DOTA2 Game Result Prediction and Hero Suggestion System

A System for Result Prediction and Hero Suggestion for DOTA2 game using Artificial Neural Networks

Abhishek Kumar, B.Tech, Mahesh Mishra, B.Tech, and Rahul Verma, B.Tech.

Department of Computer Science and Engineering, Indian Institute of Technology, Banaras Hindu University, Varanasi

(abhishek.kumar.cse12, mahesh.mishra.cse12, rahul.verma.cse12)@iitbhu.ac.in

**Abstract.** In this paper, we present a result prediction engine for the popular MOBA game DOTA2 detailing the previous attempts at match result prediction and exploring the machine learning approaches applied to this problem.

First, we provide details of the machine learning algorithms applied to DOTA2 and in the process gain further insights into the process we use. Second, we present a model that can be used for hero suggestion and match result prediction in both casual and competitive sides of DOTA2’s massive 15+ million player base.

In the development of this model, we use Artificial Neural Network on dataset related to DOTA2 gameplay. We use back-propagation and NEAT algorithms to train this model which is used for game result prediction and hero suggestion at each level of selection.

**KEYWORDS:** DOTA2, Artificial Neural Network, Back-Propagation, NEAT Algorithm, Neuroph, ENCOG

1. Introduction

DOTA2 is a multiplayer online battle arena (MOBA) game with a constantly increasing unique player base of more than 11 million players [1].

Each DOTA2 match consists of two teams of five players each pitted against each other. Before a match begins, each team selects characters to play as, known as a “hero,” from a pool of 110 different heroes [2], which results in over a quadrillion (10^24) possible team combinations. The goal of our work is to correctly predict the outcome of the match based on the drafting phase alone and provide a winning strategy [3], using Artificial Neural Networks. We adopt different learning algorithms viz. back-propagation and NEAT algorithm and present the results obtained, in this paper.

Objectives of our work:

* We predict the result of a match based on any given matchup.
* With 110 heroes to choose from and five heroes per team, we present an approach to find the best three heroes at each level of selection for any given matchup.
* We propose a real-time winner predictor model to give live result prediction in-game.

In our paper, we first discuss the related literature which covers the details about the game and the existing approaches for predicting the results of the games. In subsequent sections, we discuss the methodology used in our approach. These sections cover a brief introduction to Artificial Neural Networks and the algorithms used to train this network. Later, we discuss the application of this trained model to achieve the aforementioned objectives. Finally, we conclude the paper by presenting an analysis of results obtained by different training mechanisms to train the neural network and suggest the possibility of integration of this model in the DOTA2 game itself.

* 1. **Background**

The topic under study has pretty narrow spectrum of previous works. Following are the most significant ones:

* **Zero sum game model:** Dota2cp [4] is an application which recommends heroes and predicts results with a self-reported accuracy of 64% by the author to predict the winning team. The author modeled the draft as a zero-sum-game. Logistic regression was then applied to learn the game matrix. The teams are considered min-max agents that pick heroes one at a time. This was the first approach for recommendation engine however, newer techniques have given better results. Logistic regression formulates the relation between the features and results as a logarithmic function. The values of the constants in the function estimated using probabilities using logistic functions and probit regression (since the choice on a hero is binary in nature).
* **Feature Vector Model:** The authors had an accuracy [5] of 69.8%. They implemented feature vector for heroes (each vector element representing a hero). The coefficients were learned using logistic regression. Furthermore, they also applied kNN classification with lesser success. Feature vectors were utilized by considering the whole pool of heroes and their draft result (pick/ban) as a vector. The output was a winning chance (percentage) which was computed as a linear combination of the vector elements. Logistic regression is applied to calculate the values of the coefficients to each vector element. The kNN approach was applied similarly by taking the feature vectors and comparing the data with previously learned data to find the closest neighbors. The output was predicted as a combination of the k nearest draft results.

