Dota 2 Game Prediction

Sean McGuire
Electrical and Computer Engineering
Rowan University
Glassboro, New Jersey
Email: mcguires5@students.rowan.edu

Jacob Epifano
Electrical and Computer Engineering
Rowan University
Glassboro, New Jersey
Email: epifanoj0@students.rowan.edu

***Abstract*—In this paper we explore the possibility of estimating the outcome of any Defender of the Ancients 2 (Dota2) game by feeding in massive amounts of match data into an MLP. Our dataset consisted of over four hundred thousand match made games and their outcomes. The features we used in testing were: heroes played (10 different features), and the duration of the game. Our training and testing split was 70% and 30% respectively. Using three different network sizes, (3, 5 and 7 Layer architectures) while using only the hero features, we were able to predict match outcome with a peak accuracy of 86%.**

I. Introduction

Defense of the Ancients 2 also known as Dota 2 is a popular online video game, created by parent company Valve, where two teams of 5 players face off to destroy the enemy's base, called the ancient. Each of the 10 players must select a hero from a pool of 115. This means that there is roughly 1.8784179e+16 combinations (115 choose 5 * 110 choose 5) for heroes in a game. A neural network was designed to predict the winning team given only the heroes picked and the game duration. A typical game of Dota can last anywhere from 20 minutes up to 2 hours, given this length of time it is assumed that it would be easier to predict the winner as different heroes perform better or worse on average at different times of the game. Dota 2 is a very complex game and has become a spot light for the applications of deep learning. One application was revealed during the summer of 2017 at the biggest Dota 2 tournament, The International, which had a prize pool of $24,787,916. Here Elon Musk's company, Open AI, revealed a 1v1 Dota 2 bot which was trained with reinforcement learning and was able to win every game against any pro player it was matched against. This bot was able to beat any player in a 1v1 match up, but the game becomes infinitely more complex when the game is 5v5. Open AI announced that they planned to train a team of 5 bots that would be able to face off against the best team in the world in the summer of 2018. This would be an amazing feat, as in a game like Dota 2 team work and communication are vital to success, there is also an infinite number of game states as no two games are alike. [1] In 2018 Valve, the maker of Dota 2 announced a paid service called "Dota Plus". This paid service uses machine learning and deep learning to look at the current game state and recommend items and skills which should be leveled or purchased based on the current game state. [2] Many individual developers have collected data and created their own team picking networks which take in all of the picks and bans of both teams to try to create the optimal team. These models are useful as they recommend the best heroes to pick in the current state of the draft to maximize your chance of winning the game.

II. Methods

A. Proposed Solution

As stated in the introduction, predicting the outcome of Dota2 games is a massive problem. To be able to solve this using deep learning techniques would take a large amount of data and computational resources. Recently Valve has released an API called Dota2API that is integrated with python to be used as a means to make the game data from every game ever played available to the public. We propose that we could use associated match data to build a very large dataset. Using this very large dataset we can use a deep Multi Layer Perceptron (MLP) to be able to predict the outcome of Dota2 games with a minimal amount of features.

B. Data Collection

The data for this project was downloaded from the Dota2API which was accessed in python by importing the Dota2api library. Using the library matches are pulled by their unique match number. A function exists in the API which pulls the next 100 games after an index it is handed. These 100 games are from different game types and certain modes include bots. To make sure that all of our data come from the same distribution only games that are "All Pick" meaning you select from the full hero pool and games where all 10 human players are present are used, the rest are not stored. This function is called inside of a loop which allows us to build a dataset of over four hundred thousand matches. The API has a forced timeout on the user if too many request are made so a time delay is added and a try catch is inserted in case the requests ever time out. The data matrix and current game count are saved to the hard drive after every iteration of the loop. This means that if the program is stopped by an error, or by the user the data pull can continue from where it left off.

C. Parameters Selected

The parameters collected in the data matrix are: win or lose; the heroes picked by each person; and Game length. The first column referring to win or lose is a 1 if the first team won the game and a zero if the first team lost the game. The heroes are associated with each player, players 1-5 are on the first team and 6-10 are on the second. Each hero in the pool is number 1-115 and this number is stored in the corresponding column of the data matrix based on the player who picked that hero. The last column simply displays the game duration in seconds, which is the length of time elapsed in the game. Two different experiments were run. We ran an experiment with only the Hero features, and a separate experiment for Hero and Duration features. Three network sizes were used to test each experiment and were trained for 2000 epochs.

