Improving LLM-Based Recommender Systems with User-Controllable Profiles

Stanisław Woźniak, Jacek Duszenko, Jan Kocoń, Przemysław Kazienko
stanislaw.wozniak@pwr.edu.pl
Department of Artificial Intelligance, Wroclaw Tech
Wroclaw, Poland

Abstract

Large Language Models (LLMs) have demonstrated significant potential across various domains, including their application in recommendation systems (RS). In this paper, we propose a method that emphasizes user control, thereby increasing the role of the human within the system. Our research investigates the effectiveness of a variety of LLMs in capturing and using user preferences for recommendation tasks. The findings reveal that incorporating user controllability into RS can enhance performance by up to 50%. Furthermore, the results highlight that textual and user-controlled representations of preferences, called user-controllable profiles, outperform historical data to improve recommendation quality.

CCS Concepts

• **Information systems** → **Recommender systems**; **Recommender systems**; • **Human-centered computing** → **HCI theory, concepts and models**; **HCI theory, concepts and models**; • **Computing methodologies** → *Natural language generation*; *Natural language generation.*

Keywords

Recommender Systems (RS), Controllable AI, User-controllable profiles, LLM-based Recommendations

ACM Reference Format:
Stanisław Woźniak, Jacek Duszenko, Jan Kocoń, Przemysław Kazienko. 2025. Improving LLM-Based Recommender Systems with User-Controllable Profiles. In *Proceedings of The International World Wide Web Conference (WWW '25).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Human-centered recommender systems represent a pivotal shift in designing recommendation algorithms, emphasizing ethical, user-focused principles over purely algorithmic optimizations. As recommender systems increasingly influence decisions in e-commerce, healthcare, media, and more, the need to align these systems with human values such as fairness, transparency, privacy, and accountability has become a paramount concern [14, 19]. Despite the transformative potential of recommender systems, current models often fail to incorporate users' evolving goals, diverse preferences, and contextual nuances.

This work addresses these challenges by introducing a novel framework for controllable, explainable, and adaptable recommender systems powered by large language models (LLMs). Unlike conventional systems reliant on historical data alone, our methodology integrates dynamically modifiable user profiles and contextual relevance. These profiles, represented in natural language, offer unprecedented user interpretability and control [14, 28]. Users can refine their profiles to reflect shifting preferences or immediate goals, ensuring recommendations remain relevant and aligned with their values.

To demonstrate the viability of this approach, we employ a suite of state-of-the-art LLMs, including both open-source and proprietary models, to evaluate the interplay between user profile representation, controllability, and system performance. By leveraging few-shot learning techniques and incorporating real-world datasets, we explore the impact of fine-grained user inputs on the adaptability and transparency of recommender systems, Fig. 1.

This research seeks to advance the discourse on ethical and user-centered recommender system design by addressing critical dimensions such as transparency, fairness, and the development of innovative evaluation metrics. By integrating algorithmic precision with human-centric principles, our work highlights the transformative potential of large language models (LLMs) to redefine the landscape of responsible AI, offering a robust framework for recommender systems that prioritize human values and ethical considerations.

In this work, we defined the following research questions:

(1) How does in-context user representation impact RS performance?
(2) Does controllability increase RS performance?
(3) How does a complex in-context user representation perform in a controllable environment?

They guided our investigation and shaped the main contributions of this paper:

(1) A robust methodology for controllable, explainable recommender systems – introducing dynamically modifiable user profiles enriched by LLMs for enhanced interpretability and user agency.
(2) Evaluation of open-source and proprietary LLMs – evaluating models like ChatGPT, Qwen, Mistral, and LLaMA, assessing their adaptability to evolving user inputs and contextual relevance.
(3) A new dataset and experimental framework – developing a dataset tailored for evaluating the impact of user-driven changes on recommendations, complemented by extensive simulations and analysis across diverse user scenarios.
(4) Insights into complex user representations – demonstrating the efficacy of combining in-context few-shot learning with

WWW '25, April 28 – May 02, 2025, Sydney, Australia
2025. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

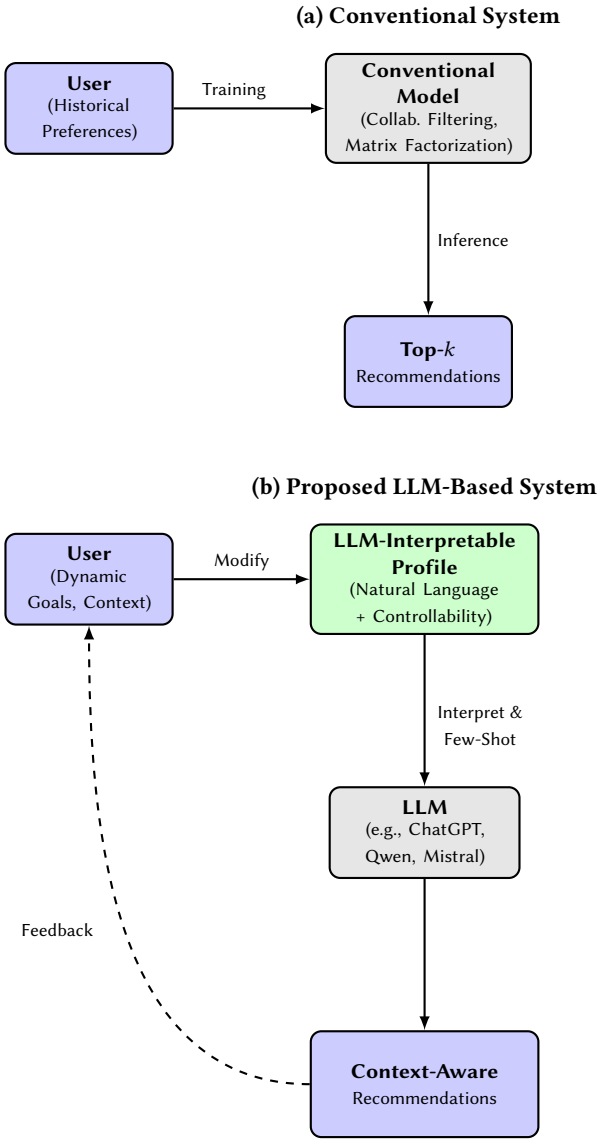

Figure 1: Comparison of a conventional recommender system (a) with the proposed LLM-based system (b). In (a), the system is trained on historical data and then infers Top-k recommendations. In (b), users can refine their natural-language profiles to incorporate shifting goals and contextual nuances, which an LLM interprets to produce context-aware recommendations.

dynamic user profiles to achieve scalable and responsive recommender systems.

