University of California, Los Angeles
CS 260 Machine Learning Algorithms

Instructor: Cho-Jui Hsieh
Student: Brandon Hua
Date: March 17, 2021

Reinforcement Learning for Autobattler AI

1 Motivation

Autobattler is a genre of strategy video games that has recently been popularized, with large companies like Riot Games and Valve Corporation releasing their versions of the game in 2019. In an autobattler, two players face each other on a chessboard-like field. During a preparation phase, each player places units onto the side of the board reserved specifically for their units. When a round starts, the units battle each other until one side is victorious (i.e. the other side is all eliminated).

A common complaint about autobattler games is that the units act unreasonably at times. For many games, the default AI of a unit follows the simple pattern of seeking the nearest enemy and attacking it, which is clearly not optimal. The actions of a player's key unit can turn the tides of a round easily. For example, if a player's main damage-dealing unit paths dangerously into the enemy frontline or decides to focus an enemy tank unit, the player will likely suffer a loss for that round.

A smarter AI would be able to reduce the number of instances where a naive AI causes unexpected results through taking insensible actions. This aims to reduce player frustration. In addition, smarter AI would make pieces' actions less straightforward/predictable and make for more exciting rounds. Overall, a smart autobattler AI contributes towards complex and dynamic gameplay, and (hopefully) a more enjoyable experience for the player.

2 Methodology

Considering that autobattler games occur in real-time and must be simulated to obtain data, the most fitting machine learning method for this problem is reinforcement learning. Specifically, the reinforcement learning platform we will be using is Unity ML-Agents, a game engine retrofitted to perform machine learning simulations. The reinforcement learning algorithm that we use is proximal policy optimization, which Unity uses by default. To train agents, we choose to use self-play, pitting units against each other as adversaries.

In our environment, we place units on a 7x7 chess-like board. One half of the board is reserved for the "red" team, the other for the "blue" team. We skip the preparation phase that is traditional in autobattler games, opting to place units in a fixed position, symmetric for both teams to ensure fairness. During the training episode, the units on both teams are allowed to take any actions. To end the episode, either one team must be eliminated or the timer must be ran out.

One team is selected as the "learning" team, which is used to update the model. At episode end, rewards are allocated for every unit on the learning team. The learning team status is swapped periodically as an overfitting measure. At the beginning of an episode, we set a 50% chance for the non-learning team's model to be swapped out for a older model, ensuring that learning team's model faces a variety of opponents during training as well as adding randomness into the environment.

In an autobattler game, we typically have a variety of units. Some are good at attacking, while others are good at tanking. For our simulations, we use three different types of units: damage, tank, and healer. We assign appropriate stats and abilities to these units in order to draw out interesting behaviors/strategies.

2.1 Actions, Observations, and Rewards

Every unit can take a number of actions, some exclusive to specific units (see Figure 2.1.1). Every unit observes information, some exclusive to specific units (see Figure 2.1.2). Every unit on the learning team receives an appropriate reward at the end of a training episode (see Figure 2.1.3). There are no explicit incentives for units to perform any sort of strategy, other than winning the round.

Action	Damage	Tank	Healer
Move up	✓	✓	✓
Move down	✓	✓	✓
Move left	✓	✓	✓
Move right	✓	✓	✓
Attack enemy target	✓	✓	✓
Change enemy target	✓	✓	✓
Heal friendly target			✓
Change friendly target			✓
Do nothing	✓	✓	✓

FIGURE 2.1.1: The actions units can take.

3 Emergent Behavior

We describe the strategies that units learn after training.

3.1 Stalling (video example)

After training a damage unit in a 1v1 situation, we see that trained unit tends to stop and wait for the enemy to approach rather than approach themselves. This allows the unit to get the first attack in, as units are only able to initiate an attack from a non-moving position.

Observation	Damage	Tank	Healer
Unit's position	✓	✓	✓
Unit's proportion of friendly team health	✓	✓	✓
Unit's proportion of friendly team DPS	✓	✓	✓
Direction to enemy target	✓	✓	✓
Enemy target within unit's attack range	✓	✓	✓
Enemy target unit type	✓	✓	✓
Enemy target's proportion of enemy team health	✓	✓	✓
Enemy target's proportion of enemy team DPS	✓	✓	✓
Direction to friendly target			✓
Friendly target within unit's heal range			✓
Friendly target unit type			✓
Friendly target's proportion of friendly team health			✓
Friendly target's proportion of friendly team DPS			✓
Friendly team healthiness - enemy team healthiness	✓	✓	✓
Friendly team DPS - enemy team DPS	✓	✓	✓

FIGURE 2.1.2: The information units observe.

Reward	Reason
+1	Win; enemy team eliminated
-1	Lose; friendly team eliminated
0	Draw; timer runs out

FIGURE 2.1.3: The rewards units receive.

3.2 Deliberate Target Focus (video example)

After training units in a 2v2 situation, with a damage and a tank unit on each team, we see that the trained damage unit tends to focus the enemy damage unit, rather than the enemy tank unit. Targetting the damage unit on the enemy side is important because it is a high priority unit compared to the tank unit (which has much higher health and lower attack). In the video linked, damage units are spheres, and tanks are cubes.

