Proceedings, The Thirteenth AAAI Conference on
Artificial Intelligence and Interactive Digital Entertainment (AIIDE-17)

A Recommender System for Hero Line-Ups in MOBA Games

Lucas Hanke, Luiz Chaimowicz
Computer Science Department
Universidade Federal de Minas Gerais

Abstract

Multiplayer Online Battle Arena (MOBA) games are currently one the most popular online game genres. In their basic gameplay, two teams of multiple players compete against each other to destroy the enemy's base, controlling a powerful unit known as "hero". Each hero has different abilities, roles and strengths. Thus, choosing a good combination of heroes is fundamental for the success in the game. In this paper we propose a recommendation system for selecting heroes in a MOBA game. We develop a mechanism based on association rules that suggests the more suitable heroes for composing a team, using data collected from a large number of DOTA 2 matches. For evaluating the efficacy of the line-up, we trained a neural network capable of predicting the winner team with a 88.63% accuracy. The results of the recommendation system were very satisfactory with up to 74.9% success rate.

1 Introduction

MOBA Games have become one of the most played genres in recent years. Games such as *League of Legends* (LOL) or *Defense of the Ancients 2* (DOTA 2) have attracted millions of online players and also become important platforms for e-sports tournaments, which distribute millions of dollars in prizes. In its basic gameplay, two teams of five players compete against each other to destroy the enemy's base. Each player controls a powerful unit known as "hero" or "champion", which is responsible for defeating enemy's armies and defensive structures and, acting together, make the team advance in the game.

Each of these games has more than one hundred heroes that can be picked by players, each one with different abilities, roles and strengths. Thus, choosing a good combination of heroes is fundamental for the success in the game. The combination of heroes in a team is generally called line-up and the number of possible combinations surpasses 10^{16} in a game such as DOTA 2.

In this paper we present a recommendation system for picking heroes in a MOBA game. By collecting data from thousands of DOTA 2 matches, we were able to develop a mechanism based on association rules that indicates the more adequate heroes for composing a team line-up against another team. For evaluating the efficacy of the line-up, we trained a neural network that, given two teams, is capable of predicting the winner team with a good accuracy.

This paper is organized as follows: in the next section we discuss some related work in the area. Section 3 gives an overview of MOBA games and discusses the importance of building a good line-up. In Section 4, we present our methodology, describing the data collecting mechanism, the recommendation system, and the neural network for victory prediction. Finally, Section 5 presents the experimental results and Section 6 brings the conclusion.

2 Related Work

Besides its huge success in the game market, MOBA games have also started to attract interest from the AI academic community as a research platform. Diverse topics such as Dynamic Difficult Adjustment (Silva, do Nascimento Silva, and Chaimowicz 2017), Combat Analysis (Yang, Harrison, and Roberts 2014) and Classification of player roles (Eggert et al. 2015) among others have been explored.

But there are some works more closely related to this research. Conley and Perry (2013) have developed a recommendation engine that, using real data from DOTA 2 matches, computes the probability of victory in a match between two specific teams using logistic regression and clustering algorithms[1]. Based on the computed probabilities, they use a greedy algorithm for recommending line-ups. Despite proposing an interesting methodology, their main focus is on the machine learning algorithms for win prediction and the specific results of the recommendation system are not clearly reported or analyzed. Other similar works have also used machine learning algorithms for predicting the winning team in MOBA matches, for example (Johansson and Wikström 2015) and (Kinkade and Kevin 2015).

Differently from prior work, in this paper we investigate the use of bundle recommendation (Zhu et al. 2014), commonly used in e-commerce, to create line-ups in MOBA games. Generally, in e-commerce, the relevance or attractiveness of an item may strongly depend on other items shown or bought by an user. Zhu et al. argue that the entire set of recommended items should be considered as a

[1] To the best of our knowledge, this recommendation engine was developed as an assignment in a machine learning class and has not been formally published.

bundle rather than treating them individually and independently. Examples of bundles are laptop and accessories, or movies by the same director. This idea is supported by two facts:

- **Clients tend to buy in bundles**: it has been reported that the average shopping cart size for *Amazon.com* is 1.5 and 2.3 for *Walmart.com*.
- **1 + 1 > 2**: there are several marketing studies which report that there is an advantage feeling given to the costumer when it is offered carefully selected product bundles.

