Turning Pages with Reinforcement: A Markov Decision Process Approach to Book Recommendations

Ori Spector
Computer Science
Stanford University
Stanford, CA
orispec@stanford.edu

Pannisy Zhao
Computer Science
Stanford University
Stanford, CA
pannisy@stanford.edu

***Abstract*—This paper presents a Markov Decision Process (MDP) modeling approach to address the sequential decision-making problem of recommending books to users based on their historical interactions. We reduce the dimensionality of the state and action spaces through K-means clustering of users and books, respectively. A reward function is defined based on user interactions, capturing the quality of recommendations. We then apply Q-learning to derive a policy that aims to maximize cumulative rewards. Experiments demonstrate the effectiveness of the proposed method in aligning recommendations with user preferences, outperforming a random policy baseline.**

***Index Terms*—Book Recommendation Systems, Markov Decision Processes, Reinforcement Learning, Q-Learning, K-Means Clustering, User Interaction Data**

I. Introduction

In the era of digital information overload, readers are faced with an overwhelming number of book choices on platforms like Goodreads and Amazon Kindle. Personalized recommendation systems are essential for helping users discover books that align with their interests, enhancing their reading experience.

Traditional recommendation methods, such as collaborative filtering and content-based filtering, often fall short in capturing the dynamic and sequential nature of user preferences. Reinforcement learning (RL) offers a promising avenue by modeling the recommendation task as a sequential decision-making problem, where the system learns to adapt its recommendations based on user feedback over time.

This paper proposes an RL-based approach using Markov Decision Processes (MDPs) to model the book recommendation problem. By leveraging K-means clustering, we reduce the complexity of the state and action spaces, making the problem computationally tractable. We define a reward function that reflects user satisfaction derived from interaction data and apply Q-learning to learn an optimal recommendation policy.

II. Literature Review

There are several relevant research papers that explore the use of Q-learning, MDPs, and related RL techniques for book recommendation systems.

A. MDP-Based Recommender Systems

Foundation work from Shani et al. [1] proposed using MDPs to model the recommendation process as a sequential decision problem rather than a static prediction task. Their approach offers two main benefits: 1) It considers the long-term effects of each recommendation and 2) It accounts for the expected value of recommendations.

The authors developed an MDP model initialized with a predictive model and deployed it on a commercial site, demonstrating its practical viability.

B. Q-Learning for Book Recommendation

There has been research that applied Q-learning and Deep Q-learning to build a movie recommendation system using the Netflix Prize dataset [2]. While not specifically for books, their approach could be adapted for book recommendation. By leveraging reinforcement learning to capture the dynamic nature of user preferences, they were able to address limitations of traditional recommendation systems like data sparsity and cold start. Moreover, they demonstrated that quality recommendations can be learned from user-item interaction data.

C. RL for Personalized Recommendations

Several papers explore RL approaches for personalized recommendations that could be applied to books. For instance, Zhao et al. [4] developed a pairwise deep reinforcement learning method to incorporate negative feedback in recommendations. Additionally, Wang et al. [3] proposed a clustering-based hierarchical reinforcement learning approach to handle data sparsity in digital library book recommendations.

D. Book Recommendation Systems

Furthermore, Du et al. [5] designed a collaborative filtering algorithm that addresses data sparseness and cold start problems by improving similarity calculations and data filling methods. Their approach incorporates user common rating weights into similarity calculations and employs a hierarchical clustering method based on user attributes and Euclidean distance for data filling. The algorithm also utilizes the Slope-One algorithm with weighted degrees to determine final filling values, resulting in improved recommendation accuracy when tested on the Book-Crossing dataset.

In summary, these papers demonstrate the potential of Q-learning, MDPs, and other RL techniques to improve book recommendation systems by capturing sequential decision-making, addressing common challenges, and optimizing for long-term user satisfaction. The field is active, with ongoing research exploring novel algorithms and hybrid approaches.

III. Problem Formulation

We model the book recommendation problem as a Markov Decision Process (MDP), represented by the tuple (S, A, T, R, γ), where:

- S is the set of possible user states, represented by user clusters obtained through K-means clustering of user features.
- A is the set of possible actions, corresponding to book clusters obtained by clustering book features.
- $T(s'|s,a)$ is the probability of transitioning from state s to s' after taking action a. In our simplified model, we assume $T(s'|s,a) = 1$ if $s' = s$ and 0 otherwise, as user states are considered static during interactions.
- $R(s,a)$ is a deterministic function that assigns a reward based on the user's interaction with the recommended book.
- γ is a discount factor between 0 and 1 that determines the importance of future rewards.

