14] Yingqiang Ge, Shuchang Liu, Ruoyuan Gao, Yikun Xian, Yunqi Li, Xiangyu Zhao, Changhua Pei, Fei Sun, Junfeng Ge, Wenwu Ou, et al. 2021. Towards long-term fairness in recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining.* 445–453.

[15] Yingqiang Ge, Juntao Tan, Yan Zhu, Yinglong Xia, Jiebo Luo, Shuchang Liu, Zuohui Fu, Shijie Geng, Zelong Li, and Yongfeng Zhang. 2022. Explainable Fairness in Recommendation. *arXiv preprint arXiv:2204.11159* (2022).

[16] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *Proceedings of the Sixteenth ACM Conference on Recommender Systems.*

[17] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management.* 843–852.

[18] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[19] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM).* IEEE, 197–206.

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

[21] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. *Advances in neural information processing systems* 30 (2017).

[22] Jurek Leonhardt, Avishek Anand, and Megha Khosla. 2018. User fairness in recommender systems. In *Companion Proceedings of the The Web Conference 2018.* 101–102.

[23] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *EMNLP* (2021).

[24] 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* (2019).

[25] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* (2021).

[26] Yunqi Li, Hanxiong Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2021. User-oriented fairness in recommendation. In *Proceedings of the Web Conference 2021.* 624–632.

[27] Yunqi Li, Hanxiong Chen, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2021. Towards personalized fairness based on causal notion. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 1054–1063.

[28] Nan Liang, Hai-Tao Zheng, Jin-Yuan Chen, Arun Kumar Sangaiah, and Cong-Zhi Zhao. 2018. Trsdl: Tag-aware recommender system based on deep learning–intelligent computing systems. *Applied Sciences* 8, 5 (2018), 799.

[29] Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on recommender systems.* 20–28.

[30] Daniel Lowd and Christopher Meek. 2005. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining.* 641–647.

[31] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A Simple and Strong Baseline for Collaborative Filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management.* 1243–1252.

[32] Aditya Krishna Menon and Robert C Williamson. 2018. The cost of fairness in binary classification. In *Conference on Fairness, Accountability and Transparency.* PMLR, 107–118.

[33] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* 21, 140 (2020), 1–67.

[34] Hossein A Rahmani, Mohammadmehdi Naghiaei, Mahdi Dehghan, and Mohammad Aliannejadi. 2022. Experiments on generalizability of user-oriented fairness in recommender systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 2755–2764.

[35] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Empirical Methods in Natural Language Processing (EMNLP).*

[36] Yash Raj Shrestha and Yongjie Yang. 2019. Fairness in algorithmic decision-making: Applications in multi-winner voting, machine learning, and recommender systems. *Algorithms* 12, 9 (2019), 199.

[37] Nasim Sonboli, Jessie J Smith, Florencia Cabral Berenfus, Robin Burke, and Casey Fiesler. 2021. Fairness and transparency in recommendation: The users' perspective. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization.* 274–279.

[38] Nasim Sonboli, Jessie J Smith, Florencia Cabral Berenfus, Robin Burke, and Casey Fiesler. 2021. Fairness and transparency in recommendation: The users' perspective. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization.* 274–279.

[39] 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. In *Proceedings of the 28th ACM international conference on information and knowledge management.* 1441–1450.

[40] Kush R Varshney. 2019. Trustworthy machine learning and artificial intelligence. *XRDS: Crossroads, The ACM Magazine for Students* 25, 3 (2019), 26–29.

[41] Guang Wang, Yongfeng Zhang, Zhihan Fang, Shuai Wang, Fan Zhang, and Desheng Zhang. 2020. FairCharge: A data-driven fairness-aware charging recommendation system for large-scale electric taxi fleets. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–25.

[42] Huaxia Wang and Chun-Nam Yu. 2019. A direct approach to robust deep learning using adversarial networks. *arXiv preprint arXiv:1905.09591* (2019).

[43] Yifan Wang, Weizhi Ma, Min Zhang, Yiqun Liu, and Shaoping Ma. 2022. A survey on the fairness of recommender systems. *ACM Transactions on Information Systems* (2022).
[44] Chuhan Wu, Fangzhao Wu, Xiting Wang, Yongfeng Huang, and Xing Xie. 2021. Fairness-aware news recommendation with decomposed adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4462–4469.
[45] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*. 495–503.
[46] Le Wu, Lei Chen, Pengyang Shao, Richang Hong, Xiting Wang, and Meng Wang. 2021. Learning fair representations for recommendation: A graph-based perspective. In *Proceedings of the Web Conference 2021*. 2198–2208.
[47] Yao Wu, Jian Cao, Guandong Xu, and Yudong Tan. 2021. TFROM: A two-sided fairness-aware recommendation model for both customers and providers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1013–1022.
[48] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Fuzhen Zhuang, Ao Xiang, Xu Zhang, Leyu Lin, and Qing He. 2022. Selective fairness in recommendation via prompts. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2657–2662.
[49] Baolin Yi, Xiaoxuan Shen, Hai Liu, Zhaoli Zhang, Wei Zhang, Sannyuya Liu, and Naixue Xiong. 2019. Deep matrix factorization with implicit feedback embedding for recommendation system. *IEEE Transactions on Industrial Informatics* 15, 8 (2019), 4591–4601.
[50] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 729–732.
[51] Yuhui Zhang, Hao Ding, Zeren Shui, Yifei Ma, James Zou, Anoop Deoras, and Hao Wang. 2021. Language Models as Recommender Systems: Evaluations and Limitations. In *I (Still) Can't Believe It's Not Better! NeurIPS 2021 Workshop*.
[52] Weimin Zhao, Sanaa Alwidian, and Qusay H Mahmoud. 2022. Adversarial Training Methods for Deep Learning: A Systematic Review. *Algorithms* 15, 8 (2022), 283.