Thus, we take these ideas and apply to hero selection in MOBA games. Since they are team-based, the synergy of two or more heroes can provide their teams a considerable advantage during the match. As will be discussed in the next sections, for this we use the Apriori algorithm (Agrawal and Srikant 1994), which was originally developed for learning association rules in large databases and has been recently used in recommendation systems (Saxena and Gaur 2015).

3 MOBA Games

MOBA games, a variant of the *Real Time Strategy* (RTS) games, have become one of the most played genres in recent years. The term MOBA can be broadly used for any game in which multi-player teams battle against each other on a map or arena in order to achieve victory. Usually this goal is achieved by eliminating the enemy base. Although the definition is quite broad, MOBA games are usually composed by two teams (in DOTA 2, they are named *radiant* and *dire* teams) with five players each, in which each player controls a single character, often called a hero. Games take place in real-time and, as with all team-based sports, the game is highly dynamic, making each game unique. Examples of highly popular MOBAs today are LOL, DOTA 2 and *Heroes of the Storm* (HOTS).

MOBAs are complex games that feature hundreds or thousands of possibilities. Each hero has an individual set of abilities that can be evolved and explored in countless different ways during the match by gaining experience (often referred to as XP) or gold. In order to win, each team must coordinate its actions and react to the opposing team's actions in an efficient and effective manner.

The meta-game of MOBAs is changed from time to time by their developers through new patches. A patch is a new version of the game in which the abilities and characteristics of some heroes or items can be changed. Generally speaking, the meta-game can be defined as the most "efficient" way to play that game patch, which is chosen informally and emergently by the community of players. The meta-game can also be altered by the behavior of the players. For example, a player "discovers" a certain feature of the game, and begins exploring it. If it succeeds, other players will start doing the same. Finally, MOBAs have a wide spectrum of possible behaviors. In this sense, mastering these games can be quite challenging. Figure 1 shows the interface of DOTA 2 from a player's perspective. Note that at the top of the screen, the heroes chosen by each team are shown.

In MOBA games, since the available heroes as well as their abilities are quite diverse, much of the strategy involves

Figure 1: Screenshot of the in-game interface of DOTA 2. The top panel shows the heroes picked by each team.

the choice of a good line-up. But the number of different line-ups in a MOBA game is generally very large. For example, by April 9, 2017, the number of heroes available in DOTA 2 was 113. Thus, the number of possible line-ups would be given by:

$$L = C(113, 5) \times C(108, 5) = 1.56 \times 10^{16}. \quad (1)$$

The choice of one hero must take into account its synergy with its allies and the advantage that it has over its opponents. In addition, there are roles that certain heroes fulfill and that should not be forgotten during the picking phase. So, given that two teams have the same skill level, the line-up can be decisive enough to give one team an advantage even before the match starts.

4 Methodology

Our basic methodology consists in collecting data from real DOTA 2 matches using an API provided by the game creators (Steam), and using these data to (i) develop a hero recommendation system based on association rules and (ii) create a neural network to predict the results of a match between two given teams. We created a web portal called houseofdota.com through which the system can be accessed and several statistics about the training can be viewed. In the next sections we detail these modules.

4.1 Data Collection

Steam exposes an HTTP based Web API which can be used to access many features. This API contains public methods that can be accessed from any application capable of making an HTTP request, such as a game client or server (Steam 2017). Using Steam's API for DOTA 2, we are able to collect data related to a large number of matches. These data include which heroes were picked by each team, the winning team, the pick mode, date and time, information regarding the players' connection during the match, among others.

