**Creation of a Gaming Agent for Russian Roulette Simulator**

**Group Members:** Davon Nelson, Samuel Heman-Ackah, Yusuf Khalid, Zekun Li

**Initial Goals**

The problem with the group selected was to create a gaming agent for a variation of Buckshot Roulette. Buckshot Roulette is a video game created by Mike Klubnika (Klubnika, 2023) that takes the real life game of Russian Roulette and adds a twist to the game. Normally Russian Roulette would involve two people, a revolver and one bullet in which each player would take turns shooting the revolver until the bullet is fired. With Buckshot Roulette it is the player versus the dealer(ai) in which the goal is for the dealer to kill the player. The player can choose to shoot the opponent or himself. If he shoots himself with the empty bullet, he gets an extra turn. The gun that is used in Buckshot Roulette is a shotgun instead of a revolver and there's a fixed amount of health for both sides. Both sides also have access to an array of items that can be used.

The difficulty of this problem then ranges in complexity based on the various different factors that are involved in the game. Each factor affects the model that needs to be developed to help train or design the gaming agent. The solution is important when it comes to game design. A lot of the times when creators are making games especially in the beginning they tend to use any AI models or software that comes with the development environment that they're using instead of creating their own especially for something like this that can range in difficulty depending on how simple or how hard you want to make the game. Therefore this kind of sets an example for others to follow if they want to take this path of developing their own game AI.

When the group first developed the idea of creating a game agent for buckshot roulette came up with two separate agent ideas. The first being the berserker which is an AI that acts a lot more aggressively when playing the game. The second one being the mimic that copies the players actions within the game. The initial goal for the berserker was for it to have a win rate of at least 50% when it is selected, as this is what the expected chance to win for human vs. human. We want this agent to behave more optimal when making decisions, and simulate the behavior of a human player. There was no surprise when training and testing the AI. Post testing the AI has a around 50% win rate which allows the game to be a lot more intense and complex.

We want to make this game simple and easy to access, so we decided to build the UI with a terminal user interface (TUI) and then port it to the web. The reason for choosing TUI is because it is a matured platform and widely available from all PCs with minimum requirements to hardware. We want to make our game accessible to all platforms, even those without graphic display, so TUI becomes our first choice. However, TUI is not suitable for distribution as it is a developer feature to most people, so we also want to port the game to a web hosting service. Web is another cross platform feature that is available in most devices. These two platforms combined together can cover most of the interested players of our game.

**Implementation**

Instead of the special items in the Buckshot Roulette, we decided to use powerup bullets as a replacement in our implementation. Powerup bullets are the replacement of regular bullets with special abilities. They are filled in the gun as it reloads, and it can apply the effect to the opponent or the person himself. We put two types of powerup bullets, one of them is healing bullets, which will heal the person who got shot. Another one is double damage bullets, which will cause double damage compared to regular bullets. There are two considerations for this change. Firstly, powerup bullets can be better integrated into our probability model. They are counted as a part of the bullets loaded, so their probability can be calculated as well as regular bullets. These probabilities will be helpful for the model to calculate the estimated reward for each special bullet and then make a decision on its next move. Secondly, players can choose to shoot the opponent or himself in their turn, so each powerup bullet can also damage the player or assist the opponent depending on the player's choice. The items in the original design have fixed use cases, and the beneficial items will definitely give you reward as long as you use it. Where in our design, players may heal their enemy with the healing bullet, or shoot themselves with double damage. This increases the strategy and complexity of the game, as the players need to be careful with their every shot.

We implemented the probability model for the agent in Python. The agent has two actions to take for their turn, either to shoot the opponent or himself. There are a total of 4 different types of bullets that can exist in the chamber of the gun. The number of live bullets is half of the meg size, and there is a maximum of 1 bullet for each powerup bullet. The rest of the chamber is filled with blank bullets. Both player and the agent know the number of each bullet in the chamber when the gun is reloaded, and assuming they won’t forget this information. They are also aware of the HP for each player. The agent will keep a record of the number of bullets, and predict the distribution of the bullets in the chamber. It will predict the probability for each type of bullet to exist in the next chamber, and then calculate the estimated reward with the predefined reward value for the th bullets.

Then we calculate the total estimated reward for each possible actions to get and . The agent will make a decision of his next action with the higher total estimated reward.

Since we don’t have training data for this task, the agent is working on probability calculation, which has two limitations. Firstly, the agent doesn’t consider the special case to gain extra turn. When there is an empty bullet in the chamber followed by a live bullet, the optimal solution would be to shoot himself with the empty bullet to get one extra turn, and then shoot the opponent with the live bullet. Our current implementation losses in this situation as the probabilities for empty and live bullets are all 50%. Secondly, our implementation doesn’t put the opponent's health in consideration. If the opponent’s health is below a certain level that can be killed by a shot, the reward for that deterministic shot should increase. We failed to resolve these limitations as it is hard to define a fixed reward value for these variables, and we don’t have enough data for training the model to learn the reward value.