The MDP for the book recommendation problem is formulated below.

A. State Space

The state space S represents the different types of users in our system. To manage the complexity of modeling individual user preferences, we reduce the dimensionality of the state space by clustering users based on their features. We apply K-means clustering to user feature vectors, which include:

- Average Rating Given: The mean of the ratings a user has assigned to books.
- Number of Books Read: Total count of books the user has read.
- Preferred Genres: Encoded as a sparse vector indicating the genres the user frequently reads.

Each cluster $s \in S$ groups users with similar reading behaviors, effectively summarizing their preferences. Mathematically the state space is $S = \{s_1, s_2, ..., s_n\}$ where n is the number of user clusters.

B. Action Space

The action space A consists of possible recommendations we can make, represented by clusters of books. We cluster books based on their features to reduce the action space's dimensionality. Book features include:

- Average Rating: The average rating the book has received.
- Number of Pages: Length of the book.
- Publication Year: When the book was published.
- Genres: Encoded as a sparse vector similar to user genres.

Each action $a \in A$ corresponds to recommending a book from a specific book cluster. Formally, the action space is $A = \{a_1, a_2, ..., a_m\}$ where m is the number of book clusters.

C. Reward Function

The reward function $R(s,a)$ quantifies the quality of a recommendation based on user interaction. It is designed to encourage recommendations that the user reads and rates highly. The reward is assigned as follows:

$$R(s,a) = \begin{cases} -1, & \text{if book is not read} \\ 1, & \text{if book is read but not rated} \\ 5, & \text{if book is read and rated 1-2 stars} \\ 10 + \text{rating}, & \text{if book is read and rated 3-5 stars} \\ 0, & \text{otherwise} \end{cases}$$

The reward function captures user satisfaction, providing higher rewards for recommendations that lead to positive interactions.

D. Objective Function

Our goal is to find an optimal policy π^* that maximizes the expected cumulative reward:

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]$$

where s_t is the state at time t, $a_t = \pi(s_t)$ is the action taken in state s_t following policy π, and $\gamma \in [0,1]$ is the discount factor that determines the importance of future rewards.

By solving this optimization problem, we aim to derive a policy that consistently recommends books aligning with user preferences, thereby enhancing user satisfaction.

IV. Approach

A. Data Preprocessing

We utilized two primary datasets from Goodreads [6] [7]. First, a books dataset that contains metadata for books, including book ID, title, authors, average rating, number of pages, publication year, and popular shelves (genres). The second dataset includes user interactions with books, such as user ID, book ID, whether the book was read (is_read), and the user's rating.

Furthermore, we utilized feature extraction. For each book, we extracted numerical features (average rating, number of pages, publication year) and encoded genres using a sparse binary matrix. Genres were mapped using a genre index built from the most common genres in the dataset. For each user, we calculated the average rating given, the number of books read, and aggregated genres from the books they have interacted with.

B. Clustering

We applied K-means clustering on feature vectors to group users into n clusters (states). Each cluster represents a distinct user profile. We also similarly applied K-means clustering on book feature vectors to group books into m clusters (actions). Each cluster groups similar books.

C. State-Action-Reward Tuples

Using the interactions data, we constructed state-action-reward tuples $(s, a, R(s, a))$ for training state s, the cluster to which the user belongs, action a, the cluster to which the interacted book belongs, and reward $R(s, a)$, calculated based on the user's interaction with the book, as defined in the reward function.

D. Q-Learning Algorithm

We employed Q-learning, an off-policy temporal-difference control algorithm, to learn optimal policy. First, we initialized the Q-table $Q(s, a)$ with zeros for all state-action pairs. Then, the Q-value updates are performed using the Bellman equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s,a) + \gamma \max_{a'} Q(s', a') - Q(s,a)]$$

Since $s' = s$ in our model, the update simplifies to:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s,a) + \gamma \max_{a'} Q(s, a') - Q(s,a)]$$

where α is the learning rate and γ is the discount factor.

A ϵ-greedy policy is used for action selection during training:

$$\pi(s) = \begin{cases} \text{random action}, & \text{with probability } \epsilon \\ \arg\max_a Q(s,a), & \text{with probability } 1-\epsilon \end{cases}$$

E. Recommendation Process

For a given user, we identify the user's state cluster s, select an action a using the learned policy $\pi(s)$, and recommend a random book from the selected book cluster corresponding to action a.

