Predicting Game Outcome in Dota 2 with NLP and Machine Learning Algorithms
This exam work has been carried out at the School of Engineering in Jönköping in the subject area Computer Science. The work is a part of the two-year Master of Science in AI Engineering programme. The authors take full responsibility for opinions, conclusions and findings presented.

Examiner: Vladimir Tarasov

Supervisor: Tuwe Löfström

Scope: 30 credits (second cycle)

Date: 06. 2023.
Abstract

Esports has grown quickly in recent years, and the business has produced a ton of specifications-based data that is simple to obtain. Because of the aforementioned traits, data mining and deep learning techniques can be used to direct participants and create winning strategies. As one of the world’s most famous e-sports, Dota 2 has grown over time by luring many players from all over the world, and it recently held its tenth professional international tournament. A Dota2 game consists of a drafting and a playing phase. The drafting phase can influence the outcome of the game. However, players frequently struggle to select the strongest lineup to participate. In order to address this gap, we present in this article an NLP and Deep Learning hybrid model as an alternative approach for optimal drafting. We report state-of-the-art results on the problem of predicting the outcome of the match in the hero select phase of the game. Hero vectors are produced by the model using the Continuous Bag of Words (CBOW) in the Word2vec model. The context of a word in a sentence can be predicted by the CBOW model. The model used in this article suggests a predictive tool for players that can provide a more reliable method for the drafting phase. Accordingly, a word is changed into a hero, a phrase into a lineup, and a word vector into a hero vector. The improved Long short-term memory (LSTM) model is used to solve the influence of timing relationships affecting selection on the prediction of winning rate.

Keywords

NLP, CBOW, LSTM, DOTA 2, Machine Learning, Hero selection
1 Introduction

In recent years, esports has gained huge popularity worldwide. Professional esports teams and players now earn millions from tournaments and sponsorships. Esports events are also held in stadiums where thousands of fans gather to watch live tournaments. This sector is a sizable financial one, and its earnings have increased dramatically in recent years. The esports industry is anticipated to generate approximately $1.6 billion in revenue (2023), exhibiting a growth rate of around 50%. [1]. It has also gained recognition as a legitimate sport, with the International Olympic Committee (IOC) acknowledging its potential as a demonstration sport in the 2024 Paris Olympics [2].

Valve Co. has crafted Dota 2, a prominent multiplayer online battle arena (MOBA) game, which has garnered substantial popularity in the esports industry. Dota 2 boasts a considerable and dedicated community of fans, leading to the emergence of numerous professional teams and a plethora of competitive events. In fact, Dota 2 is one of the top-grossing esports games, with a total prize pool of $40,018,195 USD for its tenth professional international tournament in 2019 [3].

Dota 2 matchmaking is the process of grouping players with similar Matchmaking Rating (MMR) into teams to ensure balanced gameplay. MMR is a numerical representation of a player's skill level and is utilized to facilitate balanced matchmaking in ranked matches. The MMR value is determined based on match outcomes and the comparative skill levels of the involved players. Winning a match leads to an increase in MMR, while losing results in a decrease. The magnitude of MMR adjustments considers factors such as the skill levels of opponents and the player's individual performance. The major objective of the method is to ensure that players are paired with opponents of comparable skill, promoting an equitable and competitive gaming environment.

When players queue for a match, the game’s matchmaking system searches for other players of comparable skill and forms two teams. The goal is to create a fair and competitive team distribution for all participants. Once the teams are formed, the hero drafting phase begins. In this phase, players take turns choosing their heroes from the available options. Each hero has unique abilities and characteristics. The drafting phase involves strategic decision-making, as teams aim to create a well-rounded lineup with complementary hero choices and effective strategies. Following the drafting phase, the actual gameplay begins. Dota 2 matches are played on a map divided into two teams' territories, each containing structures and objectives. The teams must work together to destroy the enemy team's structures and ultimately their Ancient, a central building that serves as the ultimate objective. Players control their heroes, engage in fights, complete objectives, and strategize to gain an advantage over their opponents. The team that successfully destroys the enemy Ancient or forces their opponents to concede the match emerges as the winner.

In the drafting phase, teams ban heroes, which cannot be selected later, and then both teams follow some strategy to select 5-5 heroes each. The goal is to select champs that complement each other within the team and counters the opponent's lineup. This requires strategic thinking and knowledge of the game mechanics, hero
abilities, and player preferences. First, both teams ban two-tow heroes, then players take turns choosing which hero they want to play. In 1-2-2-1 order, the teams choose the champs. After each pick, players are given 30 seconds to think.

Conventional methods of hero selection, which rely on experience, intuition, and trial-and-error, can be inefficient and unreliable, highlighting the necessity for advanced, data-driven approaches that harness the extensive data available in the game to achieve optimal lineups.

Recent studies have demonstrated that player performance may be enhanced by using machine learning and data mining approaches to anticipate the results of esports matches. For example, they [4] developed a machine learning model to predict the outcome of League of Legends (a very similar game to Dota2) matches based on game statistics and individual player experience with the selected hero. Similarly, another team [5] used data mining techniques to analyze the behavior of Dota 2 players and identify key factors that contribute to winning.

In this thesis, we propose a novel NLP and Deep Learning hybrid model that leverages data mining and machine learning techniques to predict the outcome of a Dota 2 match in the hero selection phase. Our model utilizes the Continuous Bag of Words (CBOW) in the Word2vec model to produce hero vectors, and these vectors capture the semantic relationship and context between heroes. We also employ an improved Long short-term memory (LSTM) model to address the influence of timing relationships affecting selection on the prediction of winning rate.

Our approach builds on the existing literature on data-driven approaches to esports and provides a new perspective on hero selection in Dota 2. We believe that our approach can contribute to the growing body of research on esports analytics and provide valuable insights into game mechanics and player behavior.

