User Profiling for Specification-Sensitive Recommendations with Large Language Model Prompting

Chih-Yu Chien,[1] **An-Zi Yen,**[2] **Hen-Hsen Huang,**[3] **Hsin-Hsi Chen**[1,4]
[1] Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
[2] Department of Computer Science, National Yang Ming Chiao Tung University, Taiwan
[3] Institute of Information Science, Academia Sinica, Taiwan
[4] AI Research Center (AINTU), National Taiwan University, Taiwan
jyjian@nlg.csie.ntu.edu.tw, azyen@nycu.edu.tw,
hhhuang@iis.sinica.edu.tw, hhchen@ntu.edu.tw

Abstract

Recently, there has been an increasing focus in research on the potential applications of large language models (LLMs) for personalized recommendations. Previous studies utilize LLMs to analyze the interaction between users and products to establish various personalized recommendation systems. However, recommendation becomes particularly challenging when items are associated with varied attributes, influenced by personal preferences, and described primarily through unstructured data. Moreover, analyzing implicit user preferences with product specifications for specification-sensitive recommendations remains largely unexplored. In this paper, we propose a framework that fully leverages prompting-based strategies to analyze user reviews and item attributes for the generation of user and product profiles, respectively. These profiles capture users' implicit preferences and enable rating prediction or product recommendation, which are crucial for personalized recommendations. Experimental results show that our proposed framework effectively handles complex item attributes and user preferences to achieve promising performances in rating prediction.

Keywords: Personalized Implicit Preference, Personalized Recommendation, Profile Generation

1. Introduction

With the Internet's growth, online services have deeply integrated into our daily lives. This highlights the importance of targeted advertising. However, accurately predicting user preferences remains a challenge. Several studies in recommendation systems have made significant progress and can be categorized into two main approaches: (1) Collaborative Filtering: recommending similar products to groups of users who hold similar preferences (Sarwar et al., 2001a; Su and Khoshgoftaar, 2009; Koren et al., 2009; He et al., 2017a), and (2) Content-Based Filtering: analyzing individual preferences and item features to make personalized recommendations (Qiu et al., 2021; Hou et al., 2022).

Recent advancements in large language models (LLMs) have demonstrated exceptional performance across various tasks. In recommendation systems, some works focus on leveraging LLMs to enhance the accuracy and relevance of recommendations. Yang et al. (2023) propose a framework that fine-tunes LLM to generate and rank recommendations based on user history. RecMind, an autonomous LLM-powered recommendation system proposed by Wang et al. (2024a), consists of three components: planning, memory, and external tools. It improves the recommendation through a "self-inspiring" technique, which explores multiple reasoning paths in the planning stage. Multi-agent is utilized in Wang et al. (2024b) to divides the recommendation process into several parts and distributes them to different LLMs.

However, existing methods are costly: traditional approaches (e.g., collaborative and content-based filtering) demand extensive feature engineering and model training, while LLM-based methods struggle to provide fast and accurate real-time predictions. Moreover, most prior studies focus on domains with relatively simple item attributes, such as movies. Products with complex specifications pose unique challenges, as subtle attribute differences can greatly influence user choices. We focus on such specification-rich scenarios where capturing implicit user preferences is crucial for accurate recommendations. To address this, we propose Verbalizing Implicit Preferences (VIP), a framework that leverages LLMs to infer implicit preferences, analyze user reviews, and generate personalized recommendations. Using Chain-of-Thought prompting (Wei et al., 2023), the LLM extracts implicit preferences and evaluates causal relations between user interests and product features to predict ratings or recommend items. By flexibly employing models of different sizes, VIP achieves efficient analysis and real-time prediction.

In this work, we focus on investigating prompting strategies for personalized recommendation under the zero-shot setting, using restaurant recommendation as our target scenario. Restaurants can be considered specification-rich items, as users' choices are often influenced by multiple factors

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 609–618
11–16 May 2026.

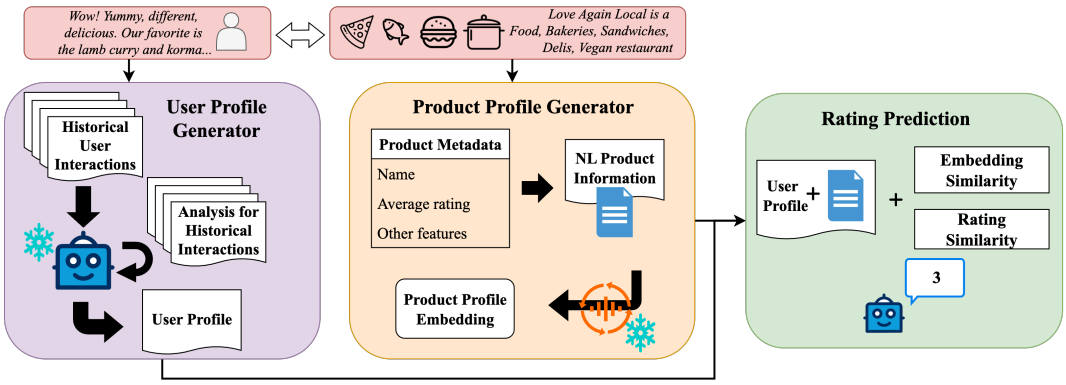

Figure 1: Overview of VIP. The snowflake icon means the model is frozen, no need for training. For embedding model, we adopt `sentence-transformers/all-MiniLM-L6-v2` for all experiments.