1. Specification of the Project

DOTA2 is a Multiplayer Online Battle Arena (MOBA) video game owned by Valve Corporation [6]. It supports Linux, OS X and Microsoft Windows from July 2013. The game was preceded by Defense of the Ancients (DOTA), a mod game for Warcraft III: Reign of Chaos (owned by Blizzard Entertainments). DOTA2 was the first game to use the Source 2 engine when it was shifted to the new engine in September 2015 [7].

A match of DOTA2 is played between 2 teams of 5 players each. Both teams occupy the opposite corners of the arena. A match is won by destroying the opponent’s base. Players have the option to pick/ban heroes from a pool of 110 heroes (with more heroes coming in each patch). Throughout the game, the player controls his/her hero and manages items, gold and experience and fights the other side while improving the economy of their own team. Along with the player owned structures and heroes, neutral AI’s exist who can be killed to further boost the team’s economy. The game ends when the core structure of a team (called ‘Ancient’) is destroyed [8].

Each hero is unique in the game with well-defined abilities and attribute gains throughout the game. The item selection also plays a major role in the player’s effectiveness. Furthermore, different heroes have the ability to counter other heroes and thus reduce their effectiveness. Similarly, there are synergies among heroes which help in boosting the team’s performance.

An official match starts with a drafting phase where the captains of both the teams decide hero picks for their team and hero bans for everyone. Clearly due to the interaction between the heroes providing an advantage for a team, the drafting phase is considered to be one of the most important phase of a match where complete games can be lost due to poor drafting or a significant advantage can be gained with superior drafting.

Our prediction engine aims to predict the outcome of a game based on this drafting phase alone.

* 1. Use Cases

At The International 2015 tournament, the prize pool exceeded $18.4 million, earning the champion team over $6 million which emphasizes the significance of each game. The Hero Suggestion model can be used in drafting phase itself for a better team composition which enhances one’s chances of winning. The system can also be used to improve the drafting skills of newbies. The result prediction model can be used for in-game betting. Real-time winner predictor can be used to monitor and improve the performance during a game itself.

* 1. Implementation Choices
* **Neuroph**: It is an object-oriented Java framework used to create and train neural networks. Neuroph [9] is a java class library for creating and training neural networks.
* **Java Applets**: A Java applet is an application provided to users in the form of bytecode. Java applets are actually written in Java programming language but applets can be written in any language that compiles its code to Java bytecode.
* **JFreeChart**: JFreeChart is a Java chart library that supports a wide range of charts. It is an open source free software distributed under the GNU Lesser General Public License. It has a flexible design and supports Swing and JavaFX components, different image file formats such as PNG and JPEG and vector image file formats as SVG, EPS and PDF.
* **Beautiful Soup**: It is a third party python library [10] used for extracting text or data from webpages from Crummy. Beautiful Soup is provided with *PyPi* so one should install it with *pip* or *easy\_install* instead of installing with system packager. It parses and converts a webpage into a complex tree of python objects.

**System Specifications and Dataset**

* **System:** Intel-i5 processor (quad-core), 4GB RAM.
* **Software:** Windows 10-Pro, Eclipse-4.2.2, python-2.7, Neuroph framework version 2.8, ENCOG framework version 3.0.
* **Datasets:**

Learning data = 90.0% of total data.

Testing data = 10.0% of total data.

Drafting data for the matches were taken from the website (datdota.com).

* 1. Software Architecture

**Step 1: Web Crawlers**

As far as crawlers (web spiders) go, they scour a page for URL's (in our case) and puts them in a new list or stack and continues iterating through each URL found, goes to it, and extracts all URL's present on that page and so on (if it has been coded further) [11]. The logic here is fairly straightforward:

* The user provides the beginning URL.
* Crawler goes to that URL (page), collects all the relevant URL’s present in the source code.
* Each gathered URL is visited by crawler in nested for loop gathering child urls.

Then we parse the source code of pages we passed through returning urls, writing child urls to file, fetching the required data from the tables and storing them in csv files.

**Step 2: Pre-processing**

In this section, we describe the algorithms used by us in the pre – processing phase of our prediction engine.

