Player Imitation Through Predicting Individual Player's Hero Picks in DoTA2

Abdelrahman Madkour
NUID: 001884122
Northeastern University
Boston ,MA 02115
madkour.a@husky.neu.edu

1 Introduction

Modeling a player's actions has been the subject of much research [24] and has many applications, such as assisting game designers with understanding the implication of the systems they have implemented, creating more human-like AI agents or adjust the game experience to cater to an individual player's preference [14]. There are many different types player models [24] each attempting to capture a different aspects of behaviour modeling with different levels of granularity. In this project we explore a data-driven approach to model a player by imitating their macro-decisions in the popular online MOBA game Defense of The Ancients 2.

Defense of The Ancients 2 (DoTA 2) is a popular Multiplayer Online Battle Arena (MOBA) game where two teams of five, the Radiant team and the Dire team, battle it out to destroy one another's base. Before the beginning of each match players make the biggest macro-decision of the entire game; which hero to pick. Heroes are player controlled characters that have various abilities and characteristics that differentiate them from the rest of the roster and are typically designed to serve a particular purpose in the meta-game. For every hero there are other heroes that compliment them in a team, counter their abilities or do poorly against them in 1 vs 1 fight. Thus every small alteration into what a hero can or cannot do has ramification that ripple through the entire meta-game and change how the game is played. Being able to understand these consequences by predicting what hero players pick in a given situation would give the game designers a very useful tool by which they can gauge the impact of their decisions. DoTA2 is also a big player in the E-Sports space. Its biggest tournament, The International, had a prize pool of over $25 million [38]. And so for competitive teams having a means by which they can predict what heroes their opponents will pick is of massive strategic importance as it paramount to have a good team composition and an advantageous match up.

To that end this projects attempts to tackle these two problems:

- Can we predict what hero will picked for a specific player given the picks and/or bans of other players?
- Can we predict what hero will picked for a specific player given the previous picks/bans?

Player picks are often very unpredictable, as we will later show, due to a variety of latent factors outside of the information given to us, and thus it is a very difficult prediction problem. In fact [35] showed that human experts have an accuracy of around 31.48% when trying to predict just one out of a sequence of 12 hero picks and bans correctly. And these experts were commentators for professional games and therefore were very knowledgeable about the teams who were playing and the current state of the game. In this project we attempt to capture some of this implicit knowledge to solve the two problems proposed above, namely our contributions are:

- A proposed feature representation for the models tackling the first problem.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

- A set non-sequential models using SVM,Random Forest and Feed-Forward Neural Networks for classifying which hero is picked formatted into the proposed representation.
- Sequential models one using Long Short Term Memory Neural Network and the other an N-Grams language model to tackle the second problem.

2 Related Work

There has been a lot of work for player imitation in First Person Shooters [22] [23][32][12][30], puzzle games [3][18][25], racing games [39][4] [26] , fighting games[27] and Real Time Strategy (RTS) games. Case based reasoning has been used on two RTS games WARGUS and StarCraft to predict build and search actions in an attempt to predict an opponent strategy and feed into an AI agent [10] [19] [28]. [9] employed a different approach using HMMs , [6] used random forests and feature expanded decisions trees, [37] a Bayesian network and [34] used an answer set programming approach. Other efforts tried to directly imitate a player, [31] attempting to build behaviour trees based on play traces, however their approach leads to a lot of branching. [20] used a 4 layer neural network to predict unit-build-order for a specific faction in StarCraft against another specific faction. None of these approaches have been effective in imitating a specific player's play-style, they tend to focus more on general player modeling as opposed to a model per individual. There has been unpublished work by [2] who used LSTMs and Random Forests to predict unit-build order for a specific player with high accuracy.

Hero drafts in DoTA2 has been studied before, [13] proposed a method via mining association rules [1] from previous games and build heuristics for recommendation via these rules. The rules being heroes that frequently show up together in winning teams or their opponents. [5] formulate the act of drafting as a combinatorial game and use Monte Carlo Tree Search [7] to find an optimal pick. Other methods [33][41] used machine learning methods to look at the composition of teams that won and those who lost. They did not however use that information to come up with recommendations.These methods all focus on the performance of the teams and trying to optimize the drafts with that in mind. However we are interested in player imitation not performance. The most relevant work to this project is [35] where they train an LSTM [17] on hero picks and bans in tournament DoTA2 games for sequence prediction. Given the first pick/ban the network then attempt to predict the rest of the draft. Their method achieved on average an accuracy of around 11.94%. They also transcribed the commentary of professional games and through those transcriptions found that the commentators were correct in predicting at least one pick/ban of the whole draft at an accuracy of around 31.48%. Our two problems differ to this work in that [35] focus on predicting what is likely to be picked next regardless of who the players are, whilst our approach is explicitly trying to figure out what hero a specific player will pick.

