Building a Game Agent to Play Avalon

Orien Zeng
Stanford University
ozeng@stanford.edu

Albert Tung
Stanford University
atung3@stanford.edu

1 Problem

The game Avalon is a zero-sum limited knowledge game in which players are assigned to one of two teams, "good" and "bad". For each of five rounds, the players choose a set number of players to go on a "mission". The selected players can then secretly choose to "pass" or "fail" the mission, with usually one player failing the mission resulting in mission failure. After the players on the mission choose whether to pass or fail, it is revealed how many passes and fails the team chose. The good team's objective is to pass 3 of 5 missions, and conversely the bad team's objective is to fail 3 of 5 missions. If at least 3 of the 5 missions pass, the bad team has a chance to guess who the "Merlin" player is, and if they guess successfully, they win the game.

1.1 Mission Selection

For each mission, up to 5 players take turns proposing a lineup for that mission and the number of members on the mission are determined by the number of overall players. For each proposal, all players publicly vote at the same time for whether they want that lineup. If a strict majority vote in favor, then the selected lineup is used for the mission. Otherwise, the next player in line proposes the lineup. If all 5 proposals are turned down, then the mission is considered a failure.

1.2 Characters

- Merlin is a good character who knows who all of the bad people are except Mordred, but does not know which bad character has which role.
- Percival is a good character who knows who Merlin and Morgana are, but not which is which (so must decide which is Merlin).
- Mordred is a bad character who is unique in that Merlin does not know who Mordred is.
- Morgana is a bad character who Percival sees at the beginning of the game, but Percival does not know whether that character is Merlin or Morgana.
- All bad characters know who the other bad characters are.
- The rest of the characters have no special information at the beginning of the game.

Traditionally, Avalon allows players to communicate and discuss what approaches to take in the game. For the purposes of this project, communication is not allowed as this would add additional natural language processing which may not be feasible. Thus, agents only receive information according to the rules of the game, e.g. public mission proposals and voting, whether missions pass or fail, and special roles.

The goal is to build an agent to play Avalon successfully which may require the agent to be able to have the ability to play any character.

2 Evaluation Metric

Success in a given game of Avalon is purely determined by whether the player's team wins or loses the game. We can arbitrarily assign a utility of 1 to each agent on the winning team, and a utility of 0 to each agent on the losing team.

The success of Avalon agents will be computed as a relative score between two models. The good team will be instantiated with multiple instantiations of the same model, and the bad team will be instantiated using a different model. Then, the teams will switch and an equal number of games will be played. The score for each model will be the portion of games that they win, with the sum of both model's scores equal to 1.

A second scoring model involves not placing identical players on the same team. That is, each player has equal chance of being either model, regardless of what team the player is on. The evaluation will then involve playing many games where the teams are completely randomized.

We can score the models researched in this project against two fixed models: the first is a random model, and the second is a skilled human player. As we continue developing our approaches, we may consider running the different models against each other to test the effectiveness of each.

3 Approaches

3.1 Bayesian Model

In the Bayesian model, the game is modeled as a sequence of decision nodes and information nodes. For example, an information node would represent the number of failures on mission 3, and a decision node would represent whether player 1 fails mission 3. The parameters of this network are probabilities for the decision nodes (a set of parameters for the decision nodes represents a mixed strategy), and games are played out by sampling from starting positions and from the decision nodes. From there, we can incentivize moves that win by increasing the probability of the decisions that were made by the winning team.

3.2 SARSA Q-Learning [2]

In Q-Learning, the algorithm attempts to take information about the game through SARSA (state 1, action 1, reward, state 2, action 2) which are experiences in which the game agent sees itself in state 1, chooses an action, and subsequently gets a reward. The action then leads to the following state and action. It attempts to greedily find the most optimal policy given a state to maximize its reward through estimating the value of the Q function. For the purposes of our experiments, we can possibly represent the states as the voting history of the entire lineup for missions as we believe that humans would need to use the information of voting history to make decisions on what action they should take. The reward we can have is whether or not the mission has passed as well as whether or not the agent's team ends up winning the game.

However, research recently has revealed that Q-Learning may have some shortcomings, partly due to its greedy nature. One article by Hasselt et al. asserts that Q-Learning often overestimates action values in different conditions and that these overestimations can lead to suboptimal play [1]. Despite this, Q-Learning is a popular reinforcement learning which may sufficient enough to find an optimal strategy for Avalon.

4 Challenges

We foresee that building a reinforcement learning agent that has the ability to learn solely on self-playing and be able to participate in multiple different actions will be an incredible challenge, especially for a game that has yet to be explored.

1. Different stages of the game that have different actions which include voting and choosing an optimal lineup
2. A large state space due to the amount of possible lineups and voting patterns
3. Understanding risk-taking and risk management
4. Opponent modeling which must be dynamic since humans can change strategies
5. Making the AI unpredictable such that opponents cannot anticipate its reactions

References

[1] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, pages 2094–2100, 2016.

[2] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, May 1992.