# AI Agents-Enhanced Reinforcement Learning for Personal Recommendation Systems

**Caroline Silva** [1]

## Abstract

Achieving personalization in recommendation systems remains a significant challenge. While Large Language Models (LLMs) exhibit strong generalization capabilities, they are trained on broad internet-scale data and tend to capture expected preferences rather than truly personalized ones. This limitation makes it difficult for LLMs to model user-specific behaviors accurately. To address this, we explore an alternative approach by leveraging AI agents that simulate real user personas to enhance reward modeling. This work compares a transformer-based reward model (baseline) trained on the MovieLens dataset with an agent-driven reward model that generates synthetic user profiles using the TinyTroupe framework and GPT-4o. Experimental results indicate that while the transformer model achieves higher accuracy in explicit rating prediction, AI agents demonstrate superior ranking performance, as measured by NDCG, suggesting their potential to better capture user preferences. However, challenges remain in refining agent interactions and improving prompt engineering to enhance personalization.

## 1. Introduction

Large Language Models (LLMs) possess the remarkable ability to process and understand unstructured data, such as user reviews, social media posts, and product descriptions, extracting valuable information that is often unavailable in traditional ID-based recommendation datasets(Yu et al., 2025). Therefore, LLMs offer a promising solution to improve RL training by acting as user behavior simulators (Corecco et al., 2024), thus reducing dependence on real-world data collection.

(Cao et al., 2024) has proposed a framework for LLM-enhanced RL based on the capabilities of pretrained knowledge-inherent LLMs, such as reasoning, generative modeling, and structured information processing, to assist in RL learning. They have defined key roles that LLMs can play in RL-based: information processor, reward designer, decision-makers and generator. For the purpose of this work, we will explore the role of the reward designer.

The Reward Designer role involves using an LLM to evaluate recommendations by giving a score that is associated with the user prediction, which will be compared against a transformer-based reward model trained on real user data. The goal is to assess whether AI agents that are described to act as a persona can outperform a transformer-based model in terms of capturing the real users preferences.

The Idea of using AI Agents as reward model in order to predict users preferences is inspired on the results from Schoenegger & Caviola. where they have shown that AI models, such as GPT4o and Claude 3 Opus, can outperform the vast majority of laypeople and academic experts in predicting correlations between personality items.

However, achieving personalization in recommendation systems remains a significant challenge. While Large Language Models (LLMs) exhibit strong generalization capabilities, they are trained on broad internet-scale data and tend to capture expected preferences rather than truly personalized ones.

Inspired by these ideas, the TinyTroupe framework (Salem et al., 2024) is utilized to generate agents that reflect user preferences. TinyTroupe is an experimental Python library designed to simulate individuals with distinct personalities, interests, and goals. These artificial agents, known as TinyPersons, can engage with other agents, exchange responses, and operate within simulated environments called TinyWorlds. The framework leverages Large Language Models (LLMs), particularly GPT-4o, to generate realistic and dynamic agent behavior, enabling more lifelike user simulations.

[1]Department of Computer Science, Stanford University, Palo Alto, USA. Correspondence to: Caroline Silva <cm199204@stanford.edu, https://github.com/cmarquesdasilva/cs234>.

*Proceedings of the $42^{nd}$ International Conference on Machine Learning*, Vancouver, Canada. PMLR 267, 2025. 

This work aims to predict user movie preferences using the well-known MovieLens dataset (Harper & Konstan, 2015). The approach begins with training a transformer-based reward model on historical user data, which is then used to train a simple policy for movie recommendation. This serves as a baseline framework, providing a foundation for further improvements using AI agents or LLMs. The primary objective is to explore whether replacing the transformer-based reward model with AI agents can enhance recommendation performance and outperform the traditional approach. The idea of this comparison is because we are comparing a model that were training on this specific user preference data while the other is levering the generalization capabilities of LLMs.

## 2. Research Question

This work hypothesizes that AI agents can effectively model user preferences and serve as a reward model within the proposed movie recommendation pipeline.

The main research question is:

- Can AI agents accurately capture and simulate user preferences in personalized interactions?

## 3. Methodology

### 3.1. Reward Model

The methodology for training the reward model is described as follows:

**Dataset Preparation**

The dataset used in this work is sourced from MovieLens (Harper & Konstan, 2015). It contains user ratings, movie metadata, and timestamps, making it well-suited for training a reward model that captures user preferences.