2 Related Work

The intersection of recommender systems and LLMs has garnered significant attention in recent years. Zhao et al. [33] provide a comprehensive survey exploring the synergies between LLMs and recommender systems, highlighting their potential to revolutionize user-centric recommendations. For instance, in [1], the authors investigate the ability of LLMs to generate movie descriptions and compare them to existing web-scraped content, demonstrating the descriptive power of LLMs. Similarly, Shu et al. [27] propose a framework of conversational LLM agents designed to mediate between users and recommender systems, offering a more interactive and personalized recommendation experience.

Mysore et al. [24] introduced an innovative approach by defining small, interpretable concepts for user descriptions, enabling a fine-grained and editable user profile. Contrastingly, Sun et al. [28] explored the application of Llama2-7b-Chat for generating movie recommendations under specific conditions. Their findings underscore the limitations of current LLMs in adapting to unique user needs despite their capability for explainability. A more generative approach is presented by Ji et al. [11], who developed GenRec, an LLM-based system that generates target items for recommendations rather than relying on traditional ranking scores. This approach underscores the potential of leveraging inherent LLM knowledge in the absence of extensive user-specific data.

The importance of user control in recommender systems has been a recurring theme in the literature [9, 15, 26, 29, 34]. Early works focused on simplistic control mechanisms such as parameter tuning, whereas recent efforts [24, 27] emphasize open-ended conversational interfaces that empower users to guide recommendations dynamically.

Explainability is another cornerstone of trustworthy recommender systems. Ge et al. [8] provide a survey on techniques for creating explainable, fair, privacy-aware, robust, and user-controllable recommender systems, emphasizing their interdependencies. For example, Tsai et al. [29] utilized visual charts and icons to elucidate recommendation logic, while Jannach et al. [10] evaluated users' understanding of how their interactions influence recommendations on platforms like Amazon.

Personalization in AI models remains a crucial avenue for enhancing system performance across subjective tasks. Prior studies have demonstrated its impact in areas like offensive language classification [13, 17], controversy detection [12], emotion recognition [13, 22], and humor detection [13]. Various methods for integrating user-specific data into models have been proposed. Kocon et al. [18] introduced user representation techniques, while Mireshghallah et al. [23] developed UserIdentifier, a token-based representation leveraging a language model's vocabulary.

Lyu et al. [20] present LLM-Rec, a set of four prompting strategies designed to enhance personalized recommendations by incorporating both general and domain-specific knowledge. This aligns with broader efforts to fine-tune LLMs for tailored tasks [16, 25], further showcasing the transformative potential of personalization in enhancing recommendation quality and user experience.

These advancements collectively underscore the importance of explainability, personalization, and user control in the design of human-centered recommender systems, forming the foundation for our proposed methodology.

3 Method

Large Language Models (LLMs) demonstrate remarkable capabilities, and in our research, we utilize them in two distinct ways, as outlined in our previous work [32].

The first phase involves using an LLM as a profile extractor. The model receives user history as an input, such as movie ratings or reviews of previously seen films. Leveraging its internal knowledge about items-movies and its ability to generate concise summaries, the LLM produces a user profile in natural language. The aim of this profile is to capture the user's preferences effectively and in a form the user can understand.

The second phase uses an LLM as a recommender system. Importantly, the LLM used in this stage does not need to be the same as the one used for profile extraction. Here, the focus is primarily on assessing the LLM's ability to understand the domain, specifically how well it can evaluate a movie's compatibility with a given user profile. The model used for recommendations could potentially be fine-tuned for this specific task, further improving its performance. However, this fine-tuning is not required.

The user profiles created by the LLM can be easily edited before entering them into the recommender system. This functionality allows users to maintain control over the information provided to the system while enabling them to adjust the profile to their current preferences and needs.

In our previous work [32], we independently evaluated the performance of the LLM-based recommender system using user profiles or user history (ratings, reviews). In this article, our aim is to combine these two approaches to provide the system with a more comprehensive set of user information, which has the potential to improve the quality of recommendations. However, recognizing that historical preferences may not always align with current user profiles due to evolving interests and goals not reflected in past behaviour, we developed two sub-methods for this approach: (1) utilizing the entire user history and (2) filtering the history to include only data relevant to the current profile.

Finally, we have three distinct methods proposed and tested:

(1) *User Profile*. In this approach, user information for the recommender system is represented as a concise natural language description of the user's preferences, which can be manually modified and adjusted for current preferences and needs.
(2) *Few-Shot*. Here, user information is provided in the form of their interaction history, such as a list of movies along with the user's reviews or ratings.
(3) *Profiles + Few-Shot*. This method combines the two above approaches by incorporating both the user profile and the interaction history in the RS input. It considers two sub-methods based on the inclusion of either the entire history or only the relevant portion of the history, as previously described.

A comparison of these methods across selected aspects is presented in Table 1

Table 1: Characteristic of three main LLM-based recommendation methods considered

LLM-based in-context RS method	User profiles	Few-shots	Profiles + Few-shots
Size in the prompt	short	long	long
Ease of control by the user	easy	moderate	moderate
Ease of fine-tuning	moderate	easy	moderate
Increase size in time	$O(1)$	$O(n)$	$O(n)$
Inter-user aggegation	difficult	easy	moderate

4 Experiments

We performed several experiments that extend the research of [32]. In addition to new methods, we tested popular open source models from HuggingFace. For those open-source models, we used 4 NVIDIA H100 GPUs to run locally.