such as food quality, service, ambiance, and pricing. User reviews in the Yelp dataset[1] not only discuss food quality but also evaluate service, ambiance, and pricing, making them ideal for analyzing implicit user preferences. Including diverse and rich textual information, these reviews reflect user preferences, such as tastes for certain dishes or attitudes toward service, which helps in accurately building user preference models. In sum, our contributions are three-fold: (1) This work aims to explore the capability of LLMs in developing a recommendation method for specification-sensitive products by capturing implicit user preferences from user reviews and measuring the relationship between these preferences and product features. (2) We propose VIP, a pilot framework leveraging LLMs to predict user ratings and support personalized recommendations. (3) Experimental results demonstrate VIP achieves promising results without fine-tuning of LLM. Meanwhile, VIP excels in analyzing long-term preferences with an aggregated approach.

2. Methodology

Figure 1 shows an overview of our proposed VIP framework, and Algorithm 1 outlines the main stages. Concretely, VIP consists of three main components:

$$\mathcal{F} = \{\mathcal{G}_u, \mathcal{G}_p, \mathcal{R}\} \tag{1}$$

where $\mathcal{G}_u$ denotes the User Profile Generator, $\mathcal{G}_p$ represents the Product Profile Generator, and $\mathcal{R}$ is the Recommender responsible for generating personalized suggestions.

Each module relies on prompting-based inference from a large language model $\mathcal{L}$, treated as a black-box function $\mathcal{L}(x;\pi)$, where x is the input prompt and π is the prompting strategy. All the prompts used in this work are provided in Table 1 and 2. Placeholders are for specifying the target product metadata.

[1] https://www.yelp.com/dataset

Algorithm 1: Verbalizing Implicit Preferences (VIP) Framework

Input: User review history $D_u = \{(r_i, t_i)\}_{i=1}^{n_u}$;
Item i metadata m_i;
Large Language Model $\mathcal{L}$ with prompting strategies π
Output: Predicted rating $\hat{r}_{u,i}$ or
Recommendation list $\hat{r}_{u,dial}$

1. User Profile Generation
$u \leftarrow \phi$ **foreach** $(r_i, t_i) \in D_u$ **do**
  $p_i \leftarrow \mathcal{L}(r_i; \pi_{\text{review}})$ `// review analysis`
  $u \leftarrow \mathcal{L}(F_{\text{agg}}(u \cup \{p_i\}); \pi_{\text{fusion}})$ `// review fusion`
2. Product Profile Generation
$v_i \leftarrow \mathcal{L}(m_i; \pi_{\text{prod}})$
3. Recommendation (Rating Prediction)
$\hat{r}_{u,i} \leftarrow \mathcal{L}(P(u, v_i); \pi_{\text{rate}})$;
return $\hat{r}_{u,i}$ or $\hat{r}_{u,dial}$

2.1. User Profile Generator

This module is responsible for generating the user profile based on the user's historical interactions.

2.1.1. Content of User Profile

A user's favor towards products can be demonstrated in many ways, including numerical ratings, natural language reviews, or even the number of times the user interacts with the item. Since user reviews contain opinions and suggestions about the product, these reviews encapsulate both the explicit and implicit preferences of the users. Therefore, user profiling achieved by analyzing the rich textual information in the reviews can help understand user preferences from diverse scenarios. Given the natural language historical review of a specific

Prompt π	Example prompt excerpt
System Prompt	You are a helpful assistant designed to analyze user preferences and recommend {{Product Category}}.
Review Analysis (π_{review}) - Sorted Preferences	You will be given a user profile and a user review for a restaurant. Based on the given information, please answer the following three questions: (1) Record History Ratings: - Extract and list the history ratings given by the user. Format the list as [{{Product Category}} Name - Rating, ...]. (2) Determine User Preferences: - Analyze the user review to identify what aspects of the {{Product Category}} the user likes and dislikes. Clearly separate likes from dislikes in your response. (3) Sort User Values: - Based on the user's historical ratings and the identified likes and dislikes, sort the aspects that will affect the user according to how the user values them. This can include aspects such as {{Domain-specific Attributes}}, etc. When answering, please follow the order of the questions. After you finish the above three questions, please update the history rating list, combine the analyzed points with the original user profile, and shorten the results to make a new profile.
Review Analysis (π_{review}) - Predefined Preferences	... (1) {{Predefined Preference 1}} (2) {{Predefined Preference 2}} (3) History ratings of the user (4) other preferences not included in (1) $\sim$ (3)...
Review Analysis (π_{review}) - Automatic Preference Selection	... (1) History ratings of the user (2) The dream restaurant the user might prefer...
Review Fusion (π_{fusion})	Please extract the user history ratings and the original user profile from your analysis to make a new user profile.