The dataset is constructed from a combination of user-movie interactions, genre metadata, and movie plot descriptions. The `MovieRatingDataset` class is implemented to preprocess and structure the data as described below:

- **User and Movie Mapping**: Users are identified by their `userId`, while movies are mapped to a compact index for efficient embedding lookup.
- **Genre and Plot Representation**: Genres and plots are treated as textual descriptions and tokenized into embeddings.
- **Rating Label Encoding**: Ratings are mapped to discrete classes, either using a 10-class full scale covering all possible ratings from 0.5 to 5.0.

**Model Architecture**

The **Reward Model** is a transformer-based neural network designed to learn an implicit user preference function. The architecture consists of the following key components:

- **User and Movie Embeddings**: Separate embedding layers are used to represent users and movies in a shared latent space.
- **Genre and Plot Embeddings**: These are processed using a MiniLM model and passed through a simple MLP to reduce the embedding space. The outputs are then combined with the movie embedding to generate the same latent space.
- **Transformer Encoder**: The user and the resulting movie latent space are concatenated and processed through a **multi-layer transformer encoder** with self-attention mechanisms.
- **Feedforward Network (FFN)**: The output of the transformer encoder is aggregated and passed through a **fully connected network** to generate rating predictions.

In this work, this model architecture is referred to as the **"vanilla encoder-only transformer" (v-tf)**.

**Loss and Optimization**

The reward model is trained using a supervised learning approach with categorical cross-entropy loss. The training is conducted for 25 epochs using the Adam optimizer with a fixed learning rate.

**Evaluation**

- The trained model is evaluated on a validation set using **accuracy**.
- After the ratings for the validation set are predicted, the movies are ranked per user, and the **NDCG** score is computed.

This reward model serves as the foundation for reinforcement learning-based policy training, providing a learned reward function for ranking and recommendation tasks.

### 3.2. Policy Training

The movie catalog comprises nearly 9,000 movies, making it necessary to reduce the action space complexity for efficient policy training. To achieve this, the first ranking stage, RankZero, is implemented. RankZero combines heuristics and similarity search to refine the candidate set before policy learning. Instead of considering all movies, RankZero

generates a curated Top-25 candidate movie list per user. This is achieved through:

- Heuristics: Movies are initially ranked based on popularity.
- Similarity Search: Movie embeddings are used to find films similar to a user's past preferences.

By limiting the action space to a refined set of 25 movies per user, the policy model is trained more efficiently, focusing on a meaningful subset rather than the entire catalog.

The learned reward model is used as the objective function for training a policy to optimize movie recommendations using reinforcement learning. The policy is trained using Proximal Policy Optimization (PPO).

**Policy Model**

The policy model processes movie embeddings and learns optimal recommendation strategies using an actor-critic reinforcement learning approach. The baseline policy was trained with Proximal Policy Optimization (PPO), where the policy selects an action based on action probabilities derived from the model.

In the policy model, the policy network utilizes a Softmax activation in its final layer, while the value network consists of a single linear layer without activation. This design allows the policy network to produce a probability distribution over possible actions, while the value network estimates the expected reward for a given state without requiring a probability distribution.

The input consists of a batch of top 25 movie embeddings, aggregated to form a latent representation of the user's context.

**Training Set-up**

- The RankZero framework retrieves the top 25 movie candidates for each user by leveraging movie embeddings and FAISS-based similarity search. The policy model then interacts with the recommendation environment as follows:
- The policy selects an action (movie recommendation) from the distribution learned by the model.
- The selected movie is evaluated using the reward model, which provides feedback in the form of a scalar reward.
- PPO updates the policy parameters by optimizing the clipped surrogate objective using the combined loss (policy + value).
- The training is conducted for a fixed number of episodes, where each of each is comprised by a single step because of computational resources constraints.

**Evaluation**

To ensure a fair and meaningful evaluation of the trained policy model, we compare its performance against a baseline random policy.

Each policy evaluation follows a single-step decision-making process per episode. The policy model operates by selecting a movie recommendation from a pre-filtered Top-25 candidate list generated by the RankZero framework. The reward function then evaluates the selected movie, producing a scalar reward that reflects the effectiveness of the recommendation.

To establish a baseline, the policy-driven recommendation is compared to a random policy. The random policy selects a movie uniformly from the Top-25 list, with its expected reward assumed to be the average of all possible rewards.