1.1 Delimitations
The study will focus on predicting the outcome of Dota 2 matches after the hero selection phase using NLP and deep learning techniques. The study will not take into account external factors that may influence the outcome of the match, such as banned heroes, player skill, team strategy, or network latency. The research will be based on a specific dataset of Dota 2 matches and may not generalize to other contexts or games.

1.2 Objectives
Develop a hybrid NLP and deep learning model for predicting the outcome after the hero selection phase of Dota 2 matches. The model will use the Continuous Bag of Words (CBOW) algorithm to produce hero vectors, and an improved Long Short-Term Memory (LSTM) architecture to account for the timing relationships of hero selection.
Evaluate the performance of the proposed model against other existing approaches and benchmarks, such as the results of previous studies on hero drafting, and through a detailed analysis, we will see whether our chosen LSTM model is superior to other commonly used machine learning methods. This will help us understand the limitations and potential of the proposed approach.

Discuss the practical implications of the proposed model for esports teams and players, and suggest possible directions for future research in this field.

Research questions to which we try to find answers with our thesis:

- Does the drafting phase in Dota 2 have a significant impact on the outcome of matches?
- Does the order in which the heroes are selected matter in predicting the winning team?

To prove the second research question, we will compare several different machine learning models, and we will measure their accuracy in predicting the winning team correctly. If the LSTM model consistently outperforms the other machine learning models, it would provide evidence to support the claim that the order in which heroes are selected during the drafting phase does matter in predicting the winning team.

1.3 Thesis Outline

The next chapter introduces the background of the research, including an overview of the algorithms and a comprehensive literature review. Moving forward, the subsequent chapter will outline the methods employed in this research. It will detail the dataset used, elucidate the Continuous Bag of Words (CBOW) algorithm employed for word embedding, describe the data processing techniques implemented, and introduce the Long Short-Term Memory (LSTM) architecture for sequence modeling. Following the methods section, the chapter will present the results obtained from the conducted experiments. These results will showcase the performance and effectiveness of the proposed approach in achieving the research objectives. Finally, the conclusion chapter will summarize the key findings, discuss their implications, and provide insights into the potential future directions of research in this area.
2 Related Work

This section provides an overview of the existing research and theoretical foundations that inform our hybrid machine learning model. First, we present the essential machine learning techniques and theories relevant to our research. Subsequently, we examine previous studies on machine learning models applied to Dota 2.

2.1 Theoretical Background

2.1.1 Log5 Formula

The log5 formula, developed by the influential sports analytics figure Bill James, is extensively utilized in various sports to gauge the likelihood of one team winning over another in competitive sports. The log5 formula is a statistical method used to evaluate the comparative abilities of teams or individuals. This formula employs the log-odds concept to calculate the winning probability of Team A against Team B by logarithmically representing the event's probability. By averaging the log-odds of both teams, it estimates Team A's likelihood of winning in the matchup. By taking into account the teams' historical performance and converting their winning percentages into probabilities, the log5 formula offers a straightforward methodology to evaluate team strengths and predict outcomes in head-to-head competitions.

It provides a simple yet effective way to assess the relative strengths of competing entities. The formula is derived from the concept of winning percentage. In its basic form, the Log5 formula is expressed as

\[ p_{a,b} = \frac{p_a - p_a \times p_b}{p_a + p_b - 2 \times p_a \times p_b} \]

Where \( p_{a,b} \) denotes the likelihood that team A will prevail over team B, \( p_a \) denotes the global win rate defined as the likelihood that team A will appear in and win all games, and \( p_b \) is the global win rate defined as the likelihood that team B will appear in and win all games.

It's important to keep in mind that, despite being a simple and popular method, the Log5 formula makes some assumptions and might not take all elements affecting the outcome of a sporting event into consideration. Because of this, it is frequently used as a starting point for research and can be improved by more complex models that take other factors and contexts into account [6].

2.1.2 CBOW

Word2Vec is a widely used neural network-based model for learning word embeddings, which are distributed representations of words in a continuous vector space. It was introduced by Mikolov et al. (2013) [7].

The main objective of Word2Vec is to capture the semantic and syntactic similarities between words. It leverages the distributional hypothesis, which suggests that words appearing in similar contexts are likely to have similar meanings.
Word2Vec consists of two primary architectures: the Continuous Bag-of-Words (CBOW) model and the Skip-gram model. Both models aim to learn word embeddings by predicting the context of a given word or predicting a word based on its context. The main difference between them is their objective and input. CBOW predicts a target word from context words, while Skip-gram predicts context words given a target word, resulting in different learning approaches for capturing word embeddings.

The Continuous Bag-of-Words (CBOW) model is a popular neural network architecture used for natural language processing (NLP) tasks, particularly in word embedding. It is a variant of the Word2Vec model, which aims to capture meaningful and distributed representations of words in a continuous vector space. The CBOW model is specifically designed to predict a target word given its surrounding context words. It takes a fixed-sized window of context words as input and tries to predict the target word in the middle. The window size determines how many context words are considered on either side of the target word.

The architecture of the CBOW model consists of three main components: an input layer, a hidden layer, and an output layer. The input layer encodes the context words as one-hot vectors, where each word is represented by a vector with a 1 in its corresponding position and 0s elsewhere. These one-hot vectors are then multiplied by a word embedding matrix to obtain continuous vector representations, also known as word embeddings.

The hidden layer is responsible for learning the relationships between the context word embeddings. It takes the average of the context word embeddings and passes them through a fully connected layer, applying weights to capture the semantic connections between words.

The output layer performs the prediction task. It takes the learned representation from the hidden layer and applies a softmax activation function to generate a probability distribution over the entire vocabulary. The predicted probability distribution represents the likelihood of each word in the vocabulary being the target word given the context.