F. Evaluation Methodology

We evaluated the recommendations through an alignment check, determining if the recommended book's genre aligns with the user's preferred genres, and a success metric. A recommendation is considered successful if there is significant genre overlap. We compared the learned policy with a random policy that selects actions uniformly at random.

V. Experiments

A. Experimental Setup

Due to compute constraints, we decided to focus on a specific genre subset 'Children's Books' instead of the entire dataset. Therefore, we conducted experiments using $124,082$ books and $\approx 25,000$ users in the subset from our Goodreads dataset. The experiments were designed to evaluate the effectiveness of our MDP-based recommendation system under different clustering configurations and training parameters. We maintained consistent hyperparameters across experiments with $\gamma = 0.9$, $\alpha = 0.1$, and $\epsilon = 0.1$.

B. Clustering Configurations

We evaluated three different clustering configurations to understand the impact of state-action space granularity:

- Configuration 1: 500 book clusters, 50 user clusters
- Configuration 2: 200 book clusters, 25 user clusters
- Configuration 3: 75 book clusters, 5 user clusters

For each configuration, we trained the Q-learning agent for 1,000 epochs. Additionally, we conducted an extended training experiment with 5,000 epochs on the best-performing configuration.

C. Evaluation

We evaluated both the learned policy and random policy using a systematic approach:

- **Test Set**: 3,000 randomly selected users from the dataset
- **Alignment Check**: For each user, we:
  - Extract user's preferred genres from historical interactions
  - Generate recommendation using either learned or random policy
  - Compare recommended book's genres with user's preferred genres
  - Consider recommendation successful if significant genre overlap exists
- **Policies Compared**:
  - Learned Policy: Uses Q-learning to select book clusters based on user state
  - Random Policy: Randomly selects book clusters with uniform probability

D. Example Recommendations

To illustrate the evaluation process, we present two representative cases:

User ID: a50c149f424cbc8443cd5ee41e6ce950

- Top Genres: fairy-tales, fantasy, magic, adventure, classics
- Recommended Book: "The Faraway Tree Collection"
- Book Genres: fantasy, classics, children, adventure, magic
- Result: Strong alignment between user preferences and recommendation

This case demonstrates successful genre alignment, with multiple overlapping genres (fantasy, classics, adventure, magic) between user preferences and the recommended book.

User ID: d594a2c0d47f7eca636801b5c1349850

- Top Genres: love, romance, British, vampire, realistic
- Recommended Book: "Harry Potter and the Prisoner of Azkaban"
- Book Genres: sci-fi, fiction, adventure, British, coming-of-age
- Result: Limited alignment with only "British" as common genre

This case illustrates a misalignment where the system recommended a fantasy book to a user who primarily reads romance and realistic fiction.

VI. Results and Analysis

A. Clustering Configuration Performance

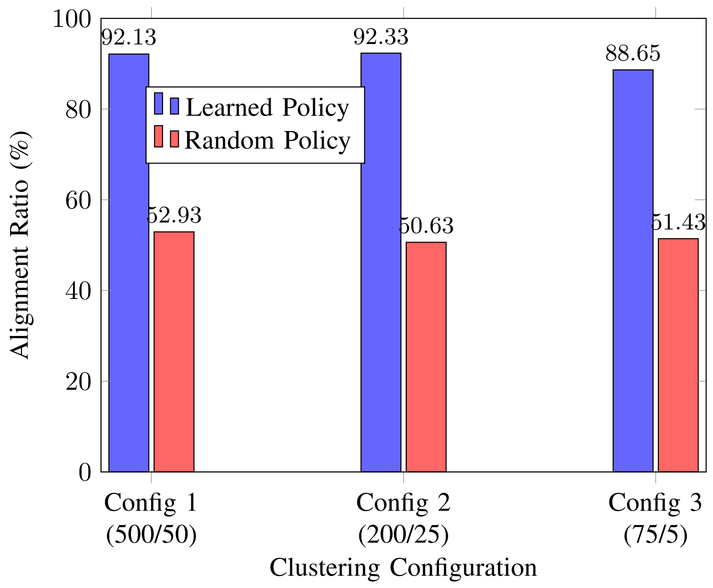

Fig. 1. Comparison of alignment ratios across different clustering configurations for 3000 random users

B. Performance Analysis

1) Clustering Configuration Performance: As shown in Figure 1, our experiments with different clustering configurations revealed:

- Configuration 1 (500/50): 92.13% alignment with learned policy vs 52.93% random
- Configuration 2 (200/25): 92.33% alignment with learned policy vs 50.63% random
- Configuration 3 (75/5): 88.65% alignment with learned policy vs 51.43% random
  Note that we decided to use higher clusters number for books due to the amount of the data compared to number of users.