**Features:** In DOTA2, 110 heroes are currently in the game. For each of the heroes, we give an ID from 0 to 109. Then we define values for each of the heroes in our processed data such that for ith hero,

Xi = 1, if the hero is picked,

= 0.33, if the hero is banned,

= 0, otherwise.

A positive score is given for the team picking first and negative for the team picking second. Using the above pre - calculating step, we generate the matrix of pre – processed values corresponding to each hero for different matches. These values are stored in a form of comma separated values (csv files).

**Step 3A: Modeling**

In this section, we describe the core section of our prediction engine. We use the Neuroph framework to implement our algorithms in this part.

* **Neural Network**: A **multilayer perceptron** is used with the heroes featuring as 110 input nodes, 1 output node and 50 hidden nodes. A multilayer perceptron is feed-forward artificial neural network model having some set of inputs onto some set of outputs. It consists of multiple layers of nodes creating a directed graph, where every node in each layer is connected to every node in the next layer using some pre-defined activation function [12].
* Hyperbolic activation function (*tanh*) was used for the neurons. The activation function of a node gives the output of that node for provided set of inputs or single input. A computer chip can be treated as a network (digital) of activation functions that can take only two values 0 and 1 or “OFF” and “ON” for given set of inputs.
* **Learning algorithm:** Back-propagation algorithm is used for training or learning the data set by adjusting the weight of neurons. Back-propagation algorithm is also called backward propagation of errors. It is used for training the artificial neural network [13] generally used with gradient descent for optimization purpose. It calculates loss function gradient corresponding to weights of the directed artificial neural network. The optimization method takes this gradient and updates the weights of the network, minimizing loss function.

Back-propagation [14] calculates loss function gradient using known output for each input value. Therefore, it comes under supervised learning method. It needs an activation function which is used by neurons. This activation function needs to be differentiable.

**Neural Network**

* Pre-processing -> Data1, Data2, … , Data20.
* Datai = Data containing pick or ban of a hero at *i* times.
* Datai -> Neural Network -> Modeli.

**Suggestion**

* Modelj suggests hero at position j.
* If having the pick or ban information of j-1 heroes, can predict win rate from ANN (learning from Dataj).

**Step 3B: Modeling**

In this section, we describe the application of the ENCOG framework for the implementation of the NEAT algorithm in our model. An initial Neural Network population is created by 110 input nodes and 1 output node. As the NEAT algorithm works, the network gradually evolves to form a structure which fits the data.

* **Neural Network Representation:** The neural networks are represented as graphs where each network is an individual. Each individual is represented by a list of node genes and connection genes. Node genes contain information of the node being input, hidden or output. Connection genes contain information of the nodes being connected by the link, weight of the link, status of the link (enabled or disabled) and an innovation number which denotes its evolution order. The innovation number is used to limit redundant mutations. Furthermore, speciation is applied to enhance mutation.
* **Learning algorithm (NEAT):** NEAT, which stands for Neuro-Evolution of Augmenting Topologies [15], is a technique for evolving the neural networks using concept of genetic algorithm viz. crossover and mutation. The algorithm works by initializing the network with a minimum number of input and output nodes. Each network is an individual with link genes and node genes. For the next generation, speciation is first applied to protect new individuals. Explicit fitness sharing is used to prevent a species from taking over the entire population. Mean square error is taken as the fitness function in generating output for given data. The genetic algorithm is gradually run until the required fitness function value is reached or the result stagnates.

1. **Phase I:** Predict the results of matches using the trained neural network. The main algorithm is divided into 4 steps:

* **Step 1:** Process the data from different matches and merge them.
* **Step 2:** Create a neural network using Neuroph framework in JAVA.
* **Step 3:** Run the network on test data and find the result.
* **Step 4:** Display the obtained results through a user-interface.