Table 1: Prompts for user profile generator used in VIP. For the two prompting strategies - Predefined Preferences and Automatic Preference Selection, they generally have the same prompts as Sorted Preferences, except for the part specified.

Prompt π	Example prompt excerpt
System Prompt	You are a helpful assistant designed to analyze user preferences and recommend {{Product Category}}.
Product Profile Template (π_{prod})	{{Product Name}} is a {{Product Category}} located at {{Address}}. It has a rating of {{Rating}} stars out of 5. It is open on {{Open Hour}}. The restaurant accepts {{Services}}.
Rating Prediction (π_{rate})	You will be given a {{Product Category}} profile and a user profile with the user's rating history, please analyze whether the {{Product Category}} matches the user's preferences. The user profile contains the user's rating history and preferences sorted by priority from high to low. The {{Product Category}} profile contains the {{Product Category}} features. Considering the user rating tendencies, give the {{Product Category}} a score from 1 $\sim$ 5 representing how much the user will like it. Note that you should think carefully and make short explanation about which feature(s) in restaurant profile meet or not meet user's preferences. Show the scores at the end of your answer in the following JSON format: { "score": predicted score for the {{Product Category}} }

Table 2: Prompts for product profile generator and recommender used in VIP. For the Product Profile Template, we take restaurant as the target product for demonstration.

user, the user profile generator generates a user profile that encompasses the user's preferences.

2.1.2. Aggregation-Based Profiling

Within a specific time range, the user profile is generated by aggregating user reviews from earlier to more recent ones. Let $u \in \mathcal{U}$ be a user with a chronological review history:

$$\mathcal{D}_u = \{(r_i, t_i)\}_{i=1}^{n_u} \tag{2}$$

where r_i is the i-th review and t_i its timestamp.

Step 1: Review Analysis Each review r_i is processed with the LLM utilizing a specific strategy π to extract user preferences $\mathbf{p}_i$:

$$\mathbf{p}_i = \mathcal{L}(r_i; \pi_{review})$$

Step 2: Review Fusion Profiles are recursively aggregated as:

$$\mathbf{u}_i = \mathcal{L}\big(\mathcal{F}_{\text{agg}}(\mathbf{p}_i, \mathbf{u}_{i-1}); \pi_{\text{fusion}}\big) \tag{3}$$

where $\mathbf{u}_0$ is an empty profile. The final user profile is $\mathbf{u} = \mathcal{G}_u(\mathcal{D}_u) = \mathbf{u}_n$. With the aggregation-based process, the system does not need to re-analyze all previous interactions to incorporate information from a new review into $\mathbf{u}$.

2.2. Product Profile Generator

This module generates product profiles using a predefined template and product information, considering similarities among products.

2.2.1. Content of Product Profile

Product profiles provide information about the items. Content-based filtering analyzes item information to infer a user's preference, while collaborative filtering captures common features among products and makes recommendations based on their similarities. We leverage both sources of information to improve the accuracy of our predictions.

2.2.2. Profile Construction with Template

For an item $i \in \mathcal{I}$ with metadata $\mathbf{m}_i$, we define the product profile as:

$$\mathbf{v}_i = \mathcal{L}(\mathbf{m}_i; \pi_{\text{prod}})) \tag{4}$$

Let $\phi(i) \in \mathbb{R}^k$ denote the feature vector representation of i-th item. The similarity between the i-th and j-th items is computed as:

$$\text{sim}(i,j) = \lambda\left(1 - \frac{|r_i - r_j|}{\max(r_i, r_j)}\right) + (1-\lambda)\frac{\phi(i)\cdot\phi(j)}{\|\phi(i)\|\|\phi(j)\|} \tag{5}$$

Dataset	#Users	#Items	#Interactions	Sparsity
Yelp	52,256	37,222	620,141	99.97%
Ours	1,000	5,323	11,491	99.28%

Table 3: Statistics of Yelp Restaurant

where r_i and r_j are the product average ratings and $\lambda = 0.5$, a value chosen based on the analysis presented in Section 4.3.

2.3. Recommender – Rating Prediction

This paper focuses on rating prediction, a core task that estimates a user's preference for unseen items, enabling effective ranking and personalized recommendations. Given a user profile $\mathbf{u}$ and a product profile $\mathbf{v}_i$, the LLM infers a user's rating for item i:

$$\hat{r}_{u,i} = \mathcal{L}(P(u, v_i); \pi_{\text{rate}}) \tag{6}$$

The function $\mathcal{P}$ defines the prompt construction strategy for joint user-product evaluation.