In each experiment, the task involves sorting a list of 20 movie titles provided in the prompt, ranking them from the most to the least preferred based on the user information. The prompt presented to the model includes the following components in order: the list of movies to be sorted, user-specific information, and a task instruction.

LLMs are characterized by variability in their output, which means that the same input can yield different results between runs. To mitigate this randomness, each experiment was repeated 10 times with the temperature parameter set to 0. Additionally, to reduce any bias introduced by the initial order of movies in the list, the order was randomized for each repetition. However, to ensure consistency between different scenarios and methods, the randomization process utilized 10 distinct seeds, keeping the randomized orders consistent within each experimental setup.

4.1 Dataset

For our study, we constructed a dataset tailored to evaluate the performance of large language models as controllable recommender systems. The dataset is structured to simulate diverse user preferences and richly captures various scenarios of user interactions, including initial profiles, profile modifications, and the impact of few-shot examples on recommendation quality. To construct our dataset, we began with movie reviews sourced from IMDb [21]. We subsequently developed initial user profiles by aggregating multiple reviews for each user and summarizing their preferences through a prompt-based interaction with a large language model. Following this, we manually created an altered user profile. With the assistance of human annotators, we established ground truth movie rankings for both the original and altered profiles. To provide relevant few-shot review examples after the user's profile was modified, we filtered the initial set of reviews using a systematic process. Reviews were ranked by their contextual relevance to the target scenarios using prompt-engineered queries (details and exact prompts are provided in the Appendix). Subsequently, reviews with the lowest contextual fit were discarded. The dataset comprises movie reviews for 50 users with columns capturing various aspects of their reviews and recommendation scenarios as well as ground truths. Below, we outline the key columns of the dataset:

- **user_id**: A unique identifier for each user.

- **reviews**: A list of review objects, each representing a review that a given user submitted for a particular movie. Each object contains:
  - **title**: The title of the movie.
  - **review_text**: The text of the user's review.
  - **rating**: The rating given by the user, represented as an integer number from 0 to 10.
- **test_movies**: A list of movies' titles to be ranked by the recommendation model in order of relevance to the user's profile.
- **changed_profile_relevant_reviews**: A list of reviews selected by another large language model (GPT-4o in our experiments) as being relevant to the modified user profile, used in scenarios evaluating whether relevant examples aid the model in achieving better ranking performance. Prompts used to filter out irrelevant samples is presented in Appendix A on Figure 5
- **z-score**: A mean z-score of user's movie ratings, used to select subsets of users best fit for particular experiment scenarios.
- **profile**: A concise representation of the user's profile, generated by the large language model based on a 10-shot summarization of their reviews.
- **profile_changed**: The modified version of the profile reflecting user-initiated changes.
- **original_labels**: The ground truth ranking for movie recommendations based on the user's original profile.
- **changed_labels**: The ground truth ranking for recommendations after the profile is altered.

4.2 Models

We evaluated the capabilities of several models following the classical transformer architecture [30], focusing on their performance as controllable recommender systems in diverse user scenarios. The selected models encompass a mix of closed-source and open-source architectures, representing cutting-edge advancements in natural language processing. These models vary significantly in size, from lightweight architectures optimized for efficiency to bigger models with tens of billion parameters. We aim to provide a comprehensive analysis of how different model classes handle tasks requiring personalization, reasoning, and dynamic adaptation. For our experiments, we have used the instruction tuned versions of the models. Below, we briefly describe the models used in our experiments:

- **Qwen 72B** [7]: an open-source, large-scale model with 72B parameters, pretrained on over 3T tokens, including Chinese, English, multilingual texts, code and mathematics datasets. Uses a vocabulary of over 150k tokens and has a context length of 32k.
- **LlaMA3.3 70B** [4]: Second to largest model from the Llama 3.3 series, comprising 70B parameters. Pretrained on a mix of publicly available online data on over 15T tokens. With 128k of context length and a vocabulary of 128k tokens.
- **LlaMA3.1 8B**: A lightweight version of the Llama 3.1 series, sharing the context length and vocabulary size with its counterpart discussed above. It balances performance and computational requirements, making it suitable for scenarios requiring fast inference and lower hardware resources by using 8B parameters.
- **Mistral 7B** [5]: A 7-billion-parameter model with context length of 8k and 32k tokens of vocabulary size, leveraging grouped-query attention [2] and sliding window attention [3] to accelerate inference speed and reduce the decoding memory requirements.
- **Mixtral 8x22B** [6]: Thanks to Sparse Mixture-of-Experts (SMoE) architecture, the model allows dynamic parameter allocation, improving computational efficiency and enabling specialized reasoning for diverse inputs. Model was trained with a 65.536k context window and a vocabulary of 32k tokens.
- **Mixtral 8x7B** : Smaller version of the above model, comprised of 8 experts totalling 56B parameters. Context length and vocabulary size are the same as those of the counterpart.
- **GPT-4o**: A large-scale proprietary model from OpenAI with estimated parameters in hundreds of billions. Features 128k context length and vocabulary size of 200k tokens. Demonstrates strong capabilities across reasoning, coding, and multilingual tasks while maintaining high factual accuracy.

These models provide a diverse set of architectures and parameter scales, enabling a robust evaluation of their capabilities as controllable recommender systems under various experimental conditions.

4.3 Scenarios

The experiments for the proposed methods were divided into three groups, each consisting of multiple scenarios. Each group is designed to address a specific research question. The primary distinction between scenarios lies in the structure of the prompt, specifically in both the type of user information provided and the manner in which it is presented.