During training, the CBOW model aims to minimize the difference between the predicted probabilities and the actual target word. It uses backpropagation and gradient descent to adjust the weights in the hidden and output layers, gradually improving the model's ability to predict the target word accurately.

Once trained, the CBOW model yields word embeddings that capture semantic relationships between words. Words with similar meanings or that frequently appear in similar contexts tend to have similar vector representations in the embedding space. These word embeddings can be used in various downstream NLP tasks, such as sentiment analysis, machine translation, or document classification, to enhance the performance of the models by leveraging the semantic knowledge encoded in the embeddings.

Overall, the CBOW model offers a computationally efficient approach to learn word representations and capture the meaning of words based on their surrounding context. It has proven to be effective in capturing semantic relationships and has become a valuable tool in the field of NLP. The architecture of a CBOW model can be seen on Figure 1.
2.1.3 LSTM
LSTM is a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data. It addresses the vanishing gradient problem of traditional RNNs by incorporating memory cells, input gates, forget gates, and output gates. The LSTM model is useful for processing and understanding sequential data, such as text or time series data. The main advantage of LSTM is its ability to retain information over long sequences, enabling it to capture dependencies that may occur far apart in the input data. This makes it suitable for tasks like language modeling, machine translation, sentiment analysis, and many others.

The attention mechanism is an enhancement to the LSTM model that allows it to focus on specific parts of the input sequence while making predictions. It addresses the limitation of the traditional LSTM, which treats all input elements equally. By assigning weights to different parts of the input sequence, the attention mechanism enables the model to dynamically focus on the most relevant information. This mechanism is particularly useful when dealing with long sequences, as it helps the model give more importance to relevant context and suppress irrelevant or noisy information. Attention has been widely adopted in various natural language processing (NLP) tasks such as machine translation, text summarization, and question-answering systems.

The attention mechanism helps improve model performance by attending to specific parts of the input, but it also introduces additional computational complexity and parameter tuning requirements. Depending on the design, attention can be computationally expensive, especially for large input sequences [9].

The multiplication layer, also known as the element-wise multiplication or Hadamard product, is a mathematical operation applied element-wise between two tensors. Multiplication layers are commonly used in neural network architectures to introduce non-linear interactions between features or to weigh the importance of different input components. They can help capture complex relationships and interactions between features, leading to more expressive and effective models.
Theoretical background

The multiplication layer is a versatile component, but its limitations depend on the specific application and context in which it is used. It can introduce additional parameters and increase computational complexity, especially if applied on large tensors. Additionally, if not carefully designed or trained, the multiplication layer could potentially amplify noise or irrelevant information \[10\].

The concatenation layer is a method of combining multiple tensors along a particular axis, typically the feature axis, to create a single, unified tensor. Concatenation layers are commonly used to aggregate features or representations from different parts of a network. By combining information from multiple sources, the model can benefit from a broader context or incorporate different modalities, leading to enhanced performance in various tasks.

The concatenation layer's usefulness lies in its ability to capture diverse information and enable the model to exploit complementary aspects of the input. However, concatenation can significantly increase the dimensionality of the data, leading to higher computational requirements \[11\].

2.2 Literature Review

There has been some research on using LSTM for win prediction in Dota2. The research paper \[12\] aims to build predictive machine and deep learning models for real-time prediction using player data collected during the game, using a new method of multi-forward step predictions. Three models were investigated and compared: Linear Regression (LR), Neural Networks (NN), and a type of recurrent neural network LSTM. Real-time data of players were collected using a Python server developed using Game State Integration (GSI). The achieved accuracy scores of the models depend on the multi-forward prediction parameters, with the worst-case accuracy being 69% for linear regression and an average accuracy of 82%, while the deep learning models achieved the highest accuracy, with an average of 88% for NN and 93% for LSTM models.

The paper \[13\] proposes a novel approach to predict the winner of DOTA 2 matches by incorporating both individual player and team performance metrics. The proposed method includes a feature extraction process using a deep neural network, followed by the application of a multi-task learning approach to predict both individual and team performances. They explored how team fights influence the outcome of a game. In Dota 2, team fights occur during gameplay (after hero selection), it refers to the pivotal moments when groups of heroes from opposing teams engage in combat. The model achieved an accuracy of 85.5% in predicting the winning team, outperforming existing methods. The study concludes that the proposed method can be applied to other esports titles with similar game mechanics.

Another research \[14\] was the first to use Logistic Regression and k-Nearest Neighbors (kNN) to show the significance of data from the game's draft stage. On the 18,000 training data points for Logistic Regression, they obtained a test accuracy of 69.8%, however, it was unable to depict the cooperative and competitive
dynamics among heroes inside and across teams. The authors chose the d
dimension value for kNN using distance metrics and 2-fold cross-validation in order
to overcome that problem. They achieved 70% accuracy on testset and 67.43%
accuracy on cross-validation for an ideal d-dimension of 4. The authors created a
web-based recommendation engine based on those findings. The biggest
disadvantage of the developed solution is that the running time of one model took
4-12 hours.

On the basis of hero lineups, article [15] experimented using logistic regression to
determine which team would win a Dota 2 match. The game developer's API was
used by them to collect data. Their best achieved accuracy was around 53%. The
paper concludes that predicting the winning side based solely on hero lineups is not
sufficient and suggests that future work should consider player information to
improve prediction accuracy. Additionally, the authors mention the need for
potential model changes to achieve better results.