### 3.3. Agent-Driven Reward Modeling

A simple prompt on GPT-4o model is utilized to generate user profiles based on historical movie rating scores. The ratings are provided as input in a prompt, allowing GPT-4o to summarize each user's preferences. This process is conducted using the same training data employed for the reward model.

The prompt provided to GPT-4o instructs it to analyze a user's movie rating history and generate a summary of their preferences. The analysis is based on psychological traits, cognitive styles, and emotional tendencies inferred from the user's rated movies.

The model extracts insights related to the Big Five Personality Traits (Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism), sensation-seeking behavior, empathy levels, cognitive style (preference for structured vs. non-linear narratives), and need for closure vs. exploration (preference for clear resolutions vs. open-ended stories).

The output containing personality traits (rated as low/medium/high), preferred genres, favorite and disliked movies, and additional observations. This profile is then used to create synthetic users using the Tiny Troupe framework, which simulates real user preferences in a recommendation system. Due to budget constraints, only a subset of users is included in this process.

After generating the agents, they are prompted to assign rating scores to movies from the validation set. These synthetic ratings are then used to evaluate and compare the agent-based reward model against the transformer-based

reward model.

The agents were generated and utilized through interactions with GPT-4o.

## 4. Results and Discussion

### 4.1. Transformer-Based Reward Modeling

The results indicate overall low accuracy across most users in both the training and validation sets. As expected, the median accuracy in the training set is slightly higher due to exposure to the learning process. To further analyze the model's performance, Normalized Discounted Cumulative Gain ($NDCG@k$) was computed, where $k$ was set to $[3, 5, 10, 25, 50, 100]$. The average NDCG was calculated for each user, and the distribution is visualized through boxplots

The accuracy distribution, shown in the boxplot below, reveals that few users achieve reasonable accuracy, a significant portion exhibits substantially lower performance. The median accuracy in the validation set remains below 0.4, indicating room for improvement, but it overcome a random selection strategy.

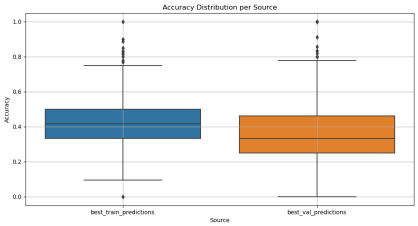


*Figure 1.* Accuracy distribution per User ID for each set: Transformer model evaluated on the Training set (Blue) and Validation set (Orange).

Since accuracy alone may not fully reflect recommendation quality, NDCG was chosen as a more suitable evaluation metric. NDCG assesses the ranking quality of recommended items by prioritizing the position of relevant movies in the recommendation list. The boxplot below illustrates that the NDCG scores are significantly higher than accuracy, suggesting that the model captures user preferences better in terms of ranking rather than absolute rating prediction.

| | |
|---|---|
| Gini and Accuracy | -0.5321 |
| Accuracy and Avg NDCG | 0.3916 |

*Table 1.* Correlations

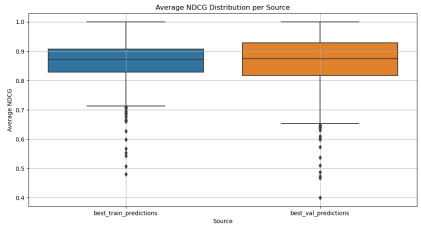


*Figure 2.* Average NDCG@k where k is = [3, 5, 10, 15, 20, 50, 100]. Blue and Orange represents the evaluation on training and validation set respectively using vanilla transformer model.

To further investigate user rating behavior, the Gini Index was computed for each user in both the training and validation sets. The Gini Index measures rating dispersion, where higher values indicate more skewed distributed ratings.

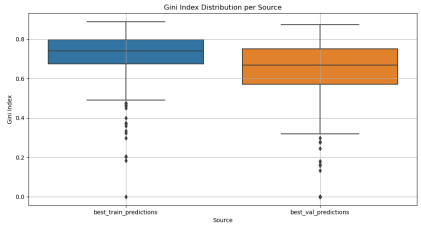


*Figure 3.* Gini Index distribution per User ID for each set: Transformer model evaluated on the Training set (Blue) and Validation set (Orange).

Additionally, the relationships between the Gini Index, accuracy, and NDCG were examined using Spearman and Pearson correlation coefficients. The results, summarized in the table below, indicate a negative Spearman correlation between Gini Index and accuracy $(-0.5321)$, suggesting that users with more skewed rating distributions tend to have lower prediction accuracy.