1. **Phase II:** Use the trained neural network to suggest heroes at a given level of selection Methodology for the Hero Suggestion System:

* Overall team selection procedure of 10 Heroes takes places in 20 steps with one hero being either picked or banned at each step.
* Using the Java applet, the user is provided with an interface to fill out his choices. At each level, the user can select any hero (without repetition) as his choice for either pick or ban.
* Using these choices, the feature vector is updated and the already trained Artificial Neural Network is then subjected to all the remaining possibilities at the next level.
* Among these possibilities, three heroes with the maximum probability (to be selected as per the prediction by the ANN) is suggested to the user.
* The user may or may not abide by the prediction and fulfill the next choice. Then the steps 3 and 4 are repeated again.

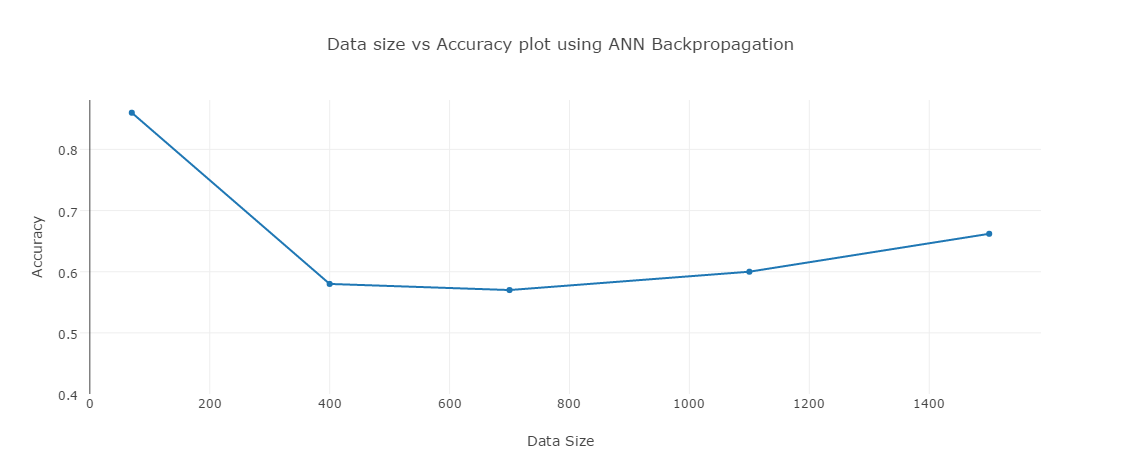
2.4 Test Cases

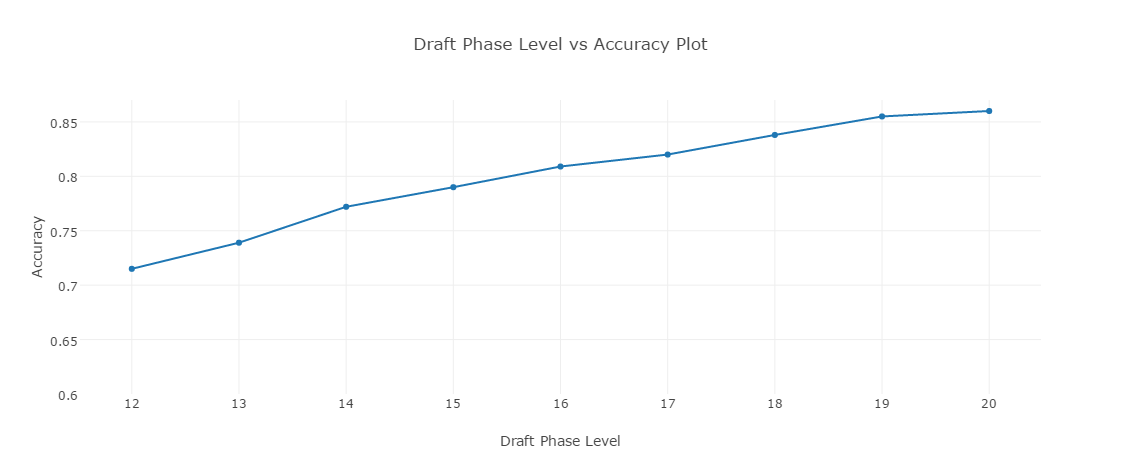
**2.5 Program Code**

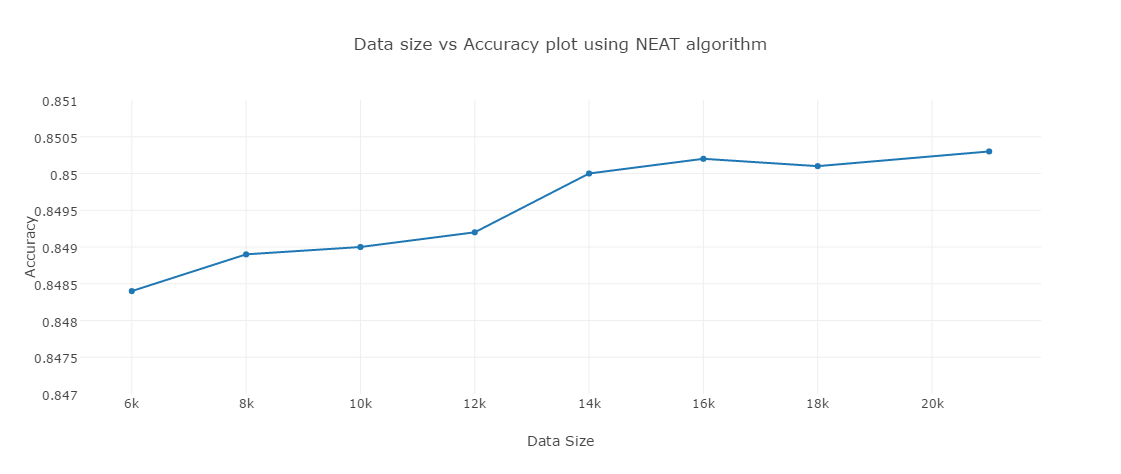
1. Status

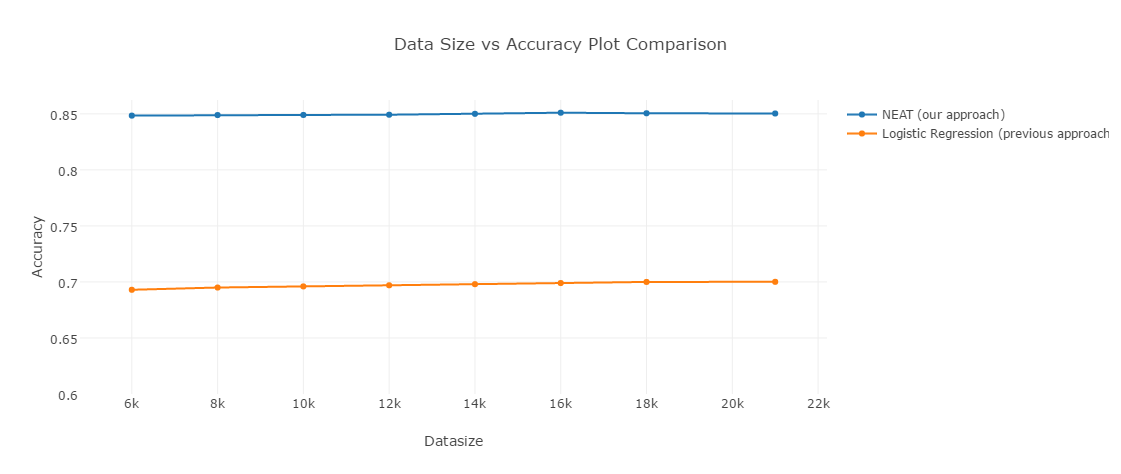
Currently, no work is under progress on the project as the desired results were achieved with very good accuracy. We got promising results with approximately 86% overall accuracy in Phase 1 from the learned network on tournament data and 84% overall accuracy on general data using the NEAT algorithm for training, which is much better than the current methods for prediction applied to this problem. Following graphs represent the results obtained in the project and a comparison with the results obtained by the previous works on the same problem.