3.3 Prefer Healing Over Attacking (video example)

After training units in a 3v3 situation, with a damage, tank, and healer unit on each team, we see that the trained healer unit tends to prefer healing over attacking. Additionally, we see that the healer tends to focus healing the friendly damage unit, rather than the tank unit. The healer's attack is heavily reduced, so its healing output is higher than its damage output. Thus, it's reasonable for the healer to find more agency in healing over attacking. Prioritizing the health of the friendly damage unit is important since the damage unit does the most damage (even more than the tank and healer combined). In the video linked, damage units are spheres, tanks are cubes, and healers are cylinders.

4 Balance Changes

We notice that in the majority of simulated 3v3 rounds, the tank unit does not play a large role. Its only advantage is having a large health pool, but there is no requirement that the enemy team attack the tank, thus making it hard for the tank to affect the outcome of the round. Other than our human observation, this phenomenon is also suggested by the ELO numbers outputted by Unity, a metric used to describe the skill level of the models; during training, the damage and healer ELO rises, whereas the tank ELO struggles to break even (see Figure 4.0.1). So, in order to give the tank more agency, we add abilities into the game, i.e. special attacks that have cooldowns (cannot be used regularly).

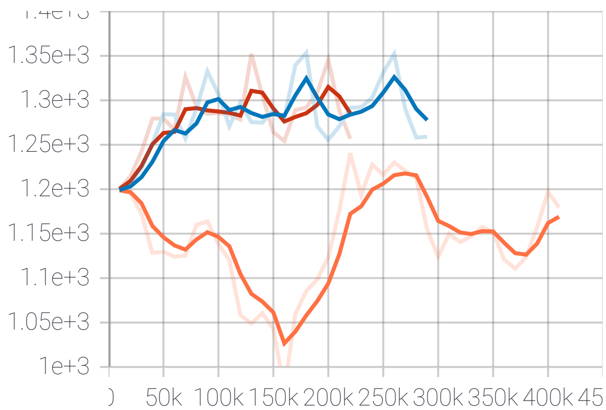

FIGURE 4.0.1: Damage and healer ELO (red and blue) rises, whereas tank ELO (orange) struggles.

We notice that the damage unit can single-handedly wipe out a team by being excessively aggressive. To combat this, we allow each unit the ability to stun an enemy, however, with a significant cooldown. If a unit is stunned, it cannot perform any actions. For damage units, their stun range is slightly larger than their attack range. For healer units, their stun range is exactly the damage unit's attack range. For tank units, their stun range is the shortest reasonable length (reaches only the adjacent positions), with their stun duration being twice as long as the other units'. The ideas behind these additions are the following:

- A damage unit should be able to approach an enemy damage unit that is stalling. Giving it a stun slightly larger than attack range enables it to do so.
- A tank unit should have the ability to stop an enemy damage unit from making overly aggressive plays. With a large health pool and long stun duration, a tank unit can now properly fend off an aggressive enemy damage unit.
- A healer unit should not be instantly eliminated if it finds itself in range of an enemy damage unit. With a stun range exactly that of the damage unit's attack range, it has the ability to stun the enemy and escape.

Overall, the abilities are meant to encourage healthy gameplay so that the tank units can train more effectively.

4.1 New Emergent Behavior in Tanks (video example)

After training units in a 3v3 situation (similar as before, but with stun abilities), we still see damage-unit-centric gameplay. Although, we do see dramatic changes in tank behavior. Before adding abilities, the tank AI never seemed to improve because of how little power it had. After adding abilities, we see the tank unit train more effectively, as shown by Figure 4.1.1 comparing ELO growth. An emergent behavior of the "new" tank is the tendency to position itself between the enemy and the friendly healer unit. We also see some instances of the tank unit playing aggressively to stun enemy units.

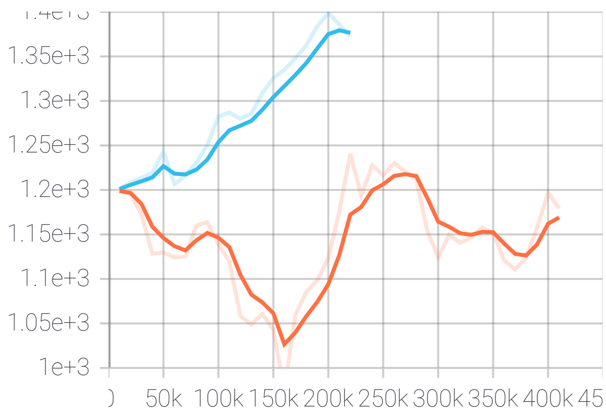

FIGURE 4.1.1: Tank ELO improves dramatically after balance changes (blue curve is after changes, orange is before).

5 Conclusion

We conclude that indeed reinforcement learning can be applied to the autobattler genre. Agents are able to learn relevant strategies based on their unique characteristics and abilities. However, for learning to be effective, there requires some amount of balance among the strength of each type of unit. At this time, it is questionable as to whether a machine learning-based AI would preferable over a simpler one. Nevertheless, these results show promise in the capabilities of reinforcement learning to create more complex and dynamic interactions among autobattler AI.