D. *Network Design*

For our experiments we had three network sizes all following a similar build: 3, 5 and 7 layer architectures. The three layer network was constructed of fully connected layers consisting of: 1000, 500, and 1 nodes. The five layer network was constructed of fully connected layers consisting of: 1000, 500, 1000, 300 and 1 nodes. The seven layer network was constructed of fully connected layers consisting of: 1000, 500, 1000, 300, 500, 100 and 1 nodes. All three network designs were kept uniform across all experiments. We used a simple MLP feed forward neural network with sigmoid as the nonlinearity applied at each layer. Batch normalization and a dropout of 0.1 were applied after each hidden layer to help with regularization of the network. The optimizing function used for each network was ADAM. The loss function used is binary cross entropy as this is a 2 class problem. Because of slight imbalances in the dataset, the class weights were also weighted using a Keras function call, this effectively weights the loss function. The model was trained on 70% of our dataset and validated on 30%. The model weights and tensorboard graphs were saved at the end of each training session. If desired the model can be loaded and these results can be reproduced.

III. Results

A. *Heroes*

The results in this experiment were obtained by training a neural network using only the ten hero based features of our dataset. Each hero from the pool of 115 has an associated integer, 1-115.

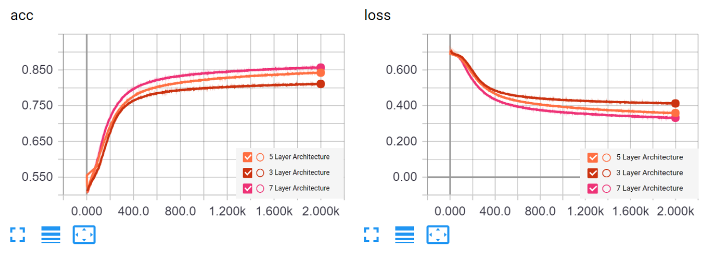

Fig. 1. Training Tensorboard Graph for all Network Sizes

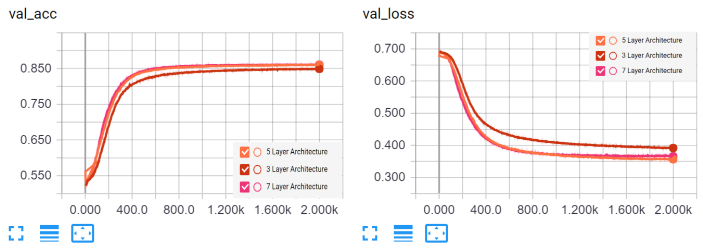

Fig. 2. Validation Tensorboard Graph for all Network Sizes

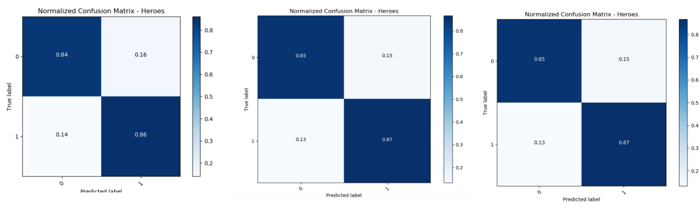

Fig. 3. Heroes Normalized Confusion Matrices for each network size. Left: 3 Layer, Middle: 5 Layer, Right: 7 Layer.

B. *HeroesAndDuration*

The results in this experiment were obtained by training a neural network using both the ten hero based features and the duration of the game in seconds.

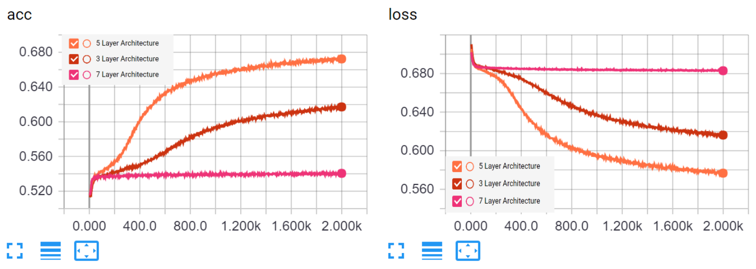

Fig. 4. Training Tensorboard Graph for all Network Sizes

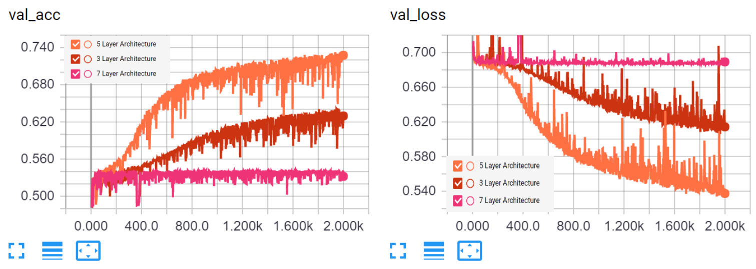

Fig. 5. Validation Tensorboard Graph for all Network Sizes

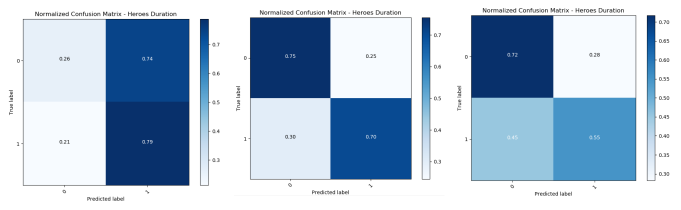

Fig. 6. HeroesAndDuration Normalized Confusion Matrices for each network size. Left: 3 Layer, Middle: 5 Layer, Right: 7 Layer.