3. Experiments

3.1. Dataset and Data Preprocessing

This paper analyzes user interactions to extract fine-grained implicit preferences for recommending specification-sensitive items, with restaurants as the primary study domain.

The Yelp dataset contains a large collection of real-world user-generated reviews, ratings, and metadata for businesses such as restaurants, shops, and services, often adopted for the evaluation of tasks involving user preferences, sentiment analysis, and personalized recommendations. In this work, we filter out non-restaurant businesses in the Yelp dataset by examining whether the item possesses the "restaurant" label in the *categories* attribute. Additionally, due to the large size of the Yelp dataset, we only keep reviews that were made after January 1st, 2019. For experimental purposes, we randomly select 1,000 users with at least 5 historical reviews in the user-item interactions. The statistics of Yelp restaurants are reported in Table 3. The sparsity is computed as Sparsity $= 1 - \frac{\#\text{Interactions}}{u \times p}$, where u is the number of users and p is the number of unique items interacted with by these users. Although our dataset contains fewer users and items compared to the full Yelp dataset, the sparsity level (99.28%) is very close to that of the original dataset (99.97%), meaning it retains a similar level of challenge for recommendation.

3.2. Experimental Setup

To compare the performances of different prompting strategies and backbone LLMs in the rating prediction, we conduct two sets of experiments: (1) selecting several traditional or prompting-based LLM recommenders as baselines, and (2) deploying distinct foundation models in two key roles within our framework: the User Profile Generator and the Rating Predictor to look into the performance differences. In addition to "Sorted Preferences" prompt, two other prompt strategies are tested to perform rating prediction: "Predefined Preferences" and "Automatic Preference Selection".

- **Predefined Preferences (PP):** We instruct the LLM to extract predefined domain-specific features related to the target item for recommendation.
- **Automatic Preference Selection (APS):** Different from Predefined Preferences, Automatic Preference Selection eliminates the need for manual feature engineering by prompting the LLM to identify relevant preferences autonomously, achieved by giving examples of ideal or potential ("dream" in the prompt) items first as guidance, and then project back to the preferences from these items.
- **Sorted Preferences (SP):** Although the LLM can extract preferences with the previous strategy, it sometimes generates miscellaneous preferences that are minor to the user. Moreover, accumulating multiple user reviews will possibly lead to repeated preferences being extracted. This prompt deals with the aforementioned problems by first deriving the preferences of the user's likes and dislikes, and then sorting these aspects based on how the user values them.

As for the backbone model, Llama 3.3-70B[2] and GPT-4o (Hurst et al., 2024) are utilized in our experiments. Moreover, the following baseline methods are used for comparison:

- **Matrix Factorization (MF):** MF is a collaborative filtering method that takes interactions between users and items and represents them in a low-dimensional latent space.
- **Factorization Machine (FM):** FM combines content-based and collaborative filtering, decomposing user and item features into latent factors to capture their relations.
- **LLMRec (Liu et al., 2023):** ChatGPT is utilized as a general-purpose recommender, predicting ratings, next interacted item, or generating review summarization, explanation under zero-shot or few-shot settings.
- **RecMind (Wang et al., 2024a):** An autonomous LLM-powered recommendation system consists of three components: planning,

[2] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

Method	Strategy	Backbone Model	Search Tool	Item Metadata	RMSE	MAE
MF	–	–	✗	✗	1.4095	1.1521
FM	–	–	✗	✗	1.3286	1.0372
LLMRec	Simple Prompting	Llama3.3-70B	✗	✗	1.5275	0.9918
		GPT-4o	✗	✗	1.3914	0.9640
RecMind	Self-Inspire Planning	GPT-4o	✓	✓	<u>1.2485</u>	**0.8570**
VIP	PP	Llama3.3-70B	✗	✓	1.6078	1.2950
		Llama3.3-70B + GPT-4o	✗	✓	1.4175	1.1095
	APS	Llama3.3-70B	✗	✓	1.5214	1.2305
		Llama3.3-70B + GPT-4o	✗	✓	1.3877	1.0817
	SP	Llama3.3-70B	✗	✓	1.2865	1.0310
		GPT-4o	✗	✓	1.2673	0.9840
		Llama3.3-70B + GPT-4o	✗	✓	**1.2220**	<u>0.9265</u>

Table 4: Performance Comparison of Various Models and Prompting Strategies on Yelp Rating Prediction. Results are organized by the underlying backbone model used. Predefined Preferences, Automatic Preference Selection, and Sorted Preferences are denoted as PP, APS, and SP, respectively. Bold values highlight the best performance within each section, while underlined values indicate the second-best.

memory, and external tools. RecMind features the use of tools to reach real-time information and preprocessed memory, which can help the agent generate more tailored answers.