2) Policy Comparison: The learned policy consistently outperformed the random baseline by approximately 40 percentage points across all configurations. This substantial improvement demonstrates that:

- The Q-learning algorithm effectively captures user preferences
- The MDP framework successfully models the sequential nature of recommendations
- The clustering approach maintains sufficient information while reducing dimensionality

C. Key Findings

1) Optimal Clustering Configuration: Configuration 2 (200 book clusters, 25 user clusters) achieved the best performance with a 92.33% alignment ratio, while maintaining reasonable computational efficiency with a training time of 173 seconds. This suggests that this configuration provides an optimal balance between model complexity and performance.

2) Impact of Cluster Granularity: We observed that:

- Fine-grained clustering (Configuration 1) achieved similar performance (92.13%) but required more computational resources
- Coarse clustering (Configuration 3) showed degraded performance (88.65%), though with faster training time
- The medium granularity (Configuration 2) provided the best trade-off between performance and computational efficiency
- Random policies performed better than expected due to sub-genres being relatively popular among the dataset.

3) Extended Training Analysis: When extending the training epochs to 5,000 for Configuration 2, we observed an improvement in alignment ratio to 94.30%, suggesting that additional training can yield marginal improvements in recommendation quality.

VII. Conclusion

The experimental results demonstrate that our MDP-based recommendation system effectively learns user preferences and significantly outperforms random recommendations. The optimal configuration with 200 book clusters and 25 user clusters provides a good balance between model complexity and performance, achieving over 94% alignment (when trained for 5k epochs) with user preferences while maintaining reasonable computational efficiency. However, there are several limitations present in our work:

- Our experiments were limited to the Children's Books subset due to computational constraints, potentially limiting generalizability to other genres
- Static State Assumption: The current model assumes user preferences remain static, which may not reflect real-world dynamics of evolving reading tastes
- While clustering reduces complexity, it may oversimplify individual user preferences and book characteristics
- The model doesn't fully utilize the temporal sequence of user interactions, which could provide valuable context for recommendations

Future work could address these limitations through:

- Expanding the model to handle the full dataset across all genres
- Incorporating dynamic state transitions to capture evolving user preferences
- Exploring more sophisticated clustering techniques or alternative dimensionality reduction methods
- Implementing sequence modeling to leverage temporal patterns in user interactions
- Investigating hybrid approaches that combine MDP-based recommendations with traditional collaborative filtering methods

Despite these limitations, our results demonstrate the potential of MDP-based approaches in book recommendation systems, providing a foundation for future research in this direction.

Team Collaboration

Our collaborative project benefited from the commitment and efforts of each team member, ensuring its success. Pannisy contributed fully in alignment with her 3-unit enrollment. Ori, enrolled for 4 units, dedicated an additional 30 hours to the project. We are pleased with the strong team dynamics and the productive outcomes achieved together.

References

[1] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-Based Recommender System," J. Mach. Learn. Res., vol. 6, pp. 1265–1295, December 2005.

[2] M. Rezaei, "Reinforcement Learning based Recommender System using Q-Learning and Deep Q-Learning," East Carolina University, Jul. 2022. [Online]. Available: http://hdl.handle.net/10342/11122

[3] X. Wang, Y. Wang, L. Guo, L. Xu, B. Gao, F. Liu, and W. Li, "Exploring Clustering-Based Reinforcement Learning for Personalized Book Recommendation in Digital Library," Information, vol. 12, no. 5, pp. 198, 2021.

[4] X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, and J. Tang, "Deep reinforcement learning for page-wise recommendations," in Proceedings of the 12th ACM Conference on Recommender Systems, pp. 95–103, September 2018.

[5] Y. Du, L. Peng, S. Dou, X. Su, and X. Ren, "Research on Personalized Book Recommendation Based on Improved Similarity Calculation and Data Filling Collaborative Filtering Algorithm," Comput Intell Neurosci., vol. 2022, pp. 1900209, September 2022.

[6] M. Wan, J. McAuley, "Item Recommendation on Monotonic Behavior Chains," in Proceedings of the 12th ACM Conference on Recommender Systems (RecSys'18), pp. 86–94, October 2018.

[7] M. Wan, R. Misra, N. Nakashole, J. McAuley, "Fine-Grained Spoiler Detection from Large-Scale Review Corpora," in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL'19), pp. 2605–2610, July 2019.