C. *Results Comparison*

Figures 7 and 8 show the tensorboard graphs for the training and validation of the two sets of features. Each set contained 3 experiments of the different network sizes.

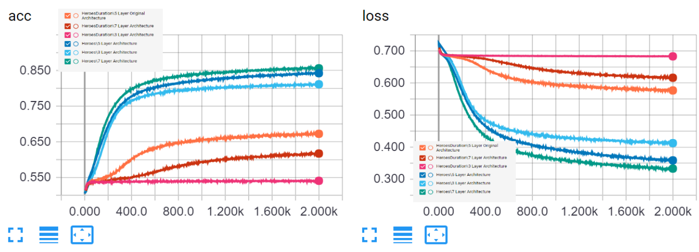

Fig. 7. Training Tensorboard Graph for all Network Sizes

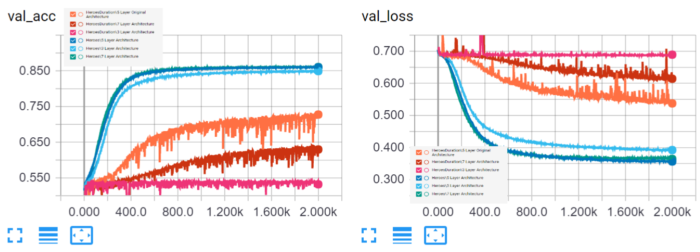

Fig. 8. Validation Tensorboard Graph for all Network Sizes

IV. Discussion

A. *Heroes*

Our best performing results were found to occur when using the Heroes only dataset. As stated above this dataset consisted over 400000 matches with 10 features representing the 10 heroes played in each game. Each of our network sizes performed about the same as they varied by only a few percent from the highest to lowest accuracy. All of our network sizes achieved mid 80% classification accuracies on our validation dataset.

B. *HeroesAndDuration*

The larger of the two datasets consisted of the 10 hero features, and an additional feature, which was the duration of the game. The results for this experiment were more varied than the last one. Results spanned from low 50% to high 70% classification accuracies depending on the network size. The 5 Layer network vastly outperformed all others and achieved an accuracy of 73%.

C. *Results Comparison*

It was originally hypothesized that as more information in the form of features was added that the model will be able to predict with a higher accuracy, but it was found that this was not the case. From these experiments it was found that the best results occurred in both experiments with a 5 layer neural network. It was also found that overall, the model was able to predict the winner with the highest accuracy when just using the heroes, adding in the duration of the game actually decreased classification accuracy. After seeing the performance of both sets of features it was found that the duration of a game did not add any useful information to the classifier. This contradicts the hypothesis formed before starting the experiments, but these results do make sense. As a match goes longer or shorter the chance that one team can take the lead and make a mistake, therefor handing the advantage to the other team increases. This added level of complexity and inconsistence does not add any useful information for our model to learn. To achieve the highest accuracy in predicting the outcome of a Dota 2 game it is recommended that you use the 5 layer network trained using just the Heroes.

D. *Challenges*

When pulling the original data it was found that after dozens of hours of runtime if the user decided to stop the program and it was in the middle of writing a row to the matrix it would save a numpy object that was not rectangular and therefor could not be opened. Another challenge faced was computation time each network took between 2 and 6 hours to run depending on the GPU used, this means that the full set of experiments took roughly 36 hours, this was time spent on top of the over 100 hours spent pulling data. Overall this project was very time intensive on the machine which would run it.

V. Conclusion

From the experiments run it was found that the best results were achieved from using just the 10 heroes picked. This means that by using the model the winner of a game of Dota 2 can be predicted with up to 86% accuracy with data available before the game even starts. The result of this experiment is consistent with popular opinion and has the potential to be deployed in a similar manner to the ”Dota Plus” service discussed in the introduction. The dataset used in our experiments, however were pulled at random from the API. There exists a large spectrum of player skill from casual players, up to the pros. It would be interesting to see if the accuracy of predictions could be improved if each skill bracket is separated, but this would require much more data. We do not yet know the effectiveness of our algorithm at higher skill levels and would like to investigate this further in the future.

VI. Code

https://github.com/mcguires5/Dota2Predictor
Contains:

- Tensorboard data files
- Trained Models for each experiment and network
- Code for each network
- Images of all Tensorboard graphs
- Completed dataset of 400000 matches
- Counter for that dataset (The dataset can be expanded by simply running ”DataCollectorFinal.py”)

References

[1] OpenAI at The International, OpenAI. [Online]. Available: https://openai.com/the-international/. [Accessed: 27-Apr-2018].
[2] ”Dota Plus”, Dota2.com, 2018. [Online]. Available: https://www.dota2.com/plus. [Accessed: 06- May- 2018].