3.3. Experimental Results

Root mean squared error (RMSE) and mean absolute error (MAE) are adopted as the evaluation metrics. The results are presented in Table 4.[3] Traditional collaborative filtering baselines (MF and FM) are included for reference. In the final row, we introduce a hybrid VIP variant that combines Llama 3.3-70B for user profile generation with GPT-4o for rating prediction (i.e., "Llama3.3-70B + GPT-4o"), achieving the overall best performance in RMSE and second-best in MAE. The findings can be summarized as follows:

Hybrid VIP (Llama3.3-70B + GPT-4o) leads in RMSE. Using Llama3.3-70B for both components performs worse than the one with the hybrid setting. The reason lies in the division of labor, where the tasks are distributed to two different models excelling in distinct fields. Additionally, this combination outperforms RecMind in RMSE without the use of tools, indicating that high-quality user representation offsets the need for external information.

Sorted Preferences generally outperforms other VIP strategies across all models. Unlike Predefined Preferences, Automatic Preference Selection and Sorted Preferences force the user profile generator to explore and analyze possible preferences based on the historical interactions, broadening the content of the generated profile. On the other hand, it asks the recommendation agent to put high-priority attributes at the front of the preference list, reminding the downstream scorer of the importance of different preferences, and to predict ratings with a relative standard.

[3]Note that RecMind could not be evaluated on Llama 3.3-70B due to the lack of tool usage support via the together.ai API.

Method	Time (secs/user)	Success Rate (%)
LLMRec	0.52	96.1
RecMind	20.28	100
VIP (PP)	10.48	100
VIP (APS)	7.01	99.8
VIP (SP)	7.90	98.8

Table 5: Time Consumption and Success Rate for Each Method. GPT-4o is utilized as the recommender model for each method.

Traditional methods demonstrate inferior performance to newer prompting-based methods. MF and FM are trained for rating prediction, which means that they lack the capability to approach the task based on semantic information. Furthermore, these methods lack the capability to explain the rationale behind an item's rating.

While tool-enhanced RecMind with GPT-4o achieves the lowest MAE, VIP provides faster and more cost-effective real-time predictions. With the reasoning process, RecMind can spontaneously ideate the information required for prediction and collect it with the tools, which helps in the real-world scenario of recommendation. However, from the point of cost, we argue that VIP is better than RecMind in the stage of predicting ratings. RecMind requires multiple runs of reasoning and tool calling to retrieve specific information, making it more time-consuming, while VIP can generate profiles in advance and provide predictions in seconds. Table 5 shows the time consumption for different methods in this experiment.

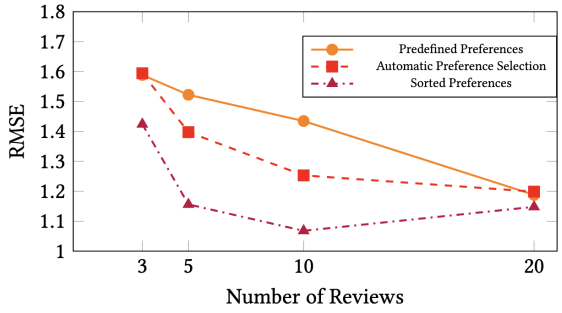

Figure 2: Comparison of RMSE for VIP with Sorted Preferences as the number of reviews increases

4. Analysis and Discussion

4.1. Impact of Review Quantity

To determine the number of user reviews necessary to effectively model implicit user preferences, we evaluate the performances of the rating prediction with user profiles generated from varying quantities of user reviews. For the users with over 20 reviews, we utilize the last c reviews to generate the user profile, where $c \in C = \{3, 5, 10, 20\}$. These generated user profiles are then utilized for rating prediction. The reason for this setup is two-fold: (1) The user preferences evolve over time, leading to more recent reviews potentially reflecting the user's current preferences more accurately. (2) There are diverse scenarios in recommendation. One is the cold-start problem, specifying where no historical interactions from the user are available. The other one is long-term preference analysis, where we extract information from numerous or even all historical interactions to model user preferences. Using different numbers of reviews to generate user profiles can simulate the two opposite conditions, enabling us to examine which setting our framework will perform better. Both the user profile generator and rating predictor are implemented using the Llama 3.3-70B model.

The results are shown in Figure 2. We find that the error decreases as the number of reviews increases, indicating that historical user-item interactions are required for modeling the users' implicit preferences. Given the specificity of reviews to individual items, accurately discerning a user's complete set of preferences is challenging. This complexity particularly exacerbates the difficulties encountered in cold-start scenarios. VIP remains comparable in performance in near cold-start scenarios, while outperforming other prompting strategies using at least 10 historical reviews. Notably, VIP with sorted preferences achieves the best performance utilizing 10 reviews for profiling, surpassing those considering 20 and interactions. This result indicates that with the VIP framework, the more user reviews it collects, the more fine-grained user preferences can be extracted. Meanwhile, this improvement will stop when reaching a specific number of historical reviews. The decline in performance may be attributed to noise generated by extended contexts but not contradictions among the users' historical interactions.