Using this API, we have developed a collector that fetches the information from a maximum of 500 matches every 5 minutes. From these matches, we select only those that meet some requirements:

- **Very high skill players**: We want to guarantee that the players of those matches have a considerable knowledge of the game and their chosen heroes.
- **No disconnects**: Matches can have some players disconnected during its course. These matches are excluded since their results can be biased by this fact.
- **Pick modes with all heroes available**: Since we want to analyze any possible association between heroes, we only use matches that have all heroes available for pick.

Unfortunately, DOTA 2 API does not provide the information regarding the match's patch. Since the metagame can change considerably with a patch update, we want to record the patch of the collected matches. So we developed a crawler that extracts information about the patches from the webpage http://dota2.gamepedia.com/Game_Versions and records it in the database. This way, when a match is collected, we are able to determine its patch knowing the date it was played.

Through this methodology, we were able to construct a dataset in which each match has 10 heroes distributed between radiant and dire teams and we have the information of the winning team, as shown in Figure 2.

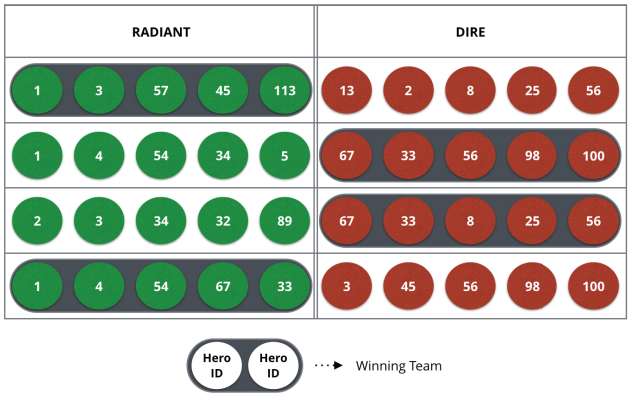

Figure 2: Representation of the matches dataset that we collected from the APIs: each match has 10 heroes distributed between radiant and dire teams and we have the information of the winning team.

4.2 Recommendation System for Hero Lineups

As mentioned in previous sections, we use association rules to recommend the heroes for a particular line-up in DOTA 2. Association rules are commonly used to identify frequent sets of items from large amounts of data. The form of an association rule is $I \Rightarrow j$, where I is a set of items and j is an item. The implication of this association rule is that if all of the items in I appear in some set K, then j is "likely" to appear in that set as well (Saxena and Gaur 2015).

The proposed recommendation system uses a set of association rules between heroes extracted using the Apriori Algorithm. Apriori uses a "bottom up" approach, through which frequent subsets are extended one item at a time. The algorithm terminates when no further successful extensions

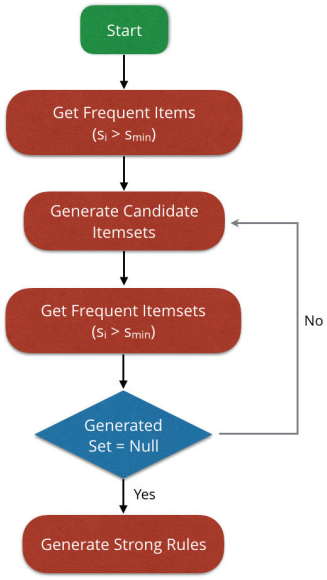

Figure 3: A simplified overview of the Apriori Algorithm: the algorithm uses a "bottom up" approach, where frequent subsets are extended one item at a time. The algorithm terminates when no further successful extensions are found.