4.3.1 ***User representation.*** The first group addresses Research Question 1 (RQ#1):*"How does in-context user representation impact RS performance?"*. The objective is to determine the most effective in-context representation across the following four scenarios:

(1) $\mathbf{FS}^{10}$: Few-shot representation with 10 samples of titles and ratings extracted from the user's interaction history.
(2) $\mathbf{P}_O$: A user profile generated by the LLM, without any modifications or input from the user for personalization.
(3) $\mathbf{P}_O$ **+** $\mathbf{FS}^{10}$: A combination of the user profile of the P_O scenario and 10 historical samples from the FS^{10} scenario. In this configuration, the prompt begins with the user profile, followed by the few shot samples.
(4) $\mathbf{FS}^{10}$ **+** $\mathbf{P}_O$: The same data as in P_O + FS^{10} but presented in reverse order, with the few shot samples preceding the user profile in the prompt.

Since this group does not explore the aspect of user control over the profile, the evaluation compares results against the unmodified ground truth without accounting for potential changes in user preferences.

4.3.2 ***Controllability.*** The second Research Question (RQ#2): *"Does controllability increase RS performance?"*, is explored in the second group of experiments. This group builds on our previous work by incorporating additional models to evaluate the effect of controllability on recommendation outcomes.

(1) $\mathbf{FS}^{10}$: This scenario is identical to the corresponding scenario in the first group, using few-shot representation with 10 samples.

(2) $\mathbf{P}_O$: Also consistent with the previous group, this scenario utilizes a user profile generated by the LLM without any user modifications.
(3) $\mathbf{P}_C$: A user profile generated by the LLM that includes adjustments made by the user, simulating controllability.

In this group, controllability is explicitly examined, so results are compared against a ground truth that simulates evolving user preferences.

4.3.3 ***Complex representation in controllable environment.*** Research Question 3 (RQ#3): *"How does a complex in-context user representation perform in a controllable environment?"* is investigated in the final group of experiments. This group focuses on creating more sophisticated user representations by combining user history with user profiles in various configurations.

(1) $\mathbf{P}_C$: Identical to the scenario in the previous group, representing a user profile generated by the LLM with user modifications to simulate controllability.
(2) $\mathbf{FS}^{10}$: The same as in previous groups, using a few-shot representation with 10 samples.
(3) $\mathbf{FS}^{all}_{relevant}$: Similar to FS^{10} but filters the samples to include only those relevant to the user's profile.
(4) $\mathbf{P}_C$ **+** $\mathbf{FS}^{10}$: Combines P_C with FS^{10}, providing the model with all user information available, including both profile and history.
(5) $\mathbf{P}_C$ **+** $\mathbf{FS}^{all}_{relevant}$: Combines P_C with $FS^{all}_{relevant}$, ensuring that only relevant samples are included to avoid misleading the model with non-relevant data.
(6) $\mathbf{P}_C$ **+** $\mathbf{FS}^{limited}$: Simulates scenarios where context length constraints prevent including all information. Instead of using all 10 samples as in P_C + FS^{10}, only 3 random samples are selected.
(7) $\mathbf{P}_C$ **+** $\mathbf{FS}^{limited}_{relevant}$: A constrained version of P_C + $FS^{all}_{relevant}$, where only 3 random samples are selected, but exclusively from the relevant ones.

In this group, the recommendation results are evaluated against a ground truth that reflects changes in user preferences, further examining how complex representations perform under dynamic conditions.

It is important to note that in scenarios involving the filtering of relevant samples - $FS^{all}_{relevant}$, P_C + $FS^{all}_{relevant}$, P_C + $FS^{limited}_{relevant}$ - the filtering process was conducted once using GPT-4o, which served as the profile extractor to generate the original user profiles.

Prompt templates are in Appendix A. For Profile scenarios (P_O, P_C) are on Figure 6, for Few shot scenarios (FS^{10}, $FS^{all}_{relevant}$) are on Figure 9. For combined scenarios (P_O + FS^{10}, P_C + FS^{10}, P_C + $FS^{all}_{relevant}$, P_C + $FS^{limited}$, P_C + $FS^{limited}_{relevant}$) prompt template is on Figure 7 and for the reversed order scenario (FS^{10} + P_O) is on Figure 8.

4.4 Metrics

We used two metrics: *NDCG* to evaluate the RS performance and *Gain* to test the improvement of one scenario over a reference one.

4.4.1 ***NDCG.*** Normalized Discounted Cumulative Gain (*NDCG*) is a standard metric for assessing the quality of recommendations. It quantifies the performance by comparing the Discounted Cumulative Gain (*DCG*) of the generated ranking to the Ideal Discounted Cumulative Gain (*IDCG*), which represents the best possible *DCG* achievable based on the ground truth ordering. The metric is calculated as follows:

$$NDCG@X = \frac{DCG@X}{IDCG@X}$$

where $NDCG@X$ means that $NDCG$ is calculated from the top X items in the ranking. Furthermore, DCG is calculated as follows:

$$DCG@X = \sum_{i=1}^{X} \frac{rel_i}{\log_2(i+1)}$$

where i is the item position, and rel_i - the relevance score of i. In this study, rel_i corresponds to the item's position in the ground truth ranking, sorted from the least to the most preferred, allowing for multiple items to share the same position. This approach ensures that the items most important to the user — those they like the most — are assigned the highest relevance scores

4.4.2 ***Gain.*** It is designed to evaluate the extent to which a given scenario improves results over the specified baseline scenario:

$$Gain = \frac{100\% \times (Mod - Base)}{100\% - Base}$$

where Mod represents the $NDCG$ value of a scenario modified compared to a baseline, reflecting its potential improvement over it. $Base$ refers to the $NDCG$ value of the baseline scenario, which, corresponds to the FS^{10} scenario within each group.

5 Results

5.0.1 ***User representation.*** In Table 2, we present a comparison of the results for scenarios within the *User Representation* group. In particular, all models demonstrate improved performance when using user profiles (P_O) compared to the few-shot representation (FS^{10}). Additionally, most models exhibit decreased performance in the combined scenarios (P_O + FS^{10} and FS^{10} + P_O) relative to the user profile scenario (P_O).