Meanhile, a positive Pearson correlation $(0.3916)$ is observed between accuracy and average NDCG, reinforcing the model's ability to rank user preferences effectively.

### 4.2. Policy Evaluation

To assess the effectiveness of the trained policy model, its performance was compared to a random policy baseline, where a movie is selected uniformly from the Top-25 list. The evaluation reveals that while the policy model assigns high rewards to certain users, it fails to generalize well for

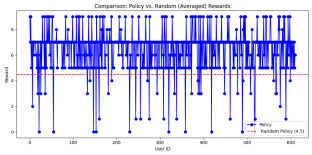


*Figure 4.* Comparison between the random policy and the trained policy model

others. This suggests that while the policy improves recommendations for some, additional refinements are needed for broader applicability. A more robust evaluation would ideally involve real user interactions rather than simulated data.

### 4.3. Do Agents Make Better Reward Models?

The accuracy distribution of the transformer model and AI agents was evaluated on the validation set, considering only the users shared between both models due to budget constraints related to GPT-4o API endpoint costs. The results, visualized in the figure below, indicate that while the transformer model achieves higher accuracy, AI agents may struggle to align precisely with user rating values. This discrepancy could stem from the subjective nature of user ratings.

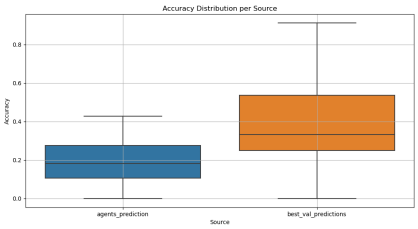


*Figure 5.* Accuracy distribution per User ID for each set: Transformer model and AI Agents evaluation on Validation set only for the users that they share in common

Despite lower accuracy, AI agents may offer advantages in capturing broader user preferences. The transformer model's superior accuracy is likely due to its ability to memorize and fit to the training data distribution. However, this reliance on pattern recognition may limit its effectiveness when encountering out-of-distribution data, where AI agents might generalize user behavior more effectively and this is evaluate when the average NDCG was computed.

The average NDCG@k per user was computed for both models. The figure below illustrates the comparison, showing how each model ranks recommendations relative to user preferences as the median from the agents is higher than the median from the transformer reward model.

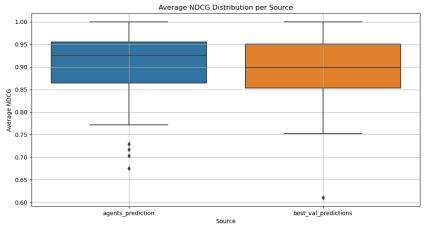


*Figure 6.* Comparison of agents (Blue) and transformer model (Orange) in terms of Average $NDCG@k$ per user.

## 5. Conclusion and Future Considerations

- AI agents have demonstrated potential as a replacement for traditional transformer-based reward models; however, further experimentation is needed to validate their effectiveness.
- Enhancing prompt design could lead to better performance, and enabling agents to interact with each other rather than relying on a single agent to represent an individual user could provide more reliable results.
- Instead of modeling individual users, agents could be designed to represent user clusters, capturing broader behavioral patterns and improving personalization.
- Replace the transformer-based reward model in the proposed framework with AI agents and assess the performance of the updated policy.

## References

Cao, Y., Zhao, H., Cheng, Y., Shu, T., Chen, Y., Liu, G., Liang, G., Zhao, J., Yan, J., and Li, Y. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2024. doi: 10.1109/TNNLS.2024.3497992.

Corecco, N., Piatti, G., Lanzendörfer, L. A., Fan, F. X., and Wattenhofer, R. Suber: An rl environment with simulated human behavior for recommender systems, 2024. URL https://arxiv.org/abs/2406.01631.

Harper, F. M. and Konstan, J. A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), December 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL https://doi.org/10.1145/2827872.

Salem, P., Olsen, C., Freire, P., Ding, Y., and Saxena, P. Tinytroupe: Llm-powered multiagent persona simulation for imagination enhancement and business insights. https://github.com/microsoft/tinytroupe, 2024. GitHub repository.

Schoenegger, Philipp, S. G. A. G. J. L. and Caviola., L. Ai can outperform humans in predicting correlations between personality items. *Communications Psychology*, 3(1):23, 2025. doi: https://doi.org/10.1038/s44271-025-00205-w.

Yu, P., Xu, Z., Wang, J., and Xu, X. The application of large language models in recommendation systems, 2025. URL https://arxiv.org/abs/2501.02178.