Research Proposal: Predicting eSports Wins Utilizing Real-Time Data

Daniel Craig

CUNY - DATA698

Dr. Arthur O’Connor

3/25/24

# Introduction

ESports is a rapidly growing industry, currently evaluated at $4.3 billion dollars in 2024, with a 7% growth rate over the next 4 years and expected value of $5.8 billion dollars by 2028 [1]. In the realm of competitive gaming, Dota 2 is one of the most lucrative titles a professional eSports player can compete in. Viewership for Dota 2’s global tournament, The International, garnered nearly 1.5 million viewers in 2023 [2]. The International holds 7 of the top 10 highest prize pools in eSports history with the highest record of $40 million dollars and Dota 2 as a game holds 8 of the top 10 spots [3]. The game holds roughly a 200,000 player count at any point in the day with peaks of up to 600,000 [4]. The game is in a 5v5 team format in a top-down perspective, such as real-time strategy games StarCraft or Command & Conquer, with players selecting from a pool of 100+ heroes with unique abilities with the goal of destroying the enemy team’s “Ancient”, taking players 35 to 45 minutes to complete the game. The “Ancient” is the furthest building from the enemy team’s spawn location and serves as an equivalent to a King in chess. The players compete for two primary resources; gold and experience. These resources can be gained by being near or killing enemy players or units. Players are placed on a square map with three distinct areas called “lanes.” There are two lanes that trace the outside of the square and one lane that spans the bottom-left to top right corner diagonal of the square. Non-controllable units called “creeps” run down the three lanes where they clash against enemy creeps. Players compete for killing more creeps than their enemies to gain gold and experience until they are strong enough to destroy the enemy’s Ancient. Due to the limited areas a player can earn gold, teams give priority to specific players and are typically denoted by position. With 5 players, the positions range from 1 to 5, with 1 being the most dependent on gold and 5 being the least. This will show to be an important factor in modeling. There are several defensive buildings called towers that are placed in each lane that serve as deterrents to keep enemies from destroying a team’s Ancient. Teams contend with each other using the unique abilities of their hero, typically with millions of dollars on the line.

As a heavily viewed eSport, statistics, predictions, and match favorability are presented during live feeds of professional matches with an entire production crew serving live overlays and stat blurbs during gameplay. One feature that was embraced and developed by the game’s developer, Valve, is a live feed of win favorability, in the form of percentages, throughout the game. Yu et al. (2018) [5] found the feature to be 68% accurate at the half-way point in games. This favorability can drastically change throughout the course of the game swinging from 80% favorability to as low as 30% at times. While a formal explanation of Valve’s methodology for predicting a win during a live game has not been made public, there have been several forays into predicting wins.

This research seeks to add to this existing knowledge focused on enhancing viewer experience with meaningful win prediction probabilities by training a model to give win probabilities every minute using historical data accessed via API. Deviations from existing knowledge and this paper are the significantly reduced workload related to data collection, the feature selection focusing on capturing unique hero to hero interactions, and a utilization of neural net architecture to improve predictions. The model design is intended to be a Long Short-Term Memory (LSTM) model, which is a specialized recurrent neural network, forecasting over hero gold acquisition and team fight outcomes, with a logistic regression layer to create win probabilities per minute mark. This change in approach from other papers, that use a Game State Integration application during each game, avoids intensive time needed to collect data for model training and a change in features guided by domain expertise to reduce feature management. The body of knowledge thus far has utilized in-game variables individual to each player to predict gold but have not captured the effect of one hero’s performance impacting another hero’s performance in the acquisition of gold in variables or models. The hypothesis of this paper is that forecasting gold for a single hero using other heroes’ gold creates an accurate forecast to predict a final winner of the match due to capturing latent hero interactions in the time-series data.

# Literature Review

Song et al. 2015 [6] predicted wins based on only the heroes selected during the drafting stage of a match before any play across 3000 matches. A logistic regression approach was utilized with features chosen as heroes drafted and hero combinations for a testing accuracy of around 60%. Akhmedov and Phan [7] created a game – state integration app to pull data during a game live for roughly 100 games, performed a sensitivity analysis around the variable *player.gold* to determine highly correlated features to create a new dataset of variables for use in a multi-step forward prediction and fed into Linear Regression, Neural Net, and Long Short-Term Memory (LTSM) models for impressive 82%, 88%, and 93% accuracies, respectively. While this method was effective, the paper did not explore the impact of this on predicting on which team won. Yang et al. 2016 [8] combines pre-game features in player ranking and player hero statistics with in-game variables as features in a multi-model architecture to create a final prediction. Yang et al. 2016 [8] used pre-game features in a logistic regression model which results in a 70% accuracy in prediction. Gold, death, and experience variables, gathered from live games, were used in an Attribute Sequence Model (ASM) to predict the transition probability of the variable *player.gold* to quantify likelihood of gold change. The outputs from the pre-game logistic regression model and the ASM transition probability metric are combined into a final logistic regression model to create a prediction in which team would win. A comparison of prediction accuracies between pre-game, real-time, and combined models are compared over game duration with the combined models sitting between 75-80% within the first 20 minutes of a game and 85-95% after the 25 minute mark. A common theme across papers using a time-series of *player.gold* is apparent.