Another study [16] presents an approach for predicting the outcome of DOTA 2
matches based on the game's draft phase. The proposed method utilizes a deep
neural network to extract features from the draft data, which are then fed into a
logistic regression model for prediction. The study compares the performance of
the proposed method with several baseline models, including random guess and
expert prediction. The results show that the proposed method outperforms the
baselines and achieves an accuracy of 65.6% in predicting the winning team. The
authors conclude that their method provides a promising approach for DOTA 2
match outcome prediction, but further improvements can be made by incorporating
additional data and refining the model.

The study [17] explores the use of the Naive Bayes classifier to analyze DOTA 2
lineups and predict game outcomes based on player choices. They explain that the
Naive Bayes classifier was chosen because of its stability and simplicity, and it
assumes independence among the attributes and the heroes in DOTA 2 are
independent of each other. The results of the experiment show that the accuracy of
the classifier on the training set is 85.33%, while the accuracy on the test set is
58.99%.

The authors of the paper [18] proposed a recommendation system for selecting
heroes in Dota2. The recommendation system used the Apriori algorithm (This
method uses relational databases to learn association rules and frequently occurring
item sets. Not relevant in our research.) to extract association rules between heroes.
They created two sets of association rules: one for heroes that won together and
one for heroes that won against specific opponents. These association rules were
used to recommend heroes based on allies and enemies. In terms of results, the
recommendation system had a success rate of up to 74.9% in recommending hero
line-ups. The experiments showed that using the win rate of heroes as the metric
for allies recommendation and the counter confidence as the metric for enemies
recommendation resulted in the highest success rate. The model differs from the
one we present in our research in that they recommended the most appropriate choice based on the current state of hero selection.

In conclusion, the literature review reveals a multitude of research studies focused on predicting Dota 2 game outcomes. While many of these papers explore the influence of hero selection on match results, it is important to note that these studies do not solely rely on this aspect. Instead, they take into account various stages of the game to develop comprehensive predictive models. These models consider factors such as team composition, player skill levels, in-game decision-making, map control, and objective prioritization. By examining beyond hero selection, these studies highlight the intricate and multifaceted nature of Dota 2 prediction, emphasizing the significance of analyzing the game holistically. As the literature review has shown, many promising results have been achieved by researchers in various aspects of the game, but the hero selection phase has been studied only by a few, and no outstanding results have been achieved. Also, no one has considered the hero selection order. In our paper, we aim to address this gap by applying a new hybrid model.
3 Methods

3.1 Dataset

To gather data for our study, we utilized the OpenDota API and obtained a dataset consisting of 20,000 high-ranking game records. These games were specifically selected based on their matchmaking ranking, ensuring that we focused on matches featuring skilled and experienced players. The average MMR must be over 6000.

Examining only high-ranking games, where the matchmaking ranking is over 6000, is important for several reasons. Firstly, high-ranking games are generally considered to represent a higher level of skill and strategic gameplay. By focusing on these games, we can gather data that reflects the performance and decision-making of experienced and proficient players. This ensures that our model captures the nuances and complexities of competitive Dota 2 matches. Furthermore, high-ranking games often involve players who have a deeper understanding of the game mechanics, advanced strategies, and effective teamwork. Analyzing these games allows us to uncover patterns and insights that may not be as prevalent in lower-ranking matches. This can lead to a more accurate and robust predictive model. However, it is crucial to acknowledge the potential limitations of an imbalanced player ranking within the gathered data. If the player rankings are significantly skewed, it can introduce bias into the model's predictions. An imbalance in player rankings may result in one team having a considerable advantage over the other, leading to predictable and less representative outcomes. To mitigate this issue, it is important to carefully consider the composition and distribution of player rankings within the dataset.

The data retrieved from the API was organized and stored in a convenient data structure known as a dataframe. This dataframe allows for efficient storage and manipulation of the game data, facilitating subsequent analysis and modeling. By collecting a substantial number of high-ranking game histories, we aim to capture a representative sample that reflects the strategic and competitive aspects of Dota 2 gameplay at an advanced level. This dataset will serve as a valuable resource for our research, enabling us to derive insights, uncover patterns, and develop accurate predictive models. Overall, the script collects data on the hero selections and pick order for 20,000 professional Dota 2 matches and stores them for use in the NLP and Deep Learning hybrid model. The following data were relevant to us from the data:

- **Match id:** is a unique identifier assigned to each match in Dota 2 game. It is used to retrieve detailed information about the match, such as the teams playing, the heroes picked, and the outcome of the match.
- **Radiant pick:** is a list that contains the names of the heroes picked by the Radiant team in a particular Dota 2 match.
- **Dire pick:** it refers to the heroes that the Dire team selects during the drafting phase of the game. These heroes will then be used to play against the Radiant team's heroes in the subsequent playing phase.
- **Radiant pick order:** This is a list that shows the order in which the heroes were picked by the Radiant team. It contains integer values indicating the order of each pick.

- **Dire pick order:** a list of integers representing the order in which the heroes were picked by the dire team in a particular match. The order starts at 0 and increases by 1 for each hero pick.

- **Radiant win:** This is a boolean variable that indicates whether the Radiant team won the match or not.

### 3.2 Natural Language Processing CBOW

The composition of the Dota 2 lineup involves the intricate dynamics between heroes, encompassing both hero matching and restraint relationships. To tackle this issue, the hero vector method is adopted in this section to address the hero matching problem, while the restraint relationship problem is resolved using the log5 formula. In this study, the CBOW model is employed to handle the hero matching challenge. As described in section 2.1.2, the CBOW model is capable of predicting a word's context based on its position within a sentence and capturing the relationship between multiple words. In our approach, we treat a hero lineup as a sentence comprised of individual hero "words" generating a hero vector representation. The correlation between words is analogous to the correlation between heroes when viewed as word vectors.