**Fig. 1.** Data Size vs Accuracy Plot using ANN Back-propagation



**Fig. 2.** Draft Phase Level vs Accuracy Plot

**Fig. 3.** Data Size vs Accuracy Plot using NEAT algorithm

**Fig. 4.** Data Size vs Accuracy Plot Comparison

1. Conclusion

The developed model gave promising results in game result prediction and hero suggestion with about 84% overall accuracy. However, the real-time prediction system could not be tested against actual game data due to lack of ways to access the data during a game as the software of DOTA2 is not open source. Major issues appeared in terms of data correlation as the data pertaining to different tournaments were not correlated at all. This adversely affected the training of our model. To tackle this, we separately ran our model on tournament data only and got 86% total accuracy. Hence, if the data is correlated well enough, the overall accuracy for result prediction on general data might further improve.

Our initial aim was to develop the model and integrate it with the game as well, but since the game code base is private, we had to restrict ourselves to an independent tool only.

If we were to do the project again, we would try out other approaches to train the neural network. We would also use a larger dataset with data related to closely correlated tournaments only.

1. Future Work

This model can be scaled to handle any number of heroes. We can extend our model not only to predict results but also to predict final scores of both the teams. Our current work is in the form of an external application, but it can be integrated within the game itself using LUA programming language. This model can also be modified to predict results for other games similar to DOTA2.

A new patch for DOTA2 is released generally after a period of few months, but its version didn’t change while we collected the data set for this project. Therefore, our prediction and suggestion engine will work with good accuracy for a given patch of DOTA2, and can produce decent results for the general cases like data collected from multiple patches or versions. So, creating an engine for general case is a tedious task and will take some good research work. We can also use sliding window technique to tackle this problem by resetting data of match history after every new patch release of DOTA2. In this way, data relevancy can be maintained. We can find appropriate match data size for training phase so that we only have to download that much of data to maintain our performance level. We believe that there are other fields as well where some improvement can be done.

1. References
2. Phil Savage, "The International 2015 prize distribution announced", *PC Gamer*. Retrieved July 21, 2015.
3. “The International Interactive Compendium.” DOTA2 Official Blog. Valve. Web. http://www.dota2.com/international/compendium. Retrieved 25 August. 2015.
4. “DOTABUFF - DOTA2 Statistics.” DOTABUFF - DOTA2 Statistics. Elo Entertainment LLC, 2013. Web. 14 Nov. 2015. <http://dotabuff.com/>.
5. ”Dota2 Counter-Pick.” https://bitbucket.org/moldovan/dota2cp. Web. 19 Nov.2014.
6. How Does He Saw Me? A Recommendation Engine for Picking Heroes in DOTA2 By Kevin Conley, Daniel Perry (Stanford University), 2014.
7. Macy, Seth, "DOTA2 Now Valve's First Ever Source 2 Game", IGN. Retrieved 9 September 2015.
8. Kinkade, Nicholas. "DOTA2 Win Prediction.", 2015.
9. “Dota2 Counter-Pick.” Dota2 Counter-Pick. N.p., n.d. Web. 14 Nov. 2013. <http://dota2cp.com/>.
10. http://neuroph.sourceforge.net/download.html, Retrieved, 16 September 2015.
11. https://www.crummy.com/software/BeautifulSoup/bs4/doc/ Retrieved 21 November 2015.
12. “WebAPI.” Official TF2 Wiki. Valve, 16 Sep 2015.
13. Piramuthu, S., Sikora R. T. Iterative feature construction for improving inductive learning algorithms. In Journal of Expert Systems with Applications. Vol. 36 , Iss. 2 (March pp. 3401-3406, 2009.
14. Stuart Russell; Peter Norvig, Artificial Intelligence A Modern Approach. P. 578.
15. Paul J. Werbos, “The Roots of Back-propagation: From Ordered Derivatives to Neural Networks and Political Forecasting”, New York, NY: John Wiley & Sons, Inc.
16. Duan, Hong. "A New Hybridization Method Using in NEAT." Applied Mechanics and Materials. Vol. 543. 2014.