Machine Learning - Final Project Proposal
Starcraft II Win Predictor

Derek LaFever*
*George Washington University, dtlafever@gwu.edu

***Abstract*—Realtime strategy games are are the next step in Game AI research due to the sheer complexity. I used the data from professional players in Starcraft II to develop a winner predictor that takes as input partial map image data from one player's perspective. I achieved a training accuracy of 64% using oly the image data. Such predictors can be made available to human players as a supportive AI component, but can more importantly be used to better inform strategic planning for an AI controlled player.**

***Keywords*—Machine Learning; Starcraft ; Computer Vision**

I. INTRODUCTION

Starcraft II is a digital real-time strategy game. Starcraft II is typically played as a 1v1 format where each player builds a base on a map and attempt to destroy the opposing players base. There are 3 races to choose from in which to build your base and units from, leading to diverse strategies and interactions. It is a complex game due to its numerous strategies, partial information of the whole map, and all events being preformed in real-time. So, unlike chess, Starcraft is continuous and therefore all actions are decided asynchronously between players in real-time. Because of this game design, multiple units can preform actions in the same moment in time, adding another layer of complexity. More so, the player must preform all of their actions with incomplete knowledge/vision of their opponent due to a fog-of-war system. Fig1 shows what a typical game of Starcraft II looks like from a players perspective. Take note of how the viewing window also does not show all of the map at once, requiring another element of controlled focus over portions of the map, while a simplfied view of the map is shown in the bottom right.

Fig. 1: Screenshot of a game of Starcraft II.

Starcraft II is a growing game of interest for researchers due to its complexity that goes beyond traditional tabletop games such as Chess or Go. There has recently been APIs [1] opened up for researchers to better develop AIs that can compete against human agents [6]. However, no current research is able to win against experts in the game, let alone the 50-30% of players [5]. This is in part to due to the various systems that have to been managed asynchronously in the game, including resource management, army management, and building placement.

Fig. 2: LEFT: Screenshot of the map for player 1 RIGHT: Screenshot of the map for player 2.

A. Links

Brief overview of the game of Starcraft II: **https://www.youtube.com/watch?v=yu1Ze3ucsfo**
Website on the tools to access Starcraft II API: **https://deepmind.com/blog/deepmind-and-blizzard-open-starcraft-ii-ai-research-environment**

II. PROBLEM STATEMENT

I proposed to create a CNN to predict which player will win a game of Starcraft II given the map image from a particular players point of view. Fig 2 shows a examples of the map images from both players perspectives at the same moment in time for a particular game. Notice on each players map how they cannot see the most updated information of their enemy, only what their units can see around them and previous information that was obtained. The rationale behind developing this predictor is to better drive larger machine learning algorithms that serve as artificial players to compete with humans. With my predictor, the goal is to aid an AI controlled player in deciding macro strategies based on the my predictor. This can lead to quicker surrenders if the AI is losing badly and faster victories otherwise. It may also lead to more dynamic strategies that adjust to the game state, allowing for better play.

III. Related Work

Predicting who will win in a game of Starcraft is not new. Antonio Sanchez-Ruiz first explored this concept in 2015 for the game of Starcraft I [3]. He, however, used only AI controlled players that the game developers made for his dataset of games. While good in theory as the players are consistent in play, it is not very representative of a real player versus player games because the AI performs set logic for each game, leading to little variation in play. Furthermore, Antonio's method was tested on full knowledge of the both players data, not just from one player's perspective. Sanchez-Ruiz & Miranda expanded their dataset to influence maps, a spatial and tactical representation of the game state, and trained again on AI controlled players and achieved better results [4]. Others have created predictors that can determine the outcome in the first 10 minutes [2] or time slices throughout a particular game to update a prediction based on partial information [7], but non have used map image data in their predictions. My method would be adding more data that is available to the player that is not represented in these other papers. Such data could lead to better predictions.

IV. Dataset

The dataset is constructed by taking map images from a recent competitive tournament consisting of 410 matches. This tournament represents some of the best players in the world, so it has less tainted data of matches where players leave abruptly due to connection failures or do nothing all match. Moreover, by training on professional players, this data is more indicative on what an AI would ultimately be competing against.

During each match, images are collected every 5 seconds along with the player whom ends up winning, what race (the civilization the player is controlling) each player is, the current time, and total time. Once collected, the data is preprocessed to be normalized from 0 to 1 for all features. Due to players variance, it is worth noting that the distribution of wins across all races is not even. This is due to player preference over certain races along with players who advance further with a particular race are representing that race more so then others. For example, the player who won the tournament was the race Zerg and therefore had to win all of his matches. This lead to a higher number of Zerg wins than other races that where bested in the tournament.

Since every one of these games is from the past, we have ground truth data to work with and this is therefore a supervised learning problem. With this ground truth data, I created a train-test split (80/20) on the data to evaluate the performance of my CNN.

A. Subsection

V. Approach

Starcraft II is a relatively new research topic of interest due to better machine learning be able to achieve unprecedented skill in strategy games, including Deep Minds AlphaGo's achievements [8]. Even more so, the novelty of analyzing not just game data, but also map data, leads towards stronger success for AI's to compete with players.