are found. A simplified overview of this algorithm is shown in Figure 3. The whole point of the algorithm (and data mining, in general) is to extract useful information from large amounts of data. The algorithm aims to find the rules which satisfy both a minimum support threshold and a minimum confidence threshold (Saxena and Gaur 2015). The "support" of the rule $S(I \Rightarrow j)$ is the fraction of observations in the union of items I and j in the complete item set K from which they were derived. The "confidence" $C(I \Rightarrow j)$ of the rule is its support divided by the support of the antecedent I (Hastie, Tibshirani, and Friedman 2009):

$$C(I \Rightarrow j) = \frac{S(I \Rightarrow j)}{S(I)}, \tag{2}$$

In order to extract these association rules, we make some preprocessing of the dataset and stored in the database. Since we want to recommend heroes both based on allies and enemies, we extract two sets of association rules: one for the associations between heroes that won together and one for the heroes that won against a particular hero.

For the first set, we provide only the information of winning teams to the algorithm and extract any association with a maximum size of 5 (number of heroes in a team). Since the number of possible line-ups are considerably large, we use a relatively small minimum support of $S_{min} = 0.01\%$ for the Apriori Algorithm. Thus, for the association to be considered, it needs to be present on the matches set K in a proportion of $1/10000$. This minimum value of support is justified when we analyze the extracted association rules. For the patch 7.06d, we have a total of 49925 rules. Most part of these (29786) have a value of support between 0.01% and 0.02%. This data is shown on Table 1.

Table 1: Number of association rules for patch 7.06d categorized by the range of their value of support.

Support Range	Number of Extracted Association Rules
$10.0\% < s$	8
$1.00\% < s < 10.0\%$	146
$0.10\% < s < 1.00\%$	3096
$0.05\% < s < 0.10\%$	2573
$0.02\% < s < 0.05\%$	14316
$0.01\% < s < 0.02\%$	29786
Total	49925

For the second set, we provide the information of both teams to the algorithm and extract 2-sized associations between heroes and their opponents. We use the same value for minimum support ($S_{min} = 0.01\%$).

Once we have those association rules extracted, we can recommend heroes based on previous picks. We propose two different 5-sized sets of recommended heroes: one based on allies and the other based on enemies.

- **Based on Allies**: given the set A, with size s, being the already selected allies, we search for association rules R of size $\geq s+1$ that contain any combination A' of heroes in A. A' can be considered the power ser of A excluding the empty set. All rules in R would be composed by $A' \Rightarrow r$, being r a possible hero recommendation for the user. Knowing R, we sort those rules on descending order based on some allies metric (for example, support) and recommend the first 5 r's.
- **Based on Enemies**: given the set E being the already selected enemies, we search for 2-sized association rules R that contain one (any) hero e of E. All rules in R would be composed by $-e \Rightarrow r$, being r a possible hero recommendation for the user. The symbol $-e$ is used only to represent that the association rule represents a counter relation (r is a good choice against an e opponent). Knowing R, we sort those rules on descending order based on some counter metric and recommend the first 5 r's.

In Section 5, we detail the metrics (allies and counters) used to select the recommended bundles.

4.3 Neural Network for Game Prediction

Given that the main goal of a recommendation system for hero selection is to help the player to build a line-up that will lead him/her to victory, one way to evaluate a recommendation is to check if its result has a greater chance of winning the match. Taking this into consideration, we want to have some sort of match result prediction in place.

In the case of this paper, since the results of the matches in the database are known, a supervised learning technique is used to develop the prediction model. The inputs are the picked heroes (for each team) and the output is the match result (which team won), as shown in Figure 4.

Following the ideas from (Huang and Chang 2010), the adopted prediction model is based on a multi-layer perceptron (MLP) with back propagation. The input layer has one neuron for each possible hero. The input value for a particular $hero_i$ neuron is given by the Equation 3.

