Course Project

Code of Honor. All external resources used in the project, including research papers, open-source repositories, datasets, and any content or code generated using AI tools, e.g., ChatGPT, GitHub Copilot, Claude, Gemini, must be *clearly cited* in the final submission. The final report must also include *a clear breakdown of individual group member contributions.* Any lack of transparency in the use of external resources or in reporting group contributions will be considered academic dishonesty and will significantly impact the final evaluation.

Topic	Using RL to Build a Recommendation System
Category	Applications of RL

OBJECTIVE Design and implement a reinforcement learning agent for sequential recommendation. The agent should recommend items (e.g., movies or products) to maximize long-term user engagement. The project aims to compare RL-based policies with traditional recommendation baselines such as collaborative filtering or random recommendation.

MOTIVATION Traditional recommendation systems typically rely on supervised learning methods that optimize short-term metrics like click-through rate or rating prediction accuracy. However, these approaches often ignore the long-term effects of recommendations on user satisfaction and engagement. Reinforcement learning offers a natural way to model recommendation as a sequential decision-making problem, where each action (recommendation) affects future states (user preferences) and rewards (engagement). Recent studies [1, 2, 3] demonstrate the effectiveness of RL in recommendation tasks. This project gives students the opportunity to apply RL to a real-world application area that directly impacts modern industry systems such as Netflix, YouTube, and Spotify.

REQUIREMENTS The final submission should address the following requirements while the details can be freely decided by the group members.

1. Implementation: in this respect, you should
   - build a recommendation environment based on a small dataset (e.g., MovieLens) or synthetic user profiles,
   - implement an RL agent (e.g., DQN, PPO) that recommends items based on user states,
   - implement simple baselines (random recommender, collaborative filtering) for comparison.
2. Environment modification: you can use a pre-implemented environment. To give the implementation some level of novelty, you **must** modify the environment with **at least one** of the following modifications:
   - adding noise to user feedback (simulating imperfect data),
   - penalizing repetitive recommendations,
   - introducing distractor features in user profiles,

- reward shaping (e.g., rewarding both engagement and diversity).

3. Evaluation: the final project should report key evaluation of the implemented algorithms in the modified environment. In this respect, the results should
   - compare long-term cumulative engagement (reward) across methods,
   - evaluate diversity and novelty of recommendations, and
   - report how environment modifications affect agent performance.
4. The results should be elaborated through
   - performing ablation studies, e.g., removing diversity penalty, different reward functions, and
   - providing discussion on trade-offs between short-term vs long-term optimization.

MILESTONES The following milestones are to be accomplished through semester.

1. Literature Review and Setup
   - Review RL-based recommender systems.
   - Select dataset and define user simulation or reward model.
2. Implementation
   - Implement baseline recommenders.
   - Implement RL agent and validate on simple simulated users.
   - Extend to modified environments.
3. Evaluation and Analysis
   - Collect and plot engagement metrics.
   - Compare RL-based recommendations with baselines.
   - Perform ablation experiments.
4. Final Report and Presentation

SUBMISSION GUIDELINES The main body of work is submitted through Git. In addition, each group submits a final paper and gives a presentation. In this respect, please follow these steps.

- Each group must maintain a Git repository, e.g., GitHub or GitLab, for the project. By the time of final submission, the repository should have
  - Well-documented codebase
  - Clear `README.md` with setup and usage instructions
  - A `requirements.txt` file listing all required packages or an `environment.yaml` file with a reproducible environment setup
  - Demo script or notebook showing sample input-output
  - *If applicable,* a `/doc` folder with extended documentation
- A final report (maximum *5 pages*) must be submitted in a PDF format. The report should be written in the provided formal style, including an abstract, introduction, method, experiments, results, and conclusion.
  Important: Submissions that do not use template are considered *incomplete.*
- A 5-minute presentation (maximum *5 slides including the title slide*) is given on the internal seminar on Week 14, i.e., *Dec 1 to Dec 5,* by the group. For presentation, any template can be used.

Final Notes While planning for the milestones please consider the following points.

1. Students are encouraged to experiment with different reward designs (e.g., combining engagement and diversity).
2. Training should remain feasible by limiting item sets and using lightweight models.
3. Creativity in simulating users and shaping rewards is encouraged as long as the core objectives are met.
4. Teams are expected to manage their computing needs and are advised to perform early tests to estimate runtime and training feasibility. As graduate students, team members can use facilities provided by the university, e.g., ECE Facility. Teams are expected to inform themselves about the limitations of the available computing resources and design accordingly.

References

[1] M Mehdi Afsar, Trafford Crump, and Behrouz Far. Reinforcement learning based recommender systems: A survey. *ACM Computing Surveys*, 55(7):1–38, 2022.

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

[3] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM conference on recommender systems*, pages 95–103, 2018.