Another notable observation is that the order in which the information is provided to the recommender system affects performance, particularly in less advanced LLMs. Although LLaMA3.3 70B and GPT-4o show minimal differences between the two combined scenarios, most models perform better when few-shot samples are presented before the user profile, rather than the reverse.

Interestingly, LLaMA3.1 8B achieved its highest performance in the FS^{10} + P_O scenario, even surpassing the P_O scenario. However, the difference is less than one percentage point, indicating only a marginal improvement in the quality of the recommendations.

Figure 2 presents the *Gain* metric across scenarios in this group, using FS^{10} as the baseline. This comparison highlights which user representation has the greatest impact on RS performance. Notably, better-performing models exhibit relatively small *Gain* across scenarios. However, in less effective models, the *Gain* from the P_O scenario is consistently the highest, demonstrating the superiority of profile representation over few-shot representation.

In conclusion, user profiles consistently outperform few-shot approaches in recommendation quality, with information ordering being significant - presenting few-shot samples before profiles generally yields superior results, particularly in less sophisticated language models.

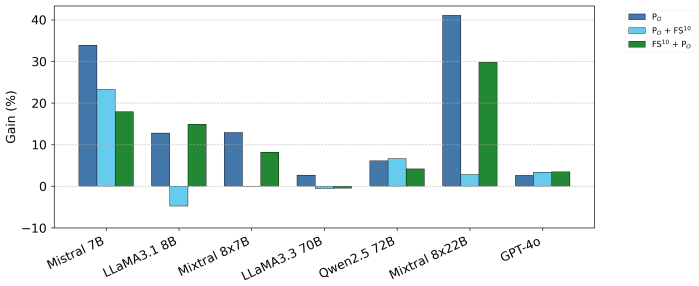

Figure 2: *Gain* across all scenarios for the *User Representation* scenarios.

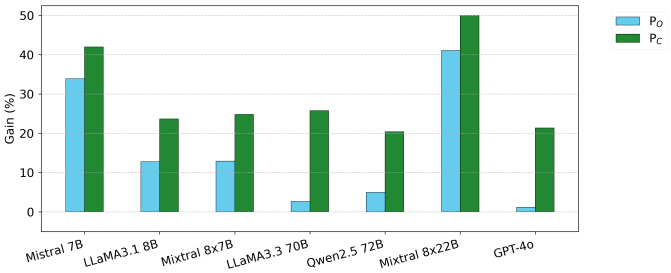

Figure 3: *Gain* across all scenarios for the *Controllability* group.

Table 2: Results for *User Representation* group. All numbers represents *NDCG@10* metric with original ground truth without taking into account evolving preferences.

Model	FS^{10}	P_O	P_O + FS^{10}	FS^{10} + P_O
Mistral 7B	0.4136	0.6123	0.5499	0.5186
LLaMA3.1 8B	0.5580	0.6145	0.5369	**0.6238**
Mixtral 8x7B	0.5664	**0.6224**	0.5662	0.6020
LLaMA3.3 70B	**0.6061**	0.6165	**0.6040**	0.6043
Qwen2.5 72B	0.5758	0.6018	0.6039	0.5936
Mixtral 8x22B	0.2924	0.5829	0.3122	0.5032
GPT-4o	**0.6322**	**0.6419**	**0.6446**	**0.6449**

Table 3: Results for *Controllability* group. All numbers represents *NDCG@10* metric with changed ground truth that takes into account evolving preferences.

Model	FS^{10}	P_O	P_C
Mistral 7B	0.4136	0.6123	0.6596
LLaMA3.1 8B	0.5580	0.6145	0.6626
Mixtral 8x7B	0.5664	**0.6224**	0.6740
LLaMA3.3 70B	**0.6061**	0.6165	**0.7076**
Qwen2.5 72B	0.5811	0.6018	0.6665
Mixtral 8x22B	0.2924	0.5829	0.6460
GPT-4o	**0.6364**	**0.6406**	**0.7141**

5.0.2 ***Controllability.*** Results of group *Controllability* are shown in Table 3. The most important observation is that the scenario with controlled profile (P_C) shows superiority over the non-controlled profiles (P_O) or the few-shot approach (FS^{10}). Each model from the smallest to the largest shows that control by the user is crucial in improving the efficacy of the recommendation.

In Figure 3, we present the gains for all profile scenarios within the *Controllability* group compared to the FS^{10} scenario used as the baseline. The results indicate that controllable representations consistently provide over a 20% improvement in performance and outperform non-controlled profile scenarios in every case.

In conclusion, controlled user profiles consistently demonstrate superior recommendation performance compared to both non-controlled profiles and few-shot approaches, highlighting the critical role of user control in enhancing recommendation efficacy across all model sizes.

5.0.3 ***Complex representation in controllable environment.*** The results of the final group, *Complex representation in controllable environment*, are presented in Table 4. It is evident from the results that LLaMA3.3 70B is the best performing open source model, showing performance comparable to GPT-4o in the recommender system.

An important observation is that filtering out non-relevant samples from the few-shot representation improves results, as the model is not misled by irrelevant data. However, the controlled profile still outperforms the few-shot scenarios.

Moreover, combining controlled profiles with few-shot samples from the user's history enhances the performance of the few-shot scenarios (FS^{10}, $FS^{all}_{relevant}$), although they still do not surpass the controlled profile scenario (P_C). Notably, the scenario P_C + FS^{10} performs consistently worse than or equal to P_C + $FS^{all}_{relevant}$, which aligns with intuitive expectations.

In the limited scenarios (P_C + $FS^{limited}$, P_C + $FS^{limited}_{relevant}$), the results are similar to each other and outperform the non-limited combined scenarios (P_C + FS^{10}, P_C + $FS^{all}_{relevant}$), but still fall short of surpassing the controlled profile scenario (P_C).