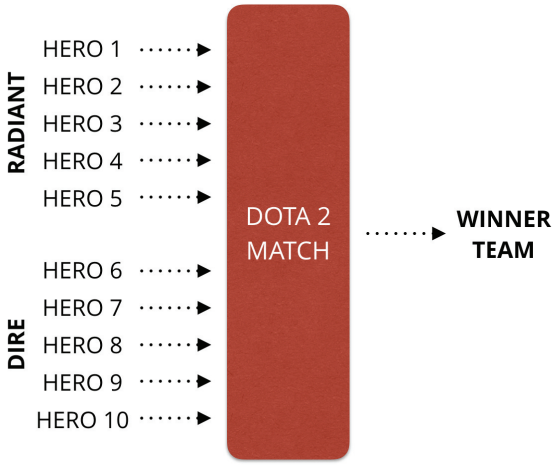

Figure 4: Representation of the supervised learning used for prediction in this paper: the heroes are the inputs and the match result is the output.

$$I_i = \begin{cases} 1 & \text{if } hero_i \in \text{radiant} \\ -1 & \text{if } hero_i \in \text{dire} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

The hidden layer is composed by 300 neurons and the output layer has one neuron only which represents the result of the match and has a Sigmoid activation function. For the training, the result of the match is given by the Equation 4. A representation of the adopted neural network is displayed in Figure 5.

$$O = \begin{cases} 1 & \text{if radiant won} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

5 Experiments and Results

In this section we detail the results we obtained from both the neural network based prediction and the recommender system. As mentioned, the system is available for use as a user friendly web application (Figure 6). The user can select the allied heroes and the enemies and, based on that, we recommend five heroes based on allies and five based on enemies (counter picks).

5.1 Performance of Neural Network prediction

In order to keep the modeled neural network up-to-date with the current meta-game, we constantly retrain it with data from recent matches. Every hour, we execute a training (for 100 epochs) with the matches from the current patch. For the training, we use 70% of the pool of matches we have for the current patch. The other 30% we use for testing. As this is an on-going process, the neural network is always being

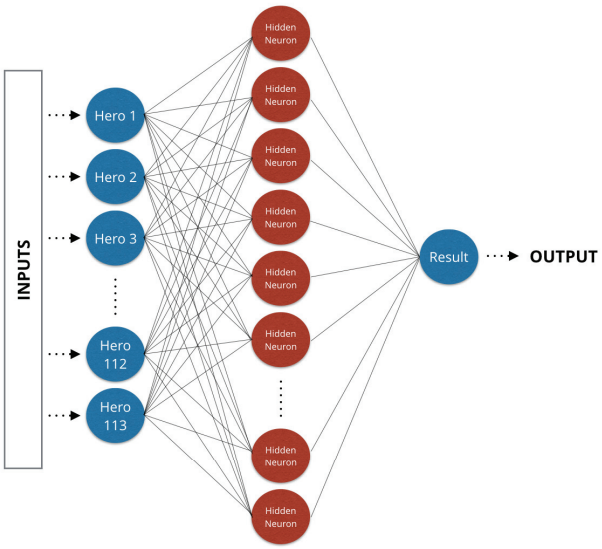

Figure 5: Representation of the adopted prediction model: a multilayer perceptron neural network. The input layer has one neuron per hero, the hidden layer is composed by 300 neurons and the output layer has one neuron only and is computed according to a sigmoid function.

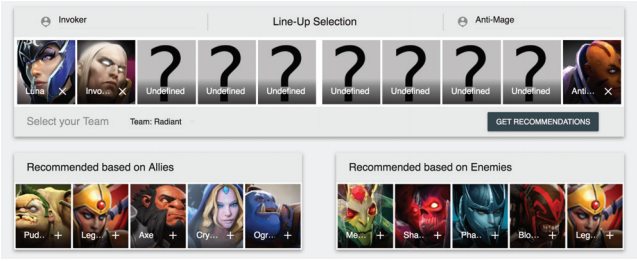

Figure 6: Screenshot of the system's interface. The user has already informed two heroes as allies and one as enemy. Then, the system recommends five heroes based on allies and five based on enemies (counter picks).

retrained for the current meta-game and patch. For example, when the new patch 7.06d was detected, the neural network was reset and a new training process was started with matches from that patch, as shown in Figure 7. After 193 training rounds (each one with 100 epochs), around 54000 matches being used for training and 23000 for testing, the accuracy of the neural network prediction was 89.28% for training and 88.63% for testing. We also show a baseline that represents the actual win rate for the radiant team. That team has a statistical advantage over the dire team on DOTA 2 matches. We used it as baseline due to the fact that if we just choose to predict that the radiant team always win, we would have not a 50% chance of getting it right, but around 63% as shown in the graph.

