Introduction and Game Mechanics

StarCraft II is the sequel to Starcraft, a timeless real-time strategy game that takes place in a science-fiction universe ruled by three balanced races: Terran, Zerg, and Protoss (abbreviated as T, Z, and P). For the purpose of this project, we will limit our focus to strictly StarCraft II, specifically, competitive 1v1 games. The goal of each game is simple: efficiently allocate resources into building an army in order to defeat the opponent’s base.

At the start of each game, both players start out with one base and twelve workers, units that can mine resources and create buildings. There are two resources in the game: minerals and vespene gas. Once harvested, these resources can be used to build army units, buildings, and additional workers. A player can boost their resource collection rate in the long run by building more bases and workers during the early stages of the game. However, this forgoes the opportunity to build offensive units, which leaves a player vulnerable to attacks from the enemy. The challenge, then, for each player is to balance collecting resources and building a sufficiently-sized army.

The keyboard and the mouse is the portal between players’ minds and their respective units. Since the game is played in real time, speed and effective multitasking abilities are imperative. This is especially true late in the game, when players must control hundreds of units and dozens of buildings. Professional StarCraft II players average up to 300 to 400 actions per minute.

StarCraft II also includes a game mechanic known as “fog of war”, which limits the vision of each player to a small radius around each of their units and buildings. As a result, each player has limited information about their opponent’s strategy, including their allocation of resources, location of army, and buildings created. It is up to the player to use the vision of his or her units to deduce the opponent’s strategy. The fog of war mechanic is one factor that makes Starcraft II such a complicated game for artificial intelligence; it is a game of imperfect information.

Background and Motivation

Last year, DeepMind and Blizzard released the SC2 Learning Environment, an open source of tools used for AI research in StarCraft II. The release includes machine learning API, a python component of the environment (PySC2), and a large dataset of replays to analyze. The Starcraft II API provides full external control and includes functionality for machine-learning based bots and replay analysis. PySC2 provides an interface for reinforcement learning agents to interact with the game by fetching observations and sending actions.

StarCraft II presents many challenges as an AI environment because of its vast observation and action space. In the observation space, there are many structural and spatial elements to consider such as map terrain and the locations of resource clusters. However, SC2LE simplifies the game down into “feature layers,” which isolate elements of the game such as unit type, health and map visibility from one another, whilst preserving the core visual and spatial elements of the game. The action space is even larger, giving a player the choice to select actions among a combinatorial space of $10^8$ possibilities using Starcraft II’s point-and-click interface.
The large action space is a byproduct of the many different types of units and buildings, which each have their own unique set of actions. Since games typically last for thousands of frames, the consequences of a player’s actions may not be realized until much later in the game. The sheer magnitude of both the observation space and action space of Starcraft II has been a major obstacle for researchers interested in creating competitive Starcraft II agents. Through DeepMind’s AI, AlphaStar, it was only very recently (January 24, 2019) that computer scientists successfully constructed a competitive agent that beat professional players. Even AlphaStar, however, has its limitations. Since each race has its own unique set of features, AlphaStar was designed to only play PvP matchups. Nevertheless, the creation of AlphaStar was a major breakthrough in the computational intelligence for games and we anticipate that AlphaStar will eventually be able to competitively play Starcraft II in the other race matchups. Unfortunately, the time, extensive processing power, and resources the AlphaStar team had access to are not available to our team. Thus, our project aims to simplify the problem and attain a more modest goal: develop a program that predicts the win probability of a given Starcraft II player during a competitive 1v1 Starcraft II match.

The eSports industry is growing at an explosive rate, and with competitive professional matches comes pre- and live game betting. Being able to predict the outcomes of matches between players has become increasingly more important, and our program may have important applications in quantifying skill levels among eSports participants. Additionally, by training an AI agent to predict win probability rather than to competitively play the game, we eliminate some of the problems that negatively affect the game-playing bots. In terms of the observation space, the AI agent would have accessibility to all observable features of the game and would no longer be hindered by the “fog of war” game mechanic. Furthermore, our agent would no longer have to make decisions on actions to take, as it would not be playing the game, only observing it.