Figure 4 illustrates the *Gain* metric for this group compared to the baseline FS^{10} scenario. The results show that the combined scenarios, profiles supplemented with few shot samples, almost always outperform the baseline. However, they rarely exceed the performance of the controlled profile scenario (P_C). Furthermore, complex representations (profiles combined with few shot samples) demonstrate similar *Gain* values, with no clearly superior scenario emerging.

The results suggest that a textual profile may provide more accurate information to the model than historical samples. Consequently, combining the profile with few-shot samples can mislead the model in making recommendations. This is why the *relevant* scenarios outperform the non-relevant ones. Additionally, the *limited* scenarios show better performance, as the model is constrained by fewer samples, thus reducing the chance of being misled by irrelevant data.

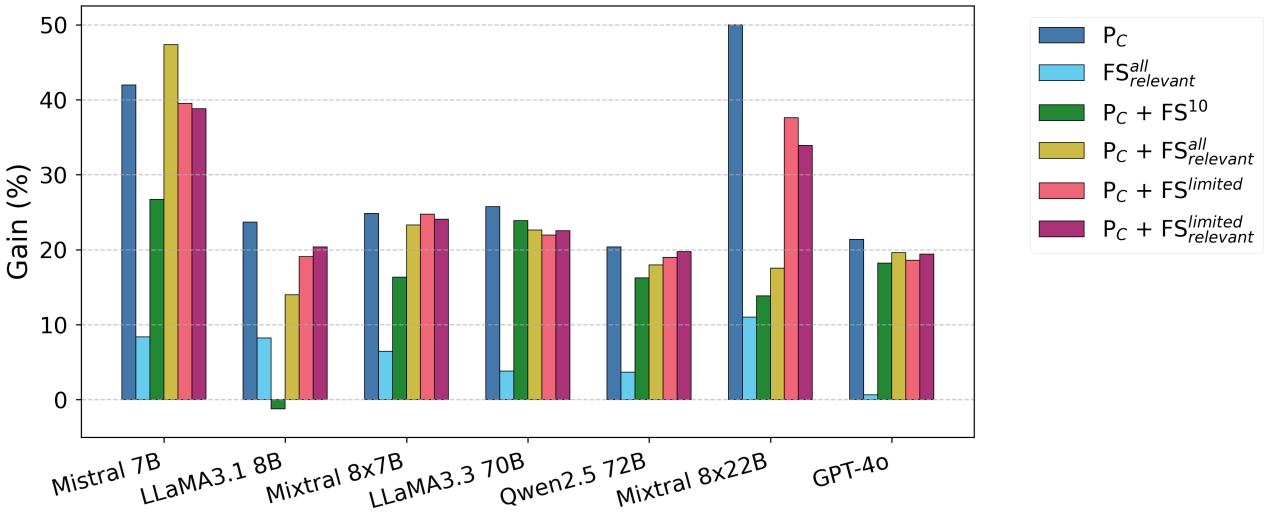

Figure 4: *Gain* across all scenarios for the *Complex Representation* group.

Table 4: Results for *Complex representation in controllable environment* group. All numbers represents *NDCG@10* metric with changed ground truth that takes into account evolving preferences.

Model	P_C	FS^{10}	$FS^{all}_{relevant}$	P_C + FS^{10}	P_C + $FS^{all}_{relevant}$	P_C + $FS^{limited}$	P_C + $FS^{limited}_{relevant}$
Mistral 7B	0.6596	0.4136	0.4628	0.5702	0.6913	0.6453	0.6412
LLaMA3.1 8B	0.6626	0.5580	0.5944	0.5525	0.6199	0.6423	0.6480
Mixtral 8x7B	0.6741	0.5664	0.5943	0.6372	0.6674	0.6737	0.6708
LLaMA3.3 70B	**0.7076**	**0.6061**	**0.6211**	**0.7002**	**0.6952**	**0.6926**	**0.6948**
Qwen2.5 72B	0.6665	0.5811	0.5964	0.6492	0.6564	0.6606	0.6638
Mixtral 8x22B	0.6460	0.2924	0.3704	0.3905	0.4165	0.5584	0.5323
GPT-4o	**0.7141**	**0.6364**	**0.6388**	**0.7027**	**0.7076**	**0.7040**	**0.7069**

6 Discussion

It is important to note that GPT-4o, a closed source LLM, consistently achieved the best results in all scenarios. None of the open-source models surpassed its performance. The closest competitor was LLaMA3.3 70B, which in some cases achieved results nearly identical to GPT-4o. However, the remaining open-source models struggled to match the performance of GPT-4o.

Interestingly, Mixtral 8x22B, despite being the largest model we tested, sometimes produced the worst results, particularly in scenarios involving few shot samples. This occurred because the model struggled to correctly sort the list of titles provided in the ground truth, occasionally including titles from the few-shot samples or even unrelated ones. As a result, the model's performance was significantly reduced due to mismatches with the ground truth titles. Although similar issues were observed in other models, they were much less frequent and had a smaller impact on the overall results.

Open-source models demonstrated significant potential as recommender systems compared to GPT-4o. However, they encountered challenges with constrained output. Although GPT-4o's output format was ideal and required no manual correction before parsing, open-source models consistently needed human intervention to fix their output. Some models, such as Mixtral 8x7B and LLaMA3.3 70B, produced only minor errors, whereas others, such as Qwen2.5 72B, generated many errors. Even when testing Qwen2.5 7B, its output format was so flawed that manually correcting each output would have been too time-consuming.

On the other hand, open-source LLMs can be fine-tuned to the particular recommendation task with the instruction data (the model better understands the task) or preferences – then, the model may be personalized, i.e. aligned to the particular user [31], [12], [13].

7 Conclusions and future work

In this work, we introduced new methods for user-controllable recommender systems based on LLMs, which directly implement our general idea of user-centred responsible recommender systems (RRSs) [14].

We demonstrated that LLMs better understand a domain through natural language descriptions, as seen in our profile method, compared to providing in-context few shot samples and expecting the model to infer user preferences from them.