Using Figure 1, the parameters of our CBOW model are: W(t-2) - W(t+2) vector is shown as a one-hot vector and is the input hero vector. The hero vector dimension (d) and the projection unit dimension (N) together make up the data dimension. The dimensions of the matrixes leading from the input layer to the projection layer are N x d. And output vector W(t) with a dimension 2d. This choice is made to accommodate the objective of the hero matching problem, which involves predicting the context or relationships between heroes. By having a 2d-dimensional output vector, the model can capture both positive and negative associations between heroes. For example, if a positive association exists between two heroes in terms of synergy or complementarity, the model can learn to assign higher values to the corresponding dimensions of the output vector. Conversely, if there is a negative association due to hero counters or weaknesses, the model can assign lower values to the relevant dimensions. The output vector with 2d dimensions provides the flexibility to capture both positive and negative hero relationships effectively.

In the context of word vector training, the separation between word vectors representing words with similar meanings should be minimal. This means that words that share comparable semantic or contextual attributes should have vectors that are close together in the vector space. For example, in a trained word vector model, the vectors for words such as "king" and "queen" would be located very close to each other, reflecting their similar meanings. In the case of Dota 2 heroes, when using the hero vector representation, certain melee heroes like "Clockwerk" and "Sand King" would be categorized as "Melee." As a result, their hero vectors would be positioned close to each other in the vector space, indicating their shared
attribute of being melee heroes. This proximity between hero vectors allows for a more nuanced understanding of the relationships and similarities between Dota 2 heroes based on their respective vectors.

When limiting factors or conditions are present in a situation, different outcomes are usually achieved by different participants, even if the difference in outcomes does not arise in an unfair or discriminatory way, but simply due to different characteristics or abilities of the participants. When forming a lineup in Dota2, the concept of heroic restraint is crucial. Heroic restraint can be described as the ability to impact the enemy hero's alignment, development, role, late-game outcome, and location in group combat with minimal effort, such as using a small skill. It is difficult to analyze the restraint relationships in Dota2 objectively because they frequently depend on the experiences of players after many games. To solve this problem, we calculate each hero's win probability against other heroes using the log5 formula, which was described in section 2.1.1.

The computed probability is the anticipated likelihood that one hero will defeat another. A hero with a high winning average has a high expectation of winning against a hero with a low winning average, thus if it is defeated by a hero with a low winning average, there is a strong restraint relationship between the two heroes. This means that the high-winning hero is againsted by the low-winning hero, and in the match against the low-winning hero, the winning rate will be lower than average. For example, the overall win rate of Medusa, a hero, is 56.8%, which is very high. Nyx Assassin Hero has a global rating of 51.3%. So Medusa should have a high probability to win against Nyx Assassin, but the actual win rate is only 44.97%. According to the Log5 formula, the predicted win rate of Medusa against Nyx Assassin is 53.7%. This knowable index indicates that Medusa has a better overall win rate, but it is opposed by Nyx Assassin. As a result, the hero restraint relationship index in this study is defined as the predicted probability of victory among heroes determined using the log5 algorithm.

The counter_rate csv file created in this way is actually an N x N matrix containing the relationship of the heroes, which defines how each hero is related to the others. The structure of the matrix looks like this:

\[
R = \begin{bmatrix}
  r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\
  r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  r_{n,1} & r_{n,2} & \cdots & r_{n,n}
\end{bmatrix}
\]

where \( n \) is the number of heroes available in Dota 2 and 
\[
t_{xy}, 0 < x, y \leq n, \ x, y \in \mathbb{Z}
\]
is the relationship index between Hero(x) and Hero(y). The numbers in the diagonal are all 0.

The dendrogram generated from the similarity matrix (Fig 2.) of the 123 Dota2 heroes provides valuable insights into the relationships and groupings between
Method and implementation

heroes. The dendrogram reveals that some heroes are closely related and grouped into clusters, while others are more distinct and fall into isolated branches. For example, the hero “Axe” is clustered with other strength-based heroes like “Sven” and “Wraith King”, while the intelligence-based heroes like “Zeus” and “Lina” form a separate cluster. The dendrogram also highlights some interesting patterns and similarities between heroes, such as the grouping of heroes with similar roles or playstyles. This analysis demonstrates the utility of dendrograms as a powerful tool for visualizing complex datasets in Dota2 and other video games. By examining the clusters in the dendrogram, players could identify groups of heroes that share similar characteristics or playstyles. This information can be used for the purpose of choosing the most suitable hero, team composition analysis, or game strategies. For example, if the dendrogram reveals that certain heroes are often picked together due to their synergy, they can take that into account when struggling to pick the best hero combinations or predicting optimal team compositions. Also, a recommendation system could be created from these data.

![Figure 1. Dota 2 Hero Relationship Dendrogram](image)

The Word2Vec model used in this study was generated with specific parameter settings to optimize the learning of word embeddings. The model was configured with a vector size of 150, which determines the dimensionality of the generated word vectors. A window size of 5 was chosen, indicating the maximum distance between the target word and its surrounding context words within a sentence. The min_count parameter was set to 1, meaning that words occurring at least once in the dataset were included in the vocabulary. To expedite the training process, 8 workers were utilized for parallel processing. The sg parameter was set to 0, indicating the use of the CBOW algorithm rather than skip-gram. Finally, the model
was trained for 10 epochs, where each epoch represents a complete iteration over the training dataset. The parameter values are summarized in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector size</td>
<td>150</td>
</tr>
<tr>
<td>Window</td>
<td>5</td>
</tr>
<tr>
<td>Min count</td>
<td>1</td>
</tr>
<tr>
<td>Workers</td>
<td>8</td>
</tr>
<tr>
<td>Sg</td>
<td>0</td>
</tr>
<tr>
<td>Epochs</td>
<td>10</td>
</tr>
</tbody>
</table>

### 3.3 Data Preprocessing

Once the data has been prepared using the Word2Vec model and the counter matrices have been computed using the log5 formula, the next step is to divide the data into train, test, and validation sets. To ensure reliable evaluation of the machine learning models, a 10-fold cross-validation data separation technique is employed. This process involves splitting the dataset into ten equally-sized subsets or "folds". In each iteration, one fold is used as the validation set, while the remaining nine folds are combined to form the training set. This process is repeated ten times, with each fold taking turns as the validation set. By utilizing this approach, the models can be trained and evaluated on multiple variations of train-test splits, reducing the risk of overfitting and providing a more robust assessment of their performance. Ultimately, this data separation technique enables the development of machine learning models that can effectively generalize unseen data and make accurate predictions in real-world scenarios.