5.2 Results for Recommender System

Before getting into the experiments executed to test the performance of the recommender system, it is important to detail the process of picking out heroes in a DOTA 2 match. Initially, a team is randomly selected to start picking. From that point on, each team takes turns in selecting a hero from the available pool until the line-up is complete (each team has 5 selected heroes). Thus, we developed an experiment to simulate the same logic where:

- Teams take turns in selecting heroes from the available pool;
- Enemy heroes are selected following one of these approaches: a) randomly among all available heroes, b) randomly among the 10 most picked heroes of the patch;
- Allied heroes are selected using the recommender: we get the recommendations based on allies and enemies (5 heroes from each) and select randomly from that pool;
- With the full line-up, we input that information into the trained neural network to predict which team has a greater chance of winning.

The logic implemented by the experiment is shown in Algorithm 1. The recommendations are selected based on the parameters $allies\_metric$ and $counters\_metric$, which will be described below.

Algorithm 1 Algorithm for an unique experiment adopting allies_metric and counters_metric parameters to select the recommendations. Enemies are selected randomly among all available heroes or among the 10 most picked heroes of the patch. At the end, the neural network is used to predict which team has more chance of winning the match based on the constructed line-up.

function EXPERIMENT($allies\_metric, counters\_metric$)
  $A, E \leftarrow []$
  while A.size $\neq 5$ and E.size $\neq 5$ **do**
    E.insert(random enemy)
    $R \leftarrow []$
    R.insert($get\_recommendations(allies\_metric)$)
    R.insert($get\_counters(counters\_metric)$)
    A.insert(pick randomly from R)
  end while
  $result \leftarrow get\_nn\_prediction(A, E)$ **return** $result$
end function

In order to improve the recommender system and check in which scenario it would perform best, we used two different types of metrics for each parameter of the experiment. For the recommendation based on allies A (allies metric), we used the support S of the association rule $A \Rightarrow r$ extracted from winning teams data (shown in Equation 5). For instance, suppose that x and y won together one time among 3 played matches. Hence, the support for this association is 33.33%.

The other metric for recommending based on allies was the win rate of the hero bundle $A \cup r$, which can be calculated as the support S of the association rule $A \Rightarrow r$ extracted from winning teams data over that same value plus the support S of that same rule extracted from the loosing teams