As some studies progressed into deep learning utilizing recurrent or convolutional neural networks, others returned to classic machine learning algorithms such as V.J. Hodge et al. (2021) [9]. V.J. Hodge et al. (2021) [9] focused on a Weka Logistic Regression, Weka Random Forest, and Microsoft’s LightGBM algorithms for prediction and parsing features using a replay parser. Specific features were not detailed outside of being labeled as ‘in-game metrics’. These models were tested during a live event, ESL One Hamburg 2017, in coordination with the event handler to generate live stats and enhance commentary and analysis by the talent panel. Game stats were retrieved using another Game State Integration app. Accuracy for predictions in professional games were placed at 75%. This is a stark difference from Yang et al.’s (2016) [8] approach of utilizing pre-game features in a layered architecture approach.

# Hypothesis

Proposal for this research would attempt to bridge a specific gap related to feature selection. This research seeks to utilize individual *player\_gold* earned as ten different time series’ to forecast on each. After forecasting, the heroes and their time series forecasts would be used as features for predicting a win. In the aforementioned documents, the player gold variable was simplified into an overall “gold differential” that represented how much more gold one team had accumulated over another. High-level players and commentators often mention that heroes are designed to impact a game more with varying amounts of gold. By simplifying *player\_gold* into team differentials, this paper posits that a loss of relationship in the data occurs, and its inclusion will improve predictive ability or require less data or features to achieve similar results found in other papers. While most research focuses on a gold differential between teams to evaluate as a feature, this research focuses on the relationship between the unique combinations of which heroes have how much gold at which point in the game.

Ultimately, predicting a win in the game is synonymous with determining which team is performing better, assuming a well-balanced game. While *player\_gold* is one measurement of performance, another are outcomes in team fights. A team may be behind in gold, but still outplay an opponent. Tracking only gold would inhibit the model’s ability to predict comebacks and upsets. In recent years, Valve’s development team have improved data storage to include metrics at time points. A log of team fights, including which hero killed whom, and individual hero gold progression by the minute have been added. Using the metrics captured in team fights, this paper aims to show that the increased detail in the features will contribute to a higher win prediction accuracy by revealing important relationships between gold acquisition, game time and team performance expressed during fights, rather than a simplified gold differential. The reasoning behind this expectation is due to an intended balance feature of the game. Heros are designed to be strong at different points in the game, as well as having different strengths such as being proficient at dealing damage to enemies or excelling at dealing tower damage. Heros typically receive a spike in power upon purchasing certain items at specific timings with a follow-up play by the team to capitalize on said timing. Using a simple gold differential between teams does not account for latent implications between gold amounts on specific heroes and its implication on the upcoming fight or interactions between heroes in fights. A goal of this paper is to evaluate the efficacy of including hero gold acquisition on minute marks and team fight interactions to more accurately judge which team is performing better.

# Statistical Methods & Variables

Anticipated models to be used for this are a combination of Support Vector Machine, Random Forest, and an LSTM. A Temporal Fusion Transformer model was considered for the task, but due to computing limitations was determined not applicable. Feature variables for use will be heroes chosen, the time-series of each heroes’ gold, and which team a hero belonged to. From the perspective of an LSTM model predicting on one time-series, hero\_id and team will be used as categorical variables with time-lagged features for all heroes present. In summary, this creates a multi-variate time-series prediction with two categorical variables and time-lagged features for each time-series. The number of time lags used will range from 4 to 10 with performance dictating which will be used in the final model. Since the LSTM model cannot handle categorical variables as they are, hero\_id will be represented as an embedding vector. Several sizes of embedding vectors will be tested ranging from 20 to 64. Since there are 124 different heroes a significantly sized one-hot encoding object would be needed that would likely impact model performance, thus an embedding vector is used. It is important to note that the embedding vector combined with the other heroes’ time-series data is the technique used to ensure that hero interactions are captured by the model and is the mechanical aspect that allows for the hypothesis to be evaluated and adding value to the knowledge base concerning real-time eSports prediction. The LSTM model will use the embedding for each hero to distinguish the unique hero relationships with gold, capturing which heroes gain more or less gold, as well as capturing which heroes are given priority in gaining gold by the team. The LSTM will predict a gold amount for the given time-step of each of the 10 heroes and pass these values, as well as the embedding vector and team variable, to a logistic regression layer to generate likelihood probabilities of winning. Challenges in this research will be finding techniques to utilize the anticipated time-series data of gold gain and team fights as features for models to work with similar to the research performed in [8] and [9].

A diagram of a diagram

Description automatically generatedAn LSTM model is a type of recurrent neural network (RNN) that handles the vanishing gradient problem often seen in other RNN’s by controlling the information in the cell states. The basic anatomy of an LSTM model contains two states and three gates. The first state, referred to as the cell state, carries information between iterations or time steps in the model. One can think of this as the long-term memory deemed important for the model. The second state, referred to as the hidden state, handles the most recently passed information to determine updates to the cell state by using the three gates. One can think of this as the short-term memory aspect of the model that gleans important information for longer term use.