4.2. Impact of Model Capability

To examine how model capability, which is often reflected by parameter size, influences the effectiveness of each part in our VIP framework, we conduct experiments to evaluate various LLMs as backbones for the user profile generator and recommender. Specifically, we select 4 other models: GPT-4o-mini, Llama 3-8B,[4] Gemma 3-27B,[5] Mistral Medium 3,[6] as well as GPT-4o and Llama 3.3-70B for comparison. Each model is prompted without fine-tuning, relying solely on a zero-shot setting.

Comparing the first two plots in Figure 3, using Llama 3.3-70B as the user profile generator yields lower errors across rating predictors than GPT-4o in most cases. For example, when paired with GPT-4o as the rating predictor, Llama 3.3-70B achieves an RMSE of 1.2220 and an MAE of 0.9265, while GPT-4o as the generator with the same predictor records 1.2673 and 0.9840, respectively. This suggests that Llama 3.3-70B produces user representations that more effectively capture implicit preferences in this setting.

Across all rating predictors, larger models generally lead to lower errors, with GPT-4o consistently producing the best results when used as the predictor. An interesting observation is that Llama 3-8B performs competitively in all three plots, such as achieving an RMSE of 1.1439 (MAE) and 1.3629 (RMSE) when paired with Llama 3.3-70B as the generator, outperforming some larger counterparts (e.g., Genna 3-27B). In the third plot, where the same model is used for both profile generation and rating prediction, Llama 3-8B maintains strong performance (RMSE 1.4149, MAE 1.1952), indicating that it can be a practical choice in resource-constrained scenarios.

4.3. Impact of Similarity Weight

In the rating prediction task, Equation 5 in Section 2.2.2 demonstrates that two different types of similarity are combined to help make recommendations. To disclose the impact of the two similarities,

[4] https://huggingface.co/meta-llama/Meta-Llama-3-8B

[5] https://huggingface.co/google/gemma-3-27b-it

[6] https://mistral.ai/news/mistral-medium-3

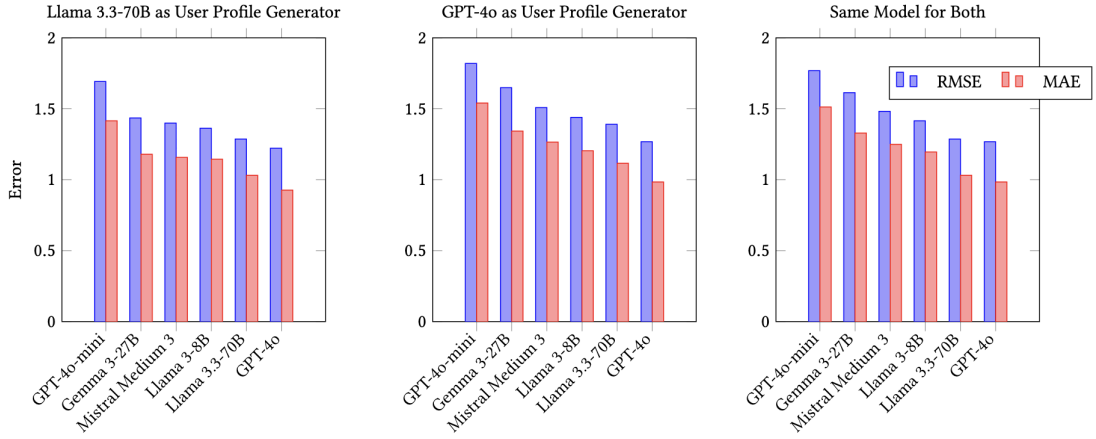

Figure 3: Performance Comparison across Combinations of User Profile Generators and Rating Predictors

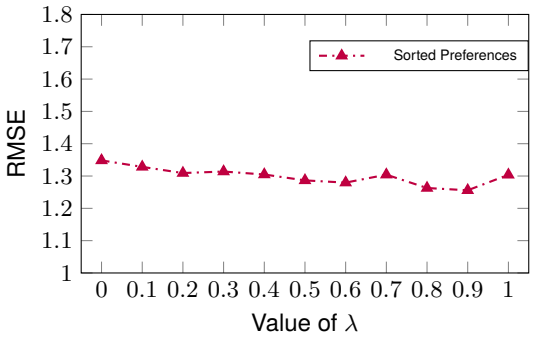

Figure 4: Comparison of RMSE for VIP with Sorted Preferences as the number of reviews increases.

Similarity Setting	RMSE	Improvement
No Similarity Information	1.3647	0%
With Rating Similarity	1.3038	4.46%
With Embedding Similarity	1.3483	1.20%
With Both Similarity ($\lambda = 0.5$)	**1.2865**	**5.73%**

Table 6: Impact of Similarity for VIP using Sorted Preferences prompting strategy.

Dataset	#Users	#Items	#Interactions	Sparsity
MobileRec	1,000	10,173	26,237	99.74%
Ours	250	7,073	4,185	99.32%

Table 7: Statistics of MobileRec

we evaluate our framework with varying λ, representing different similarity signals between target and historical items.