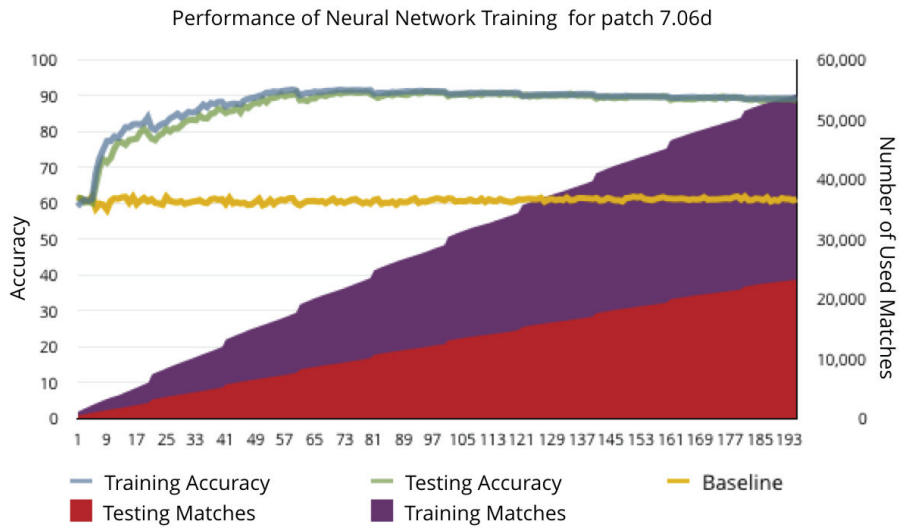

Figure 7: Accuracy for match result prediction using a Neural Network. 70% of the matches were used for training and 30% for testing. The baseline represents the win rate for the radiant team which has a statistical advantage over the dire team.

data (shown in Equation 6). For the same example given above, suppose that, despite winning just once, the heroes x and y were in the same team just once as well. So, the win rate for this association would be 100.00%.

$$support(A, r) = S(A' \Rightarrow r) \tag{5}$$

$$win\_rate(A, r) = \frac{S(A' \Rightarrow r)}{S(A' \Rightarrow r) + S(-A' \Rightarrow -r)} \tag{6}$$

For the recommendation based on enemies E, we used the confidence C' of association rules $-e \Rightarrow r$ extracted from all teams data (shown in Equation 7). For instance, suppose that hero x won one match against hero y, but hero y was present in all 3 played matches and lost them all. Thus, the confidence of this association is 33.33%/100%=33.33%. However, if z also lost one match against x and that was the only match that z lost, the confidence of that association would be 33.33%/33.33%=100.00%.

The other metric for recommending based on enemies was named counter coefficient and can be computed as the support S of the association rule $-e \Rightarrow r$ over that same value of support plus the support S of the opposite association rule $-r \Rightarrow e$ (shown in Equation 8). This metric tries to eliminate cases where the relation between e and r exists as opponents, but we do not know who is the counter of whom (the support for both $-e \Rightarrow r$ and $-r \Rightarrow e$ are high).

$$confidence(e, r) = C(-e \Rightarrow r) = \frac{S(-e \Rightarrow r)}{S(-e)} \tag{7}$$

$$counter\_coefficient(e, r) = \frac{S(-e \Rightarrow r)}{S(-e \Rightarrow r) + S(-r \Rightarrow e)} \tag{8}$$

For each combination of metrics we executed the experiment, represented in Algorithm 1, 1000 times and verified the winning rate of the team that used the developed recommender system. The results are shown in Table 2.

Table 2: Results for experiments using the recommender system against a team chosen randomly among all heroes. The numbers represent the winning rates (with match results predicted by the neural network) for the team built using the combination of the Allies and Counters Metrics.

		Counters Criteria	
		Counter coefficient	Confidence
Allies Criteria	Support	56.9%	60.5%
	Win rate	73.4%	**76.4%**

Analyzing the results, we can conclude that the worst combination of metrics is support for allies recommendation and counter coefficient for counters recommendation with 56.9% successful rate. That is, the team using the recommender system won only 56.9% of the simulated matches according to the neural network prediction. However, using the combination of metrics of win rate for allies recommendation and counter confidence for counters recommendation, we achieved a 76.4% success rate.

We executed another set of experiments picking the enemy heroes randomly among the 10 most picked heroes from patch 7.06d. The results are shown in Table 3.

The results from these experiments are similar to the ones with the randomly chosen enemies: the worst combination of metrics is support for allies recommendation and counter coefficient for counters recommendation with 59.9% successful rate. And, again, using the combination of metrics of

Table 3: Results for experiments using the recommender system against a team chosen amongst the most picked heroes from patch 7.06d.

		Counters Criteria	
		Counter coefficient	Confidence
Allies Criteria	Support	59.9%	60.2%
	Win rate	71.7%	**74.9%**

win rate for allies recommendation and counter confidence for counters recommendation, we achieved a 74.9% success rate.