*Figure A: Forget Gate*

(Dolphin, 2022) [10]

A diagram of a number of words

Description automatically generatedAs the time-step value is passed to the hidden state, the first gate called the ‘forget gate’ evaluates the information and updates values in the cell state to remember or forget the past information by point-wise multiplication. This is effectively viewing the current information in the hidden state for current context and determining past information in the cell state to determine which of it is still relevant. The forget gate uses a sigmoid activation function that forces mathematical values between 0 and 1 to accomplish this “remembering” or “forgetting” to alter the values in the cell state.

*Figure B: Input Gate*

(Dolphin, 2022) [10]

The second gate, called the ‘input gate’, contains two parts and is responsible for adding information to the cell state. The input gate first evaluates which information currently held in the hidden state should be used to update the cell state with a sigmoid function forcing values between 0 or 1. Secondly, it creates a vector of candidate values based on the current time-step in the hidden state. The vector of sigmoid values from the first part and the candidate values from the second are multiplied together to create the new values which are added to the cell state. *A diagram of a number of words

Description automatically generated*

*Figure C: Output Gate*

(Dolphin, 2022) [10]

A diagram of a diagram of a number of words

Description automatically generated with medium confidenceA diagram of a diagram of a number of words

Description automatically generated with medium confidenceThe third gate, called the ‘output gate’, controls what the hidden state will look like for the next iteration of the model by using the cell state. This would be akin to dealing with a current situation using one’s long-term memory. First, the output gate applies a tanh function, which forces values between -1 and 1, to the cell state. Secondly, the hidden state and current input data are concatenated and passed through a sigmoid activation function. The tanh values of the cell state and the sigmoid values of the hidden state are then point-wise multiplied together and passed on as the new hidden state to iterate over the next time-step as input data and repeat the process (Mohammadi et al., 2015).

A differentiator, of practical importance for companies, between this research and others is the lack of a Game State Integration used by [7], [8]. And [9] to collect all data available from a game for training purposes, and instead relying on the recorded minute mark values of variables in the historical database to re-create a “live-game” scenario for the model to train on. Papers focusing on live-game statistics with a type of game state integration tended toward smaller sample sizes due to the work involved with data collection. The advantage of using the recorded minute mark values of variables that this paper suggests is access to hundreds of thousands of samples that can be used as training data. The disadvantage is the lack of numerous other variables that a Game State Integration app is able to gather. The data will be collected using the OpenDota API that gives access to Valve’s database storing Dota 2 game metrics. The database is updated per game and thousands of games are played every day.

# Citations & References

[1] *DOTA 2 TI viewers 2019 | Statista*. (2019). Statista; Statista. https://www.statista.com/statistics/518201/dota2-championship-viewers/

[2] *DOTA 2 TI viewers 2019 | Statista*. (2019). Statista; Statista. https://www.statista.com/statistics/518201/dota2-championship-viewers/

[3] *Top eSports tournaments by prize pool 2022* | *Statista.* (2022). https://www.statista.com/statistics/517940/leading-esports-tournamets-worldwide-by-prize-pool/#:~:text=The%20eSport%20tournament%20that%20has

[4]Djundik, P. (2013, July 9). *Dota 2 Steam Charts* [Review of *Dota 2 Steam Charts*]. Steam Charts - SteamDB. https://steamdb.info/app/570/charts/#1y

[5]Yu, L., Zhang, D., Chen, X., and Xie, X., “*MOBA-Slice: A Time Slice Based Evaluation Framework of Relative Advantage between Teams in MOBA Games*”, arXiv e-prints, 2018. doi:10.48550/arXiv.1807.08360.

[6] [Song, K., Zhang, T., & Ma, C. (2015). *Predicting the winning side of DotA2.* https://cs229.stanford.edu/proj2015/249\_report.pdf](https://cs229.stanford.edu/proj2015/249_report.pdf)

[7] Akhmedov, K. and Phan, A. H., “*Machine learning models for DOTA 2 outcomes prediction*”, <i>arXiv e-prints</i>, 2021. doi:10.48550/arXiv.2106.01782.

[8] Yang, Y., Qin, T., and Lei, Y.-H., “*Real-time eSports Match Result Prediction*”, <i>arXiv e-prints</i>, 2016. doi:10.48550/arXiv.1701.03162.

[9] V. J. Hodge, S. Devlin, N. Sephton, F. Block, P. I. Cowling and A. Drachen, "*Win Prediction in Multiplayer Esports: Live Professional Match Prediction*," in IEEE Transactions on Games, vol. 13, no. 4, pp. 368-379, Dec. 2021, doi: 10.1109/TG.2019.2948469.

[10] Dolphin, R. (2022, February 28). LSTM Networks | A detailed explanation | towards Data science. *Medium*. <https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9>

[11] Mohammadi, M., Mundra, R., & Socher, R. (2015). CS 224D: Deep Learning for NLP. In *Lecture Notes: Part IV*. https://cs224d.stanford.edu/lecture\_notes/LectureNotes4.pdf