### 3.4 LSTM

The structure of the model used in this research is the following and shown in Figure 3:

- The LSTM model used in this study consists of four input layers. Two input layers are used to feed in the preprocessed data for each of the two teams. Each of these input layers has a shape of (5, 150), which means that the team picks 5 heroes in order, one pick is one time step, and each hero has a word vector of 150 features. The other two input layers are used to provide the average counter-pick rates for each hero of the two teams. These input layers have a shape of (5,1), which means that they contain a sequence of five values, each representing the average counter-pick rate for a particular time step. The purpose of these input layers is to provide additional information to the LSTM model so that it can learn to predict the outcome of a match not only based on the hero picks but also based on hero relationship.
Two LSTM layers with 150 hidden units each are used in the model. The input shape of each LSTM layer was (batch_size, time_steps, input_dim), where the batch_size was the number of samples in each batch, the time_steps were the number of time steps in each sequence, and the input_dim was the number of features for each time step. In this model, the input shapes for the LSTM layers were (batch_size, 5, 150), as each input sequence had 5 time steps, and each time step had 150 features.

The attention mechanism is used in this LSTM model to enable it to focus on certain parts of the input sequence that are more relevant to the task at hand. This model uses two attention layers, one for each LSTM layer. The attention layer takes the output of the LSTM layer as input and applies an attention mechanism to generate a weighted sum of the output sequence. The first layer is applied to the output of the first LSTM layer, and the second layer is applied to the output of the second LSTM layer. Each attention layer uses a 3D tensor as input and outputs a tensor of the same shape. The attention mechanism is implemented as a dot product between a weight vector and the output of the LSTM layer. The weight vector is learned during training and determines the importance of each time step in the input sequence.

After applying the attention mechanism, the output sequence of each LSTM layer is multiplied element-wise with a vector that represents the attention weights. This step serves to emphasize the most important parts of the input sequence and downplay the less important parts.

The model uses two multiplication layers to compute element-wise multiplication between the output of the attention layers and the counter rates. The first multiplication layer takes the output of the attention layer from the LSTM layer applied on input_layer0 and the counter_rate0 input tensor as input. The second multiplication layer takes the output of the attention layer from the LSTM layer applied on input_layer1 and the counter_rate1 input tensor as input. The counter rates are normalized before being fed to the model.

These multiplication layers are crucial in allowing the model to weigh the importance of the different attention elements based on the corresponding counter rates. This ensures that the model is able to effectively capture the importance of the different attention elements in predicting the output.

The Concatenate layer is used to combine the outputs of the two multiplication layers. The inputs to this layer are the outputs of the two multiplication layers, which have the same shape (batch_size, time_steps, 150). The Concatenate layer concatenates these two outputs along the last dimension (axis=-1), resulting in an output tensor of shape (batch_size, time_steps, 300). This layer is important as it allows the model to combine the features of input1 and input2 which are processed by the attention mechanism to provide richer feature expression and achieve better model performance.
• The concatenated tensor is then passed through a fully connected dense layer with 256 units and ReLU activation, followed by another fully connected dense layer with 64 units and ReLU activation. The Flatten layer is used to convert the output of the previous layer into a 1D vector that can be fed to a fully connected layer.

• The Output layer is the final layer of the LSTM model and produces the predicted output. In this model, it consists of a single neuron with a sigmoid activation function, which outputs a probability value between 0 and 1. This probability value represents the likelihood of the input sequence belonging to the positive class. The output layer is trained using binary cross-entropy loss and accuracy as the evaluation metric. The model aims to minimize the loss and maximize the accuracy during the training process, which helps it to produce accurate predictions on unseen data.

3.4.1 LSTM Hyperparameters
A machine learning model's performance and behavior are greatly influenced by its hyperparameters. They impact the model's capacity to learn, its ability to generalize to new data, the speed of convergence during training, and the computational efficiency. Making informed choices and carefully tuning hyperparameters can lead to improved model performance and more accurate predictions.

The hyperparameters we chose were selected after much deliberation and are presented in this section, with a brief explanation of why they were the best choice.

The number of LSTM units determines the complexity and representational power of the LSTM layers. Since the vector size in the CBOW model is set to 150, it means that each hero has 150 features, so using an LSTM layer with 150 units allows the model to capture intricate patterns and dependencies in the input sequences while balancing computational complexity.

Using 64 units in the first dense layer introduces non-linearity and allows for the extraction of higher-level features and representations from the concatenated inputs. The number of units in the second dense layer adds another layer of non-linear transformations and increases the model's capacity to learn complex patterns and relationships in the data.

The learning rate determines the step size at which the model adjusts its weights during training. Choosing a small learning rate like 0.0001 helps ensure stable and gradual convergence of the model's optimization process, especially in cases where the data or the model architecture is complex.

Binary cross-entropy loss is commonly used for binary classification problems. It is well-suited for predicting the outcome of matches since it measures the dissimilarity between the predicted probabilities and the true labels, encouraging the model to produce accurate probability values for each class [19].
Adam is a popular optimization algorithm that combines the benefits of both AdaGrad and RMSprop. It adapts the learning rate based on the estimates of the first and second moments of the gradients, allowing for efficient and effective optimization. Adam is commonly used in deep learning models due to its good convergence properties and robustness [20].

