Personalized Teaching Methods Through Trained Reward Models: Final Report

Rajat Sethi, *Clemson University*

1. Abstract

In the paper I originally reviewed, Reddy et al. built a special reinforcement learning model for an autonomous vehicle simulation. Their research particularly covered morally ambiguous scenarios, where different developers may follow a different ethics code. In addition, the navigation system was able to detect a number of "unsafe" states and avoid them before attempting to make another move.

Unfortunately, there were a number of problems with the source code. The repository itself had very little documentation, contracts, or interfaces to work with, making replication quite difficult. The models also only worked on 2D navigation tasks, which made applying and improving the study a great challenge.

With the code I was able to understand, I have been working on a system that moves away from the 2D navigation experiments. In my research, I have developed an educational interface that recommends learning materials for the user. Through reinforcement learning and the reward model, this system evaluates the user's progress and helps them learn the material that they need.

2. Background

The ReQueST Model developed by Reddy et al. was a new approach to standard reinforcement learning (RL). In normal reinforcement learning, an agent takes a snapshot of its current circumstances, and the system will reward/punish the agent depending on how favorable the situation is. The agent will perform actions in response to the reward/punishment. If the agent was rewarded, it will learn from the experience and continue seeking out more rewards. If the agent. Likewise, if the agent was punished, it will attempt to rectify its mistake and seek out a higher reward.

In standard RL, the reward function is often clear-cut. For example, in Chess, an agent can be rewarded for gaining pieces, and punished for losing pieces. However, some scenarios may be morally ambiguous with respect to rewards. Using the example in the ReQueST paper, the trolley problem can be approached differently based on the developer's philosophical beliefs. To account for this, Reddy et al.trained an Artificial Neutral Network (ANN) to serve as the reward function. That way, the system could "learn" about the developer's ethical objectives and pass that onto the agent.

The other novelty that this paper provides is its use of trajectory optimization. In autonomous driving, there are many :unsafe" states that must be avoided. This could include anything from pedestrians to buildings to ditches. As such, it is imperative that the agent "predicts" what will happen next based on its current trajectory. If the vehicle rewards/punishes itself after the damage has already been done, then the consequences would be catastrophic leading to the loss of many lives. In a sense, the agent must reward/punish itself prematurely before taking any actions in a particular direction.

3. Changes from Midterm Report

In my midterm report, I had originally attempted to implement the ReQueST Model to train an agent to play Blackjack. However, there were a few flaws in this approach, both from the model I pulled and from my own mistakes. As mentioned earlier, the repository was uncommented and mostly undocumented. This made changing parameters and implementing functions very difficult. In addition, the model itself only worked for 2D navigation problems. This would make running anything else impossible without a complete overhaul.

On my end, my mistake was attempting to implement Blackjack using ReQueST. There were two main flaws with my approach. First, the game itself was not as morally ambiguous as I initially thought. My idea was that the risk of gambling could have different effects on different users. The problem is that at the end of the day, the reward can be simply described as "money earned", negating the need for training a reward model. The other problem is getting a proper output. Since ReQueST only works on 2D Navigation problems, I would have to create a new environment and output method. Unfortunately, this never came to fruition because the models were too encapsulated. With these flaws in mind, I have determined that the only way to move forward is to make these models, environments, and outputs from scratch.

4. Project Objectives

For this phase, my main objective is to use the ideas from ReQueST and implement them in a non-navigation fashion. Specifically, for this project I would like to create an education system that is trained to understand what the user wants to learn about. Once it has a grasp of the number and types of questions, the system will move forward and "teach" the user.

Using the trained reward model, the agent will recommend lessons for the user depending on what material needs to be covered next. After the lesson, the user will be quizzed on the material and the number of correct answers will be stored. For the sake of simplicity, this process will repeat indefinitely. Through this system, I hope to make a "smart" teaching platform that is personalized towards the user.

5. Reinforcement Learning Agent

RL algorithms are composed of 4 different components:

- Action – The possible steps that an agent can take to change the environment.
- Environment – The interface between the agent, user, and data.
- State – The current configuration of the environment and its related data.
- Reward – A number that quantifies how accurate/inaccurate the agent's actions are. The ultimate goal of the model is to maximize the rewards.

5.1. Actions

For simplicity, there are only three actions that the agent can take. Give a "literature" lesson, a "history" lesson, or an "art" lesson. Once the user is given the lesson, they will read the recommended passage and answer the quiz questions.

5.2. Environment

The environment is different from most algorithms. One could say that the environment is the user's mind, where the actions can improve the user's knowledge of a specific subject. The environment would also include the user/agent interface where the user receives lessons and quizzes.

5.3. State

The state can be identified by the overall accuracy of the user's quizzes. How many literature, history, or art questions did the user get right compared to the number of total questions asked.

5.4. Reward

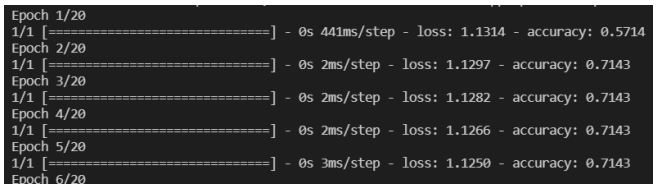
The reward is a comparison of the total accuracy of each subject against the expected accuracy. If the user is outperforming their expectations, then the agent will be rewarded and move onto other subjects. If the user is underperforming in a particular subject, then that field will be prioritized. The "expected" values are determined by a previously trained reward model.

6. Reward Model

To understand what the user wants to learn, they are given multiple "100-question" scenarios. Imagine that a course consisted of 100 questions, and the subjects were split up in a random manner. The user would determine which subjects needed more attention, based on the hypothetical source. In this case, the percentage of questions would act as the input layer, and the user's labeling would create the output layer. In practice, the model would receive the current number of correct questions from the user in each subject, then use a softmax function to determine the reward.

7. Results

Running the reward model under a small sample size yielded a 71% accuracy rate.



Due to the time constraints and resource limits, running a full accuracy test on the reinforcement model was not possible. However, testing the RL algorithm was successful, as the lessons provided matched the output of the reward model.

This section will be expanded upon in future research attempts with a proper quantitative analysis. In the meantime, please explore the GitHub page and run the RL algorithm to see how the system works.

8. Discussion and Future Endeavors

The reward model is the key aspect of both the ReQueST paper and this project. Creating a proper training set with a large enough sample size is extremely important and must be improved in later improvements. In addition, other reinforcement learning algorithms like Q-learning should be incorporated into the system.

Overall, there are two important takeaways from this experience. First, is the idea of reward modeling. When a reward function is ambiguous, training a neural network to create a reward model is a valid methodology that works in many scenarios. Second, is the idea of proper documentation. Writing a research paper is important, but its even more critical to have functioning, documented code. Without examples and guidelines, code replication and innovation would be nearly impossible.

9. Works Cited

Inc., Lazy Programmer, and Lazy Programmer Team. "Tensorflow 2.0: Deep Learning and Artificial Intelligence." *Udemy*, Udemy, 21 Nov. 2021, https://www.udemy.com/course/deep-learning-tensorflow-2/.

Reddy, S, Dragan, A.D., Levine, S, Legg, S, and Leike, J. *Learning Human Objectives by Evaluating Hypothetical Behavior*. Retrieved from https://par.nsf.gov/ biblio/10183732. *International Conference on Machine Learning (ICML)* .