The research revealed that incorporating such samples into the model's input generally leads to lower performance than using textual profiles alone.

This study opens several directions for future research, the main one being the fine-tuning of LLM-based recommendation systems. The other ones are: the use of different models to filter relevant samples, prompt extension with domain knowledge about items, and continual adaptation with reinforcement learning.

Acknowledgment

This work was financed by: (1) the National Science Centre, Poland, project no. 2021/41/B/ST6/04471; (2) the statutory funds of the Department of Artificial Intelligence, Wroclaw University of Science and Technology; (3) CLARIN ERIC – European Research Infrastructure Consortium: Common Language Resources and Technology Infrastructure (period: 2024-2026) funded by the Polish Minister of Science under the programme: "Support for the participation of Polish scientific teams in international research infrastructure projects", agreement number 2024/WK/01; (4) the European Regional Development Fund as part of the 2021–2027 European Funds for a Modern Economy (FENG) programme, project no. FENG.02.04-IP.040004/24; (5) the Polish Ministry of Education and Science within the programme "International Projects Co-Funded"; (6) the European Union under the Horizon Europe, grant no. 101086321 (OMINO). However, the views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency. Neither the European Union nor European Research Executive Agency can be held responsible for them.

References

[1] Arkadeep Acharya, Brijraj Singh, and Naoyuki Onoe. 2023. LLM Based Generation of Item-Description for Recommendation System. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 1204–1207. https://doi.org/10.1145/3604915.3610647

[2] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. arXiv:2305.13245 [cs.CL] https://arxiv.org/abs/2305.13245

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. arXiv:2004.05150 [cs.CL] https://arxiv.org/abs/2004.05150

[4] Aaron Grattafiori et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[5] Albert Q. Jiang et al. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[6] Albert Q. Jiang et al. 2024. Mixtral of Experts. arXiv:2401.04088 [cs.LG] https://arxiv.org/abs/2401.04088

[7] Jinze Bai et al. 2023. Qwen Technical Report. *ArXiv* abs/2309.16609 (2023). https://api.semanticscholar.org/CorpusID:263134555

[8] Yingqiang Ge, Shuchang Liu, Zuohui Fu, Juntao Tan, Zelong Li, Shuyuan Xu, Yunqi Li, Yikun Xian, and Yongfeng Zhang. 2022. A survey on trustworthy recommender systems. *ACM Transactions on Recommender Systems* (2022).

[9] F Maxwell Harper, Funing Xu, Harmanpreet Kaur, Kyle Condiff, Shuo Chang, and Loren Terveen. 2015. Putting users in control of their recommendations. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 3–10.

[10] Dietmar Jannach, Sidra Naveed, and Michael Jugovac. 2017. User control in recommender systems: Overview and interaction challenges. In *E-Commerce and Web Technologies: 17th International Conference, EC-Web 2016, Porto, Portugal, September 5-8, 2016, Revised Selected Papers 17*. Springer, 21–33.

[11] Jianchao Ji, Zelong Li, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Juntao Tan, and Yongfeng Zhang. 2023. GenRec: Large Language Model for Generative Recommendation. *ArXiv* abs/2307.00457 (2023). https://api.semanticscholar.org/CorpusID:259332879

[12] Kamil Kanclerz, Alicja Figas, Marcin Gruza, Tomasz Kajdanowicz, Jan Kocoń, Daria Puchalska, and Przemyslaw Kazienko. 2021. Controversy and conformity: from generalized to personalized aggressiveness detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 5915–5926.

[13] Przemysław Kazienko, Julita Bielaniewicz, Marcin Gruza, Kamil Kanclerz, Konrad Karanowski, Piotr Miłkowski, and Jan Kocoń. 2023. Human-centered neural reasoning for subjective content processing: Hate speech, emotions, and humor. *Information Fusion* 94 (2023), 43–65.

[14] Przemysław Kazienko and Erik Cambria. 2024. Toward Responsible Recommender Systems. *IEEE Intelligent Systems* 39, 3 (2024), 5–12.

[15] Oznur Kirmemis and Aysenur Birturk. 2008. MoRe: A user controlled content based movie recommender with explanation and negative feedback. In *International Conference on Web Information Systems and Technologies*, Vol. 2. SCITEPRESS, 271–274.

[16] Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, et al. 2023. ChatGPT: Jack of all trades, master of none. *Information Fusion* (2023), 101861.

[17] Jan Kocoń, Alicja Figas, Marcin Gruza, Daria Puchalska, Tomasz Kajdanowicz, and Przemysław Kazienko. 2021. Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. *Information Processing & Management* 58, 5 (2021), 102643.

[18] Jan Kocoń, Marcin Gruza, Julita Bielaniewicz, Damian Grimling, Kamil Kanclerz, Piotr Miłkowski, and Przemysław Kazienko. 2021. Learning personal human biases and representations for subjective tasks in natural language processing. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1168–1173.

[19] Yang Li, Kangbo Liu, Ranjan Satapathy, Suhang Wang, and Erik Cambria. 2024. Recent developments in recommender systems: A survey. *IEEE Computational Intelligence Magazine* 19, 2 (2024), 78–95.

[20] Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, and Jiebo Luo. 2023. Llm-rec: Personalized recommendation via prompting large language models. *arXiv preprint arXiv:2307.15780* (2023).

[21] Andrew L. et al. Maas. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015

[22] Piotr Miłkowski, Marcin Gruza, Kamil Kanclerz, Przemyslaw Kazienko, Damian Grimling, and Jan Kocoń. 2021. Personal bias in prediction of emotions elicited by textual opinions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*. 248–259.

[23] Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2022. UserIdentifier: Implicit User Representations for Simple and Effective Personalized Sentiment Analysis. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 3449–3456.

[24] Sheshera Mysore, Mahmood Jasim, Andrew McCallum, and Hamed Zamani. 2023. Editable User Profiles for Controllable Text Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 993–1003.