My CNN outputs a binary classification on which player is expected to win based on the map data and game state. I hope that with the CNN, I can gain more insight as to the key parts of the map that matter to the success or failure of a particular player. This could not only help in developing future AI's, but also for players to better understand key points of the map to observe and their play.

VI. Experiments

My work consisted of two main experiments: using the whole dataset and only using the data from the last 30 seconds of each game.

I ran ResNet-18 initialized with the weights pretrained on ImageNet for these two experiments. I tested ResNet-18 with uninitialized weights, but did not preform better than the pretrained ones. To best find the right parameters for the network, I took a small sample of the data (100 images) and trained on those images until I could get close to 1.0 training accuracy. The loss function chosen was binary cross entropy due to its reliable performance.

The data augmentation methods applied were slight resizing of the image and normalizing the image by subtracting the mean image from the source image and then dividing by the standard deviation. The slight resizing of the image is due to the fact that all maps are not the same size, so by resizing I can better generalize for various size maps.

VII. Results

The first experiment that ran over the whole dataset achieved a training accuracy of 0.6457 over 50 epochs. The second experiment consisting of the last 30 seconds achieved a training accuracy of 0.5718. Both the results are shown in Fig 3.

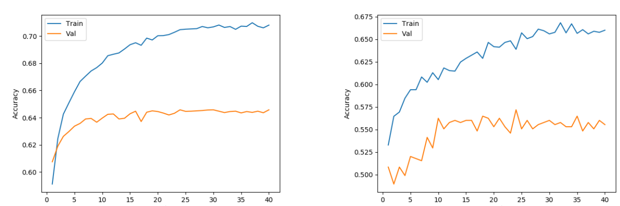

Fig. 3: LEFT: Accuracy over Epoch for the full dataset RIGHT: Accuracy over Epoch for the Last 30 seconds dataset.

While these results are taken from images where only one player's perspective is shown on the map, I wanted to compare this to results where all information of the map is present. Due to error in data collection, only 25 games were able to processed. Fig 4 shows the results from this work for the two experiments on this shortened dataset. The highest training accuracy was 0.9298 and 0.8400 for experiment one and two, respectively.

VIII. Discussion

It is interesting to note in both the partial vision and full vision dataset, using all of the data preforms better than using

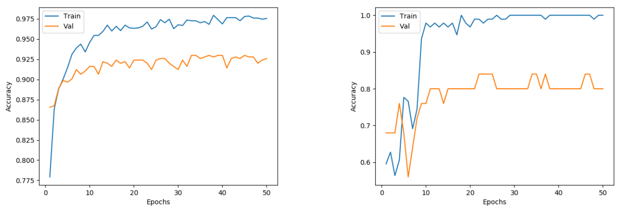

Fig. 4: LEFT: Accuracy over Epoch for the full dataset RIGHT: Accuracy over Epoch for the Last 30 seconds dataset.

only the last 30 seconds of each game. My best guess for this is due to many professional players leaving the game early once they believe they are in a losing position. Much like a master chess player, they can predict the outcome far in advance from their current position. This is problematic as the map can still show both players having most of their buildings and units alive even though the position is lost. Players can also make better inferences because they have more data to work with. They have information such as what army composition (types of units) they have versus their opponent, how much resources they have left, etc. This is further evidenced by previous work preforming better than my results: they simply have more data over the entire game to work with.

In regards to the second dataset of full vision map images, I cannot directly compare these results to my partial vision dataset due to the great imbalance of the two sizes. The dataset will need to be properly fixed and recreated in order to properly evaluate the potential performance gains of being able to see the whole map.

IX. Conclusion

In the end, I successfully developed a CNN to classify who would win in a game of Starcraft II better than chance (50%), but I believe that I can preform better results by combining my visual map data with previous work's streaming data of things such as resources at a given moment, army composition, and general performance over the duration of a game.

References

[1] "Starcraft ii learning environment," https://github.com/deepmind/pysc2, accessed: 2018-11-19.

[2] A. Álvarez-Caballero, J. J. M. Guervós, P. García-Sánchez, and A. Fernández-Ares, "Early prediction of the winner in starcraft matches." in *IJCCI*, 2017, pp. 401–406.

[3] A. A. Sánchez-Ruiz, "Predicting the winner in two player starcraft games." in *CoSECivi*, 2015, pp. 24–35.

[4] A. A. Sánchez-Ruiz and M. Miranda, "A machine learning approach to predict the winner in starcraft based on influence maps," *Entertainment Computing*, vol. 19, pp. 29–41, 2017.

[5] P. Sun, X. Sun, L. Han, J. Xiong, Q. Wang, B. Li, Y. Zheng, J. Liu, Y. Liu, H. Liu *et al.*, "Tstarbots: Defeating the cheating level builtin ai in starcraft ii in the full game," *arXiv preprint arXiv:1809.07193*, 2018.

[6] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser *et al.*, "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782*, 2017.

[7] V. Volz, M. Preuss, and M. K. Bonde, "Towards embodied starcraft ii winner prediction."

[8] F.-Y. Wang, J. J. Zhang, X. Zheng, X. Wang, Y. Yuan, X. Dai, J. Zhang, and L. Yang, "Where does alphago go: From church-turing thesis to alphago thesis and beyond," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 2, pp. 113–120, 2016.