Early Stopping is used to stop training early if the model's performance on a validation set does not improve for a specified number of epochs, preventing overfitting. Reduce LR on Plateau reduces the learning rate if the model's validation loss plateaus, helping the model to fine-tune its optimization process and potentially escape local minima.

Choosing 500 epochs provides a sufficient number of iterations to allow the model to learn from the data and adjust its weights, aiming for convergence and optimal performance.

The batch size represents the number of training samples processed before updating the model's weights. A larger batch size like 2000 reduces the frequency of weight updates, leading to more stable gradients and potentially faster training when parallel processing resources are available.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM units</td>
<td>150</td>
</tr>
<tr>
<td>Dense layer1 units</td>
<td>64</td>
</tr>
<tr>
<td>Dense layer units</td>
<td>128</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.0001</td>
</tr>
<tr>
<td>Loss function</td>
<td>Binary cross-entropy</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
</tr>
<tr>
<td>Callbacks</td>
<td>Early stopping, Reduce LR on Plateau</td>
</tr>
<tr>
<td>Number of epochs</td>
<td>500</td>
</tr>
<tr>
<td>Batch size</td>
<td>2000</td>
</tr>
</tbody>
</table>

Table 2. Hyperparameter setting values
Figure 2. LSTM architecture
4 Findings and analysis

In the LSTM model described, the initial epoch was set to 500 to allow for extensive training and refinement of the model's performance. However, during the training process, the model exhibited early stopping at the 175th epoch. This means that the model reached a point where further training did not result in significant improvement in performance. After 25 epochs, the model already achieved a relatively high accuracy of 70% and a loss of 0.58. This indicates that the model was learning effectively and making accurate predictions on the training data. As the training progressed, the accuracy improved further, reaching around 73%, while the loss decreased to 0.56. These metrics remained consistent not only on the training data but also on the test data, suggesting that the model was able to generalize well and make accurate predictions on unseen samples. Overall, the early stopping at the 175th epoch, with an accuracy of 73% and a loss of 0.56, indicates that the model successfully learned the patterns and relationships in the data, achieving a satisfactory level of performance. The results of the training are shown in Figure 4. The hyperparameter setting used by the model is shown in Table 2.

4.1 Hypothesis Testing

To test the hypothesis that hero composition has a significant effect on the outcome of Dota 2 matches, we performed our model (CBOW with LSTM) analysis. We set up two hypotheses: the null hypothesis that hero composition does not affect the match outcome and the alternative hypothesis that hero composition does have a significant effect on the match outcome.

To perform the hypothesis testing, we used the CBOW with LSTM model to predict the probability of a team winning a match based on the hero composition of each team. We then compared the predicted probabilities to the actual match outcomes and calculated the accuracy of the predictions.

We calculated the chi-squared statistic and perform the test using the provided results:
Findings and analysis

Observed Frequencies:
- Correctly predicted wins (a): 991
- Incorrectly predicted losses (b): 574
- Incorrectly predicted wins (c): 500
- Correctly predicted losses (d): 1183

Expected Frequencies: Under the null hypothesis, assuming a 50% chance of accuracy:
- Expected wins (c): \( (a + b) \times 0.5 = (991 + 574) \times 0.5 = 782.5 \)
- Expected losses (d): \( (c + d) \times 0.5 = (500 + 1183) \times 0.5 = 841.5 \)

Then we calculated the chi-squared statistic:

\[
X^2 = \sum \frac{(O - E)^2}{E} = \frac{(991 - 782.5)^2}{782.5} + \frac{(574 - 841.5)^2}{841.5} + \frac{(500 - 782.5)^2}{782.5} + \frac{(1183 - 841.5)^2}{841.5} = 395.5
\]

Next, we needed to determine the degrees of freedom (df), which is (number of rows - 1) \( \times \) (number of columns - 1). In this case, \( df = (2 - 1) \times (2 - 1) = 1 \).

The critical value depends on the chosen significance level \( (\alpha) \) and degrees of freedom (df). Assuming \( \alpha = 0.05 \), you can consult a chi-squared distribution table or use statistical software to find the critical value. For \( df = 1 \) and \( \alpha = 0.05 \), the critical value is approximately 3.841.

Since the calculated chi-squared value (395.5) exceeds the critical value (3.841), we can reject the null hypothesis.

The analysis of our showed that hero composition has a statistically significant effect on the outcome of Dota 2 matches (\( p < 0.05 \)). The accuracy of the model was found to be 73%, indicating that hero composition is a strong predictor of match outcomes.

Based on these results, we reject the null hypothesis and conclude that hero composition does have a significant effect on the outcome of Dota 2 matches. These findings suggest that teams should carefully consider their hero composition when drafting for a match in order to increase their chances of winning.

4.2 Model comparison
The results of this study indicate that the CBOW with LSTM model was able to accurately predict the outcome of Dota 2 matches with an overall accuracy of 73%. This suggests that the model has some level of predictive power and could be useful in predicting future match outcomes.

To ensure a comprehensive evaluation of the LSTM model's performance and to assess its ability to achieve optimal results, we compared it with several other machine learning algorithms. For this analysis, we divided the data into training and test sets using an 80-20% split, where the distribution of won and lost matches is equal. This balanced distribution allowed for an unbiased evaluation of all the algorithms, providing a fair comparison of their predictive accuracy and generalization capabilities. By conducting this thorough comparison, we gained
valuable insights into the strengths and weaknesses of each model and determined whether the LSTM model outperformed the other algorithms in accurately predicting match outcomes. The result is shown in Table 2.