In both experiments we can observe that changing the allies metric has a greater impact in the percentage of victories of the recommended team. Using the win rate metric, we probably got better results because it focuses on heroes that have a high percentage of victories. On the other hand, the variation of the counters metric does not have a great impact. One possible cause is that the counters metric uses only sets of size two: the ally hero that win against one specific enemy, while the allies metric uses the power set of allies to generate the association rules.

6 Conclusion

In this paper we presented a recommendation system for picking heroes in DOTA 2. By collecting data from thousands of DOTA 2 matches, we developed a mechanism based on association rules that suggests the more adequate heroes for composing a team line-up against another team. For evaluating the efficacy of the line-up, we trained a neural network that, given the line-up for two teams, was capable of predicting the winner team with a good accuracy of up to 88.63%.

We executed multiple experiments to determine which metric had the largest win rate with the neural network. The experiments simulated a real hero picking phase for DOTA 2: teams take turns in picking their heroes, in which allies were chosen using the recommendation system and enemies were picked randomly (among all heroes or among the 10 most picked of the patch). Using the parameters of win rate to recommend based on allies and the counter confidence to recommend based on enemies, we reached a 74.9% percentage of vitories in 1000 experiments.

As future work, we intend to perform more experiments and analysis with the system. We want to evaluate other metrics and also investigate other mechanisms for improving the obtained results. Moreover, it would be interesting to perform some sort of qualitative validation with real users. For example, a set of real players could be asked to use the system during the picking phase of some matches. And, at the end, could fill out a form that could help evaluating the user's satisfaction and quality of the recommender system.

Acknowledgments

The authors would like to thank Prof. Rodrygo Luis Teodoro Santos for some insights on recommendation systems. This work was partially supported by Fapemig and CNPq.

References

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, 487–499. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Conley, K., and Perry, D. 2013. How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2. http://cs229.stanford.edu/proj2013/PerryConley-HowDoesHeSawMeARecommendationEngineForPickingHeroesInDota2.pdf. Course Project. Stanford University. [accessed on 08/01/17].

Eggert, C.; Herrlich, M.; Smeddinck, J.; and Malaka, R. 2015. Classification of player roles in the team-based multi-player game dota 2. In *Entertainment Computing - ICEC 2015: 14th International Conference, ICEC 2015, Trondheim, Norway, September 29 - Ocotober 2, 2015, Proceedings*, 112–125. Springer International Publishing.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2009. *The Elements of Statistical Learning*. Springer, second edition.

Huang, K.-Y., and Chang, W.-L. 2010. A neural network method for prediction of 2006 World Cup Football Game. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Johansson, F., and Wikström, J. 2015. Result prediction by mining replays in dota 2. Master's thesis, Faculty of Computing, Blekinge Institute of Technology,.

Kinkade, N., and Kevin, L. 2015. Dota 2 win prediction. https://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/018.pdf. Course Project. University of California San Diego. [accessed on 08/01/17].

Saxena, A., and Gaur, N. K. 2015. Frequent Item Set Based Recommendation using Apriori. *International Journal of Science, Engineering and Technology Research* 4(5).

Silva, M. P.; do Nascimento Silva, V.; and Chaimowicz, L. 2017. Dynamic difficulty adjustment on MOBA games. *Entertainment Computing* 18:103 – 123.

Steam. 2017. Web api (steamworks documentation). https://partner.steamgames.com/documentation/webapi. [accessed on 08/01/17].

Yang, P.; Harrison, B. E.; and Roberts, D. L. 2014. Identifying patterns in combat that are predictive of success in MOBA games. In *Proceedings of the 9th International Conference on the Foundations of Digital Games, FDG 2014*.

Zhu, T.; Harrington, P.; Li, J.; and Tang, L. 2014. Bundle recommendation in ecommerce. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval - SIGIR '14*, 657–666. New York, New York, USA: ACM Press.