[25] Sebastian Porsdam Mann, Brian D Earp, Nikolaj Møller, Suren Vynn, and Julian Savulescu. 2023. AUTOGEN: A personalized large language model for academic enhancement—Ethics and proof of principle. *The American Journal of Bioethics* 23, 10 (2023), 28–41.

[26] Verus Pronk, Wim Verhaegh, Adolf Proidl, and Marco Tiemann. 2007. Incorporating user control into recommender systems based on naive bayesian classification. In *Proceedings of the 2007 ACM conference on Recommender systems*. 73–80.

[27] Yubo Shu, Haonan Zhang, Hansu Gu, Peng Zhang, Tun Lu, Dongsheng Li, and Ning Gu. 2024. RAH! RecSys–Assistant–Human: A Human-Centered Recommendation Framework With LLM Agents. *IEEE Transactions on Computational Social Systems* (2024).

[28] Ruixuan Sun, Xinyi Li, Avinash Akella, and Joseph A Konstan. 2024. Large Language Models as Conversational Movie Recommenders: A User Study. *arXiv preprint arXiv:2404.19093* (2024).

[29] Chun-Hua Tsai and Peter Brusilovsky. 2021. The effects of controllability and explainability in a social recommender system. *User Modeling and User-Adapted Interaction* 31 (2021), 591–627.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 5998–6008.

[31] Stanisław Woźniak, Bartłomiej Koptyra, Arkadiusz Janz, Przemysław Kazienko, and Jan Kocoń. 2024. Personalized large language models. In *SENTIRE 2024 - Workshop on Sentiment Elicitation from Natural Text for Information Retrieval and Extraction at ICDM'2024 - The 24thd IEEE International Conference on Data*

Mining. IEEE.
[32] Stanisław Woźniak, Sidney Suen, Bartłomiej Koptyra, Maria Kazienko-Sobczuk, Aleksander Szczęsny, Przemysław Kazienko, Jan Kocoń, Erik Cambria, and Kenneth Kwok. [n. d.]. Explainable and User-Controllable Profiles Including Chatgpt-Memory: Toward Better Llm Recommendations. *Available at SSRN 4982389* ([n. d.]).
[33] Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. 2024. Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering* (2024).
[34] Luyao Zhu, Rui Mao, Erik Cambria, and Bernard Jim Jansen. 2024. Neurosymbolic AI for Personalized Sentiment Analysis. In *Proceedings of HCII*.

A Prompts

Filtering out irrelevant samples.

Prompt

You will be given a user profile and a list of ten movie titles with their ratings. Discard the movies that are irrelevant to the user's profile.
If you don't find any irrelevant movies, do not discard anything. Present your output as a list of movies along with their titles and ratings in a Python dictionary format.
Return ONLY items from the output list.
- Do not add any explanation
- Do not include any additional text
- Respond with ONLY the items in the list exactly as they appear
- Do not add any extra formatting strings such as "'python. Just return a raw string dictionary

User profile:
user profile
List of movies:
list of movies

Figure 5: Prompt for filtering out irrelevant samples.

Profile scenarios

Prompt

Given this list of movie titles:
movies list

and users preferences:
user profile

reorder this list items from the most preferred to the least preferred. Return a stringified Python list.
Return ONLY items from the output list. ALWAYS use a double quote sign (") to wrap list items in the output list.
- Do not add any explanation
- Do not include any additional text
- Respond with ONLY the items in the list exactly as they appear
- Do not add any extra formatting such as "'python. Just return a raw string.
- Always use double quotes sign " to wrap string list items, for example : ["Ocean's Twelve", "The Fast and the Furious", "Gone in 60 Seconds"]

Figure 6: Prompt for scenarios using only Profiles

Profile + Few shot scenarios

Prompt

Given this list of movie titles:
movies list

and users preferences:
user profile

and sample movies the user has watched and reviewed along with their reviews on a scale from 1 to 10:
user reviews

reorder this list items from the most preferred to the least preferred. Return a stringified Python list.
Return ONLY items from the output list. ALWAYS use a double quote sign (") to wrap list items in the output list.
- Do not add any explanation
- Do not include any additional text
- Respond with ONLY the items in the list exactly as they appear
- Do not add any extra formatting strings such as "'python. Just return a raw string.
- Always use double quotes sign " to wrap string list items, for example : ["Ocean's Twelve", "The Fast and the Furious", "Gone in 60 Seconds"]

Figure 7: Prompt for scenario with Profile and Few shot approaches with ordered respectively

Few shot and Profile scenarios

Prompt

Given this list of movie titles to rank:
movies list

and sample movies the user has watched and reviewed along with their reviews on a scale from 1 to 10:
user reviews

and users preferences:
user profile

reorder this list items from the most preferred to the least preferred. Return a stringified Python list.
Return ONLY items from the output list. ALWAYS use a double quote sign (") to wrap list items in the output list.
- Do not add any explanation
- Do not include any additional text
- Respond with ONLY the items in the list exactly as they appear
- Do not add any extra formatting strings such as "'python. Just return a raw string.
- Always use double quotes sign " to wrap string list items, for example : ["Ocean's Twelve", "The Fast and the Furious", "Gone in 60 Seconds"]

Figure 8: Prompt for scenario with Few shot and Profile approaches with ordered respectively.

Few shot scenarios

Prompt

Given this list of movie titles to rank:
movies list

and sample movies the user has watched and reviewed along with their reviews on a scale from 1 to 10:
user reviews

reorder the list of movie titles from the most preferred to the least preferred taking user reviews into consideration. Return a stringified Python list.
Return ONLY items from the output list. ALWAYS use a double quote sign (") to wrap list items in the output list.
- Do not add any explanation
- Do not include any additional text
- Respond with ONLY the items in the list exactly as they appear
- Do not add any extra formatting such as "'python. Just return a raw string.
- Always use double quotes sign " to wrap string list items, for example : ["Ocean's Twelve", "The Fast and the Furious", "Gone in 60 Seconds"]

Figure 9: Prompt for scenarios using only few shot approach