Figure 4 presents the results. Although the setting $\lambda = 0.9$ yields a marginally lower RMSE compared to $\lambda = 0.5$, we select $\lambda = 0.5$ as the default configuration in VIP for several important reasons. We have conducted experiments for not providing similarity information for the LLM to predict the ratings, whose performance is worse than any setting that adopts the similarity information. The improvements of performance for different settings can be found in Table 6. As shown in the figure, the RMSE decreases steadily and consistently from $\lambda = 0.1$ to $\lambda = 0.6$, indicating a stable performance trend. In contrast, the performance becomes less stable for λ values beyond 0.6, with noticeable fluctuations observed between $\lambda = 0.7$ and $\lambda = 0.9$. From a generalization perspective, $\lambda = 0.5$ provides a more robust similarity signal, which is likely to yield more consistent results across different target products or scenarios. The performance gain at $\lambda = 0.9$ is negligible and may fall within the margin of experimental variability, thus not using the extreme weighting between two types of similarities.

4.4. Framework Generalization Analysis

We introduce further experiment using an alternative dataset that possess varying characteristics. Specifically, MobileRec (Maqbool et al., 2023) is utilized to examine the generalizability of VIP. MobileRec is constructed from the data derived from Google Play. It contains a wide variety of apps with detailed specifications, pricing, descriptions, and extensive user interaction data (ratings and reviews). Its richness in both textual and numerical information makes it suitable for evaluating the effectiveness of our recommendation system. Table 7 presents the statistics of MobileRec.

We employ prompts that are almost the same as those used in the primary experiments. We choose a total of 250 users, each having at least 5 interactions, as the targets for rating prediction. Notably, since the habits of commenting on mobile apps differs from that on Yelp restaurants, making it lengthy to feed all historical reviews for user profile generation, we only take at most 50 reviews.

Table 8 presents the results. VIP with sorted preferences surpasses RecMind, indicating that our proposed method has better capability to generalize to the other domain of data. Interestingly,

Method	Backbone Model	RMSE	MAE
LLMRec	GPT-4o	1.8905	1.5823
RecMind	GPT-4o	1.7799	1.3200
VIP (PP)	Llama3.3-70B + GPT-4o	1.8232	1.5560
VIP (APS)	Llama3.3-70B + GPT-4o	1.7344	1.4372
VIP (SP)	Llama3.3-70B + GPT-4o	**1.6443**	**1.2722**

Table 8: Results of Rating Prediction on MobileRec

after studying the prediction results, we found that a specific case appears frequently: the LLM predicts 5 stars while the golden answer is only 1. We hypothesize that users' tendency to assign the lowest rating (i.e., 1) in response to minor but personally significant flaws may contribute to prediction errors. VIP with Sorted Preferences addresses this particular issue by explicitly outlining the user's prioritized aspects, distinguishing itself from the conventional self-reasoning method used by RecMind.

5. Related Work

5.1. LLMs for Recommendation

Recommendation systems primarily adopt collaborative filtering (CF) (Herlocker et al., 1999; Sarwar et al., 2001b; Koren et al., 2009) and content-based filtering (CBF) (Mooney and Roy, 2000; Lops et al., 2011), where CF leverages user or item similarities and CBF relies on item features to build user profiles, though both suffer from sparsity and cold-start issues. Since the advent of LLMs, growing research has explored leveraging their language understanding and generation abilities for recommendation tasks, including retrieval, user interaction, and personalization.

The use of LLMs in recommendation systems generally falls into two functions: retrieving relevant items based on user preferences or queries, and generating natural language outputs such as explanations, summaries, or predicted ratings. Recent work has begun integrating these two traditionally separate tasks into unified frameworks. For example, ChatRec (Gao et al., 2023) and ChatCRS (Li et al., 2025) propose interactive, explainable recommendation systems that leverage LLMs' ability to reason over both structured and unstructured data, providing a more seamless and human-like recommendation experience.

5.2. Natural Language User Profile

User profiles serve as a basis for making accurate recommendations. Mobasher et al. (2000) explored web usage mining for building dynamic user profiles from click-stream data, introducing adaptive and session-based modeling. The method of using latent representations to models user profiles have also explored. Neural collaborative filtering (He et al., 2017b) and DeepFM (Guo et al., 2017) use neural networks to learn non-linear user-item interactions from implicit feedback. Sequence-based models like SASRec (Kang and McAuley, 2018) learn dynamic user preferences by modeling sequential behavior with self-attention. Transformer-based architectures (Sun et al., 2019) have enabled richer user profiles by capturing long-range dependencies and semantic information from user interaction sequences or textual content.