Table 3. Machine Learning algorithms comparison

<table>
<thead>
<tr>
<th></th>
<th>LSTM</th>
<th>Random Forest</th>
<th>Gradient Boost</th>
<th>Gaussian Naive Bayes</th>
<th>KNN</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>73.13 %</td>
<td>63.36 %</td>
<td>67.76 %</td>
<td>60.49 %</td>
<td>54.37 %</td>
<td>67.21 %</td>
</tr>
<tr>
<td>Precision</td>
<td>.72</td>
<td>.63</td>
<td>.68</td>
<td>.6</td>
<td>.54</td>
<td>.67</td>
</tr>
<tr>
<td>Recall</td>
<td>.68</td>
<td>.56</td>
<td>.65</td>
<td>.6</td>
<td>.55</td>
<td>.66</td>
</tr>
<tr>
<td>F1-score</td>
<td>.69</td>
<td>.62</td>
<td>.6</td>
<td>.55</td>
<td>.54</td>
<td>.66</td>
</tr>
</tbody>
</table>

Among the algorithms tested, the LSTM model exhibited the highest performance, achieving an accuracy of 73.13%. The LSTM model’s ability to capture long-term dependencies and retain important information over time proved advantageous for our task.

Along with these prominent machine learning algorithms, we also assessed KNN, Logistic Regression, Gradient Boost, Random Forest, and Gaussian Naive Bayes. While each algorithm demonstrated respectable performances, the LSTM model outperformed them all, surpassing the accuracy achieved by any other algorithm. The comparison clearly illustrated the superiority of the LSTM model in this task. This also shows that the hero selection order plays an important role in the drafting phase of the game. These findings highlight the value of utilizing LSTM models for this and similar tasks and contribute to the existing body of knowledge in the field of machine learning.

Overall, the results of this study provide promising evidence that the CBOW with LSTM model could be a useful tool for predicting Dota 2 match outcomes. However, further research is needed to fully evaluate the model’s performance and to determine its usefulness in practical applications.
5 Discussion and conclusions

In this study, our main objective was to implement a CBOW-LSTM hybrid model in the context of Dota 2 to predict the outcome of a game during the hero selection phase. Our primary goal was to demonstrate the influence of hero selection on the final outcome of the game. Additionally, we aimed to investigate the impact of the order of hero selection on the accuracy of the model's predictions. By combining the CBOW model's ability to capture semantic relationships between heroes with the LSTM model's sequence learning capabilities, we sought to develop a robust predictive model that could effectively analyze and predict game outcomes based on the heroes chosen by each team.

Our initial hypothesis posited that hero selection plays a substantial role in shaping the outcome of Dota 2 games. Through our comprehensive analysis using a chi-square test and baseline and various machine learning algorithms, we were able to confirm this hypothesis. The results demonstrated a statistically significant relationship between hero selection and game outcomes. Furthermore, our hypothesis regarding the significance of the order of hero selection was also supported by the findings. Specifically, our LSTM model outperformed other methods, indicating that the order of hero selection indeed contributes to the accuracy of outcome predictions. Thus, our research successfully aligned with and substantiated our original hypotheses regarding the influential role of the drafting phase and the impact of selection order on game outcomes.

When examining our findings in relation to existing literature, it is important to acknowledge that certain studies have achieved superior accuracy in predicting Dota 2 outcomes. However, these studies concentrated on different facets of the game, such as team fights or team standings after 20 minutes of gameplay. These aspects offer a greater abundance of data and may be comparatively easier to predict. In contrast, our research focused specifically on the hero selection phase, which presents distinct challenges due to the scarcity of available data and the intricate decision-making process involved. Despite these challenges, our study successfully addressed the complexities of hero selection and achieved notable accuracy. Thus, our research contributes to the existing body of literature by emphasizing the significance of analyzing the hero selection phase and showcasing the potential of our approach within this specific context.

We focused exclusively on high-level games lasting a minimum of 20 minutes. This approach may overlook games that end prematurely due to various reasons, such as players going offline or server errors. Additionally, we did not incorporate the individual skills of players into our model. While these limitations are challenging to address directly through the model, it is worth noting that accounting for the proficiency of players with their chosen heroes could potentially enhance the performance of our model significantly. This does not mean that after such an update our model will become unusable, but it is certain that over time a version will become obsolete and needs to be updated.
By demonstrating the influence of hero selection on game outcomes, our study provides valuable insights for players and teams strategizing their draft phase. Understanding the impact of hero choices can help players make more informed decisions and potentially gain a competitive edge. This predictive capability can be leveraged by teams and players to anticipate the outcome of a game based on the heroes chosen by both teams. Such insights can inform draft strategies, allowing teams to optimize their chances of victory. Additionally, our research highlights the complexity of Dota 2 and the need for further exploration in understanding the various stages and factors influencing game outcomes.

However, there is still much to be explored in this area of research. In this section, we will discuss some potential future plans and directions for this project:

- **Further analysis of the game data:** While our research focused on win prediction using a specific model, there are many other variables and factors within the game data that could be analyzed. For example, we could look at the impact of certain heroes or player roles on win rates.

- **Low skill level:** Lower-level matches could also be investigated. This would probably greatly spoil our current model because at a lower level the players do not have much experience yet and thus the outcome of a game is often random. But a model specifically trained for such games might work well.

- **Individual performance:** The historical data of the players with the selected hero could also be included in the model and this information could be taken into account with some weighting in order to determine the final result. As a first assumption, this would help in more accurate forecasting.

- **Extension to other esports or competitive games:** While our research focused specifically on Dota 2, the principles and methods could be applied to other esports or competitive games. Future research could explore the potential for win prediction models in other games, or compare the effectiveness of models across different games.
6 References