**Evaluation**

Our goal for the agent is to be as similar to a human player as possible, and the optimal winning rate for a game-of-chance should be 50%. So we tested 10,000 simulated games with AI vs. AI to estimate the performance of our model. Out of 10,000 games, the AI who plays as the player wins 4985 of them, which is 49.85% chances to win. This number is close to our goal and expectation.

However, this evaluation statistic is less accurate as our goal of the game is human v.s. AI. Testing results on AI v.s. AI is less representative to the actual game setting. The game itself also largely depends on luck, where both agents don’t know the optimal solution. So as the number of games increases, the winning rate stabilizes at around 50%. For 100 test games, the winning rate will vary between 40% to 60%.

Overall we finished the initial goals of this project. We have a working simulator of a customized Russian Roulette game with a human-like AI, who has the winning chance of 50%. The lack of actuarial training and test data of human decision is the main cause of the limitation in both the implementation and the evaluation. If we have the series of decisions made by humans as a reference to train and test or data, the model will be more intelligent and be able to understand more of the game environment.

**Contribution for each member**

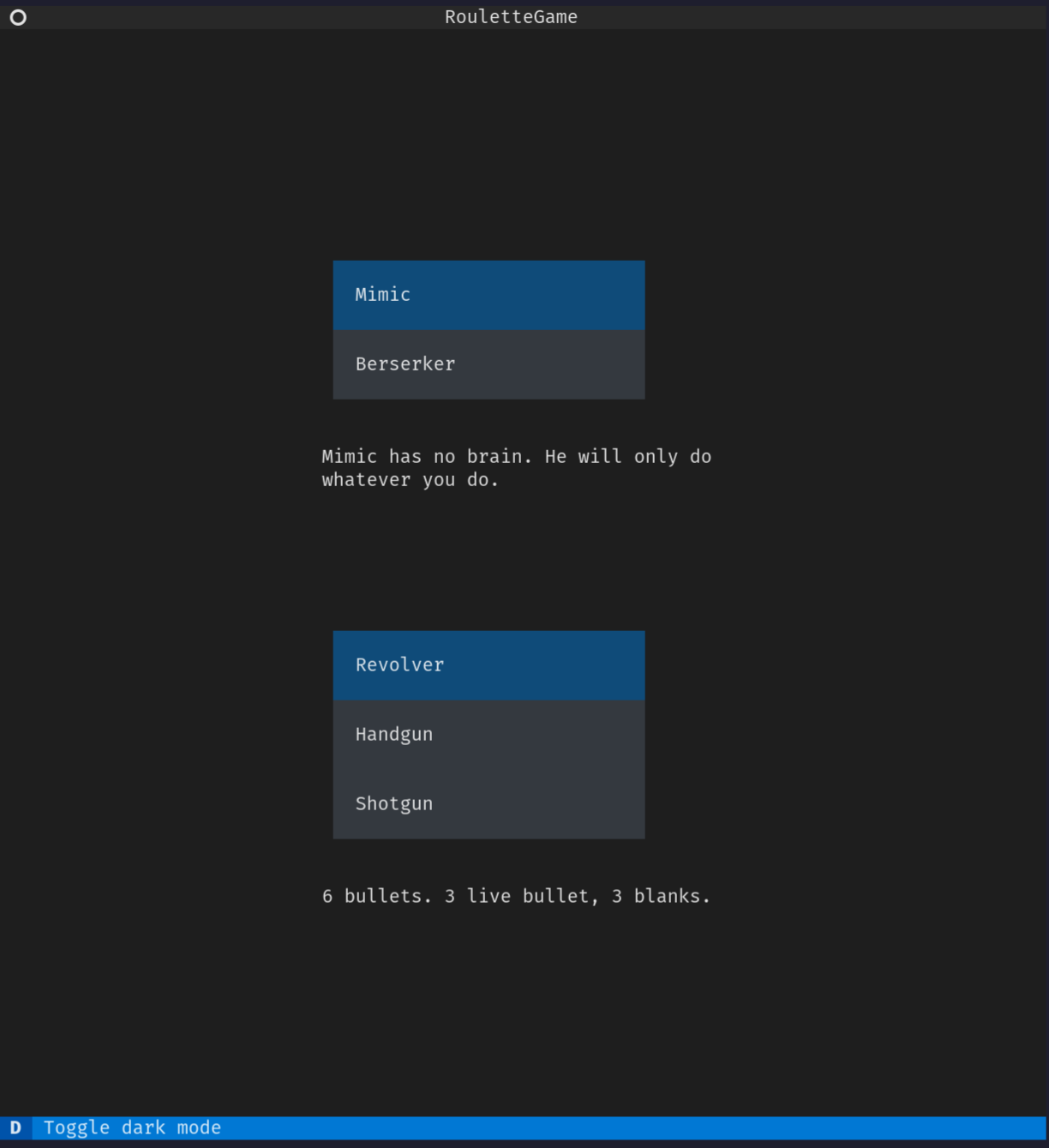
* Zekun Li: Created the Graphical User Interface(GUI) for the game, gave insight into how the probabilistic model should be designed, incorporated AI model and weapon mechanics into the GUI.
* Yusuf Khalid: Created Probabilistic model used for AI alongside side Samuel, gave insight into how weapons and power ups should work.
* Samuel Heman-Ackah: Created Probabilistic model used for AI alongside Yusuf.
* Davon Nelson: Created the mechanics behind the weapons and power ups used in the game, gave insight into how the probabilistic model should be designed.