**Project Details**

The project will consist of three main parts. Part I will consist of constructing the basic framework to analyze the replays of Starcraft II games and to access in-game data. During Part II, I will initially develop a simplified neural network that disregards race matchups to roughly estimate win probability. This neural network will ignore the nuances specific to each race and will calculate probability on more generalized inputs such as army supply, number of upgrades researched, resources mined, and players’ matchmaking ratings. To conclude Part II, I plan on enhancing this basic neural network to focus specifically on PvP matches to incorporate game mechanics specific to the Protoss race. These features include but are not limited to army shields, health regeneration, and warp gate. The neural networks in Part II will be trained using the 1v1 replay packs provided by SC2LE. Multiple frames from every replay will be taken as individual data points. The network will then be used to predict the outcome (a player’s win probability) of each of these frames. Part III will consist of analyzing the accuracy of my PvP neural network and improving its accuracy by changing the input vectors. In order to assess the accuracy of my model’s win probability, I intend to assign scores to each data point prediction, rewarding the neural network if it predicts a winning player to have higher than a 50% win probability in the current game frame, and penalizing it if it predicts otherwise.

**Division of Work**

While I work on designing the basic framework in Part I, my partner Andrew will work on designing a simple neural network for microRTS, a simplified implementation of RTS games.
like *Starcraft II*. Building a neural net using *microRTS* will allow our group to get familiar with preprocessing complex data inputs like unit positioning or unit composition that we may try to replicate in our *StarCraft II* model. In Part II, I will use Andrew’s *microRTS* neural network as a foundation to guide the structure of my race-independent neural network for *Starcraft II*. Since *microRTS* oversimplifies many aspects of *Starcraft II*, this segment will require me to account for additional complexities such as upgrades and number of bases. After I have completed this simple neural network, I will proceed to focus on Protoss vs. Protoss matchups, while Andrew will begin constructing a neural network specific to Terran vs. Terran matchups. Because each race in *Starcraft II* has idiosyncratic mechanics, units, and abilities, we anticipate our neural networks to concentrate on different, original inputs. We anticipate testing and refining various inputs to each of our models, creating multiple iterations of our *Starcraft II* neural network along the way. While we intend on working independently on our individual neural networks, we also expect to communicate with one another regarding methods of preprocessing data and the types of inputs that were successful at achieving more accurate predictions. Our project will culminate in separate analyses of our own neural networks, detailing our race-specific inputs and the performance of our models. If time permits, we hope to extend our research project by predicting the win probability of the other race matchups.

**List of Deliverables**

- Source code of simplified neural network that disregards race matchup
- Source code of the PvP neural network agent
- Final paper that details the methodology, the various neural networks tested, and their respective results
- Oral presentation to present my findings

**Timeline**

- **Week 1**: Refine and finalize research proposal with Professor Glenn.
- **Week 2-3**: I will get familiarized with PySC2 and build the framework necessary to fetch relevant data from the replays of *Starcraft II* games. Simultaneously, Andrew will work on *microRTS* to develop a basic neural network that implements basic win prediction.
- **Week 4-6**: I will use Andrew’s *microRTS* neural network to develop a race-independent neural network for *Starcraft II*. Simultaneously, Andrew will further develop his *microRTS* neural network to handle more complex input that may be applicable to the more complicated *Starcraft II* neural networks we will eventually design.
- **Week 7-11**: Develop a *StarCraft II* neural network specifically for the PvP matchup. Use both Andrew’s *microRTS* neural network and my race-independent neural network as a guide to refine a list of inputs to the PvP neural network.
- **Week 12-14**: Analyze the accuracy of the model and observe how certain inputs affect its accuracy.
- **Week 15-17**: Synthesize all of the data and write a final research paper with our results.