Textual reviews provide abundant and informative signals about user preferences but often include irrelevant context that complicates knowledge extraction. To address this, Zheng et al. (2017) proposed DeepCoNN, a dual-network model that jointly learns user and item representations from reviews and fuses them through an interaction layer. However, its user representations are domain-specific and lack interpretability. Hence, explainable recommendation has gained attention. Gao et al. (2025) proposed LangPTune, an end-to-end framework that fine-tunes language-based user profiles via reinforcement learning from ranking metrics, jointly optimizing profile generation and recommendation for both accuracy and interpretability. Complementarily, Radlinski et al. (2022) advocate representing users through concise natural language summaries that enhance transparency, allow preference editing, and improve knowledge transfer across domains.

6. Conclusion

This paper addresses the challenge of recommending specification-sensitive products, where even minor variations in product features can significantly shape user preferences and purchasing decisions. To tackle this, we developed the Verbalizing Implicit Preferences (VIP) framework, which leverages LLMs to extract and model implicit user preferences from unstructured textual data such as reviews, and integrates these representations into the recommendation process to improve rating prediction accuracy. We evaluate VIP under diverse experimental settings. For prompting strategies, we propose Predefined Preferences (PP), Automatic Preference Selection (APS), and Sorted Preferences (SP). PP uses fixed, manually designed attributes; APS dynamically selects preference attributes based on user reviews; and SP further organizes these attributes in a ranked order to emphasize the most influential factors. We conduct experiments on Yelp restaurant reviews and the MobileRec dataset, both of which contain rich textual feedback and diverse product specifications that align with the complexity of our target recommendation scenario. To assess the effect of model capacity, we test multiple LLMs ranging from lightweight

models such as Llama 3-8B to larger models like Llama 3.3-70B and GPT-4o as both the user profile generator and rating predictor. SP achieved the best rating prediction performance, with the combination of Llama 3.3-70B as the user profile generator and GPT-4o as the rating predictor. Its effectiveness stems from structured and ranked preference attributes that reduce noise and emphasize key decision factors. For specification-sensitive products, this structure helps the model identify core attributes and guide the predictor's attention to the most influential features.

7. Ethics Statement and Limitations

While VIP demonstrates strong potential, it still faces notable limitations. Even with the sorted preferences strategy, generating high-quality user profiles requires a large model, leading to substantial computational and inference costs, especially on large-scale datasets. Developing more efficient approaches to reduce these costs is essential for real-world deployment. For future work, extending VIP beyond rating prediction to other recommendation tasks, such as sequential recommendation, could further validate its applicability.

Acknowledgement

This research was partially supported by Google gift toward the work on Multi-Perspective Sentiment Analysis on User Reviews for Knowledge-based Recommendation System (Reference Id 00007739), National Science and Technology Council, Taiwan, under grants NSTC STC 114-2221-E-002-070-MY3 and NSTC 113-2634-F-002-003-, and Ministry of Education (MOE) in Taiwan, under grants NTU-114L900901.

8. Bibliographical References

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

Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2025. End-to-end training for recommendation with language-based user profiles.

Huifeng Guo, Ruiming TANG, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: A factorization-machine based neural network for ctr prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1725–1731.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017a. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 173–182. International World Wide Web Conferences Steering Committee.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017b. Neural collaborative filtering.

Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 230–237, New York, NY, USA. Association for Computing Machinery.

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

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

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

Chuang Li, Yang Deng, Hengchang Hu, Min-Yen Kan, and Haizhou Li. 2025. ChatCRS: Incorporating external knowledge and goal guidance for LLM-based conversational recommender systems. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 295–312, Albuquerque, New Mexico. Association for Computational Linguistics.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study.

Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. 2011. *Content-based Recommender Systems: State of the Art and Trends*, pages 73–105. Springer US, Boston, MA.

M. H. Maqbool, Umar Farooq, Adib Mosharrof, A. B. Siddique, and Hassan Foroosh. 2023. Mobilerec: A large scale dataset for mobile apps recommendation. In *Proceedings of the 46th International*

ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, page 3007–3016, New York, NY, USA. Association for Computing Machinery.

Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. 2000. Automatic personalization based on web usage mining. *Commun. ACM*, 43(8):142–151.

Raymond J. Mooney and Loriene Roy. 2000. Content-based book recommending using learning for text categorization. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, DL '00, page 195–204, New York, NY, USA. Association for Computing Machinery.

Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. 2021. U-bert: Pre-training user representations for improved recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4320–4327.

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, page 2863–2874, New York, NY, USA. Association for Computing Machinery.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001a. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001b. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 285–295, New York, NY, USA. Association for Computing Machinery.

Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009.

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*, CIKM '19, page 1441–1450, New York, NY, USA. Association for Computing Machinery.

Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2024a. Recmind: Large language model powered agent for recommendation.

Zhefan Wang, Yuanqing Yu, Wendi Zheng, Weizhi Ma, and Min Zhang. 2024b. Macrec: A multi-agent collaboration framework for recommendation. In *SIGIR*, page 2760–2764.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Fan Yang, Zheng Chen, Ziyan Jiang, Eunah Cho, Xiaojiang Huang, and Yanbin Lu. 2023. Palr: Personalization aware llms for recommendation.

Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, page 425–434, New York, NY, USA. Association for Computing Machinery.