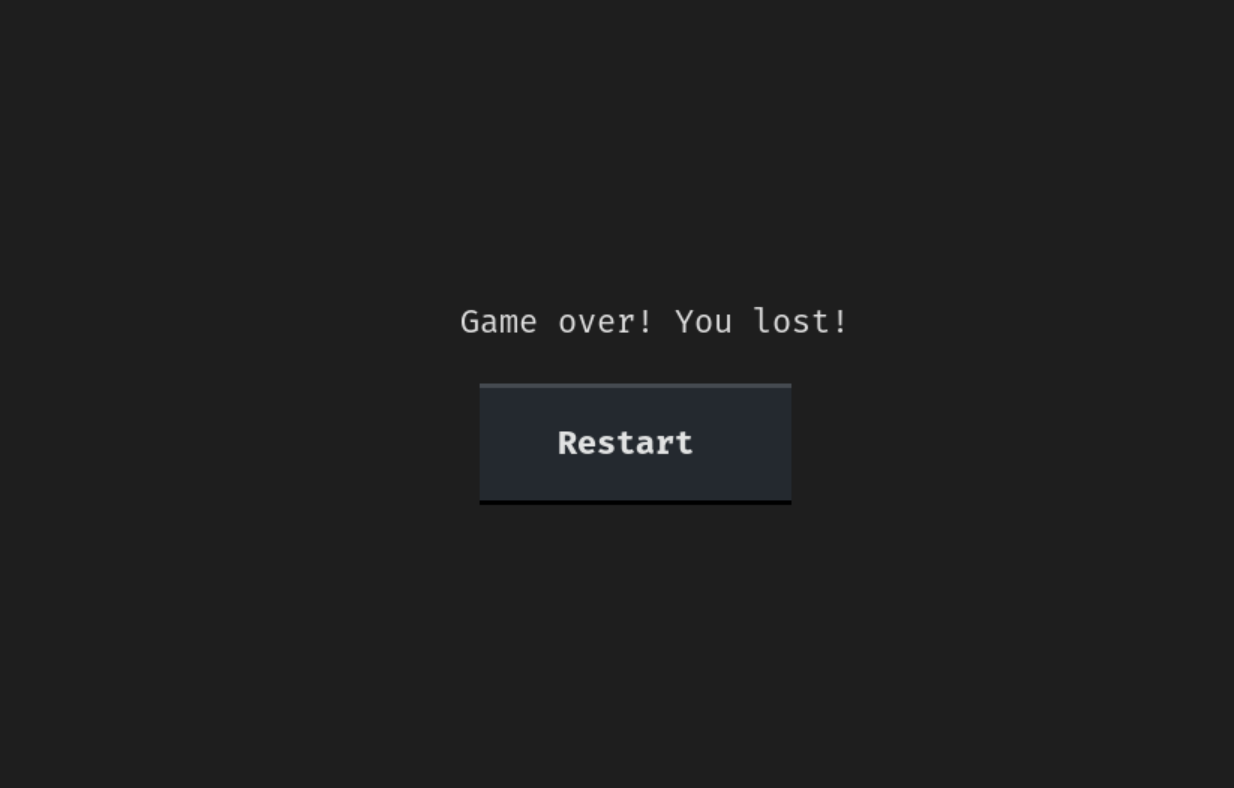
**Reference**

1. Klubnika, M. (2023, December 23). *Buckshot Roulette - Play Online Without Download!* Buckshot Roulette. <https://buckshotroulette.com/home>

**Appendix: GUI Screenshots**

Pic 1: Agent and gun selector (light mode)

Pic 2: Agent and gun selector (dark mode)

****

Pic 3: Game over screen with restart button

****

Pic 4: Game board with loaded revolver and berserker as opponent

****

Pic 5: Web server output