3 Method/Model

Both of the proposed problems we are trying to tackle in this project are posed as classification problems. That is we have a label $y \in 1 \dots 115$ which signifies the class of a match (i.e the hero the player picked) and a feature vector x. More information about the feature representation is given in the next section, here will focus on the types of models we used for the two problems.

3.1 Non-Sequential Models

For this problem we opted to try a Linear SVM, a Random Forest and 2 layer deep neural network. See [36] and [16] for an overview of SVM and Random Forest respectively. We will briefly describe the Neural Network in the next subsection as we will see later it performs the best.

3.1.1 Neural Network

A neural network consists of a series of layers each with a specified number of nodes. Initially an input layer, which consists of the input vector is multiplied by a weight matrix and added to a bias vector. That is

$$a_j^1 = \sum_k w_{jk}^1 a_k^0 + b^{(1)_j}$$

where w_j^1 is the vector associated with layer 1 node j, a^0 is the input vector and b^1 is the bias vector for layer 1. a_j^1 is the feed into a non-linear function which in our case is a Leaky ReLU which is defined as:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0.01x & \text{otherwise} \end{cases}$$

After which the result $f(a_j^1)$ gets passed on to the next layer where the same process is repeated for how many hidden layers there are in the network before finally reaching the output layer which gives the desired output y' from the network. For our network we used two hidden layers, one with 100 nodes and the other 50.The output is then compared with the value we wish to predict,y by some error metric. For us we used Cross Entropy Loss which is defined as :

$$f(y, y') = -\sum_i y_i' log(y_i)$$

After this loss is computed we can use it update the weights and biases of the network using gradient decent and an algorithm called back-propagation. For an overview on back propagation please see [15]. For gradient decent we used ADAM of which more details can be found here [21]

3.2 Sequential Models

3.2.1 N-Grams

N-grams are simply adjacent sequences of n items. They are produced by generating the all contiguous n elements in a sequence. For example given the sentence "This is a tree.' All of its 1-grams, uni-grams, at a word level are {This,is,a,tree}, its 2-grams,bi-grams, are {This is,is a, a tree} and so on and so forth. We can then use these N-grams to build a probabilistic language model for sequence prediction in the form of a $(n-1)$ order Markov Model. That is we first make the assumption that the next element in a sequence is only based on the $(n-1)$ elements that precede it which is more formally:

$$P(w_m) = P(w_m | w_{m-1}, \ldots, w_{m-n})$$

Then we count the number of times $w_{m-1}, \ldots, w_{m-n}, w_m$ in our data and divide that the number of times $w_{m-1}, \ldots, w_{m-n}$ appears to get $P(w_m)$. We do this for all n-grams $w_{m-1} \ldots w_{m-n}$ in our corpus of data. Then when we see that n-gram we can predict the next element as the one with the highest probability. N-Gram models have been used successfully in NLP [11], in game AI and procedural content generation [8], [40] which made it a good candidate approach for this project.

3.2.2 Long Short Term Memory Networks

Long Short Term Memory Networks represent the current state of the art in sequence prediction and classification. They are a type of Recurrent Neural Networks (RNN)s with specific structures in place to alleviate the vanishing gradient problem found in vanilla RNNs and structures to learn when to forget,filter and output information. Specifically each instance of the network outputs two nodes: a hidden state and a cell state. The hidden state is the output of the instance and the cell state carries some information about what the network aught to remember. These structures are other neural nets with their own weights. Specifically four networks called 'gates' which are comprise of; the forget gate which is in charge of learn when to forget information and what information to forget, the candidate gate which picks out the most relevant candidate information from the input, the input gate which is just a non-linear function applied to the input and the output layer which is in charge of figuring out what to output. More concretely the network is defined as:

$$\begin{aligned} f_t &= \sigma(W_f x^t + U_f h_{t-1}) \\ C_t^{'} &= tanh(W_C x^t + U_C h_{t-1}) \\ I_t &= \sigma(W_I x^t + U_I h_{t-1}) \\ O_t &= \sigma(W_O x^t + U_O h_{t-1}) \\ C_t &= f_t \odot C_{t-1} + I_t \odot C_t^{'} \\ H_t &= O_t \odot tanh(C_t) \end{aligned}$$

where W_*^t,U_*^t are weight matrices weight vectors for f, C, I, O at time step t, x_t is the input at time step t, C_t the cell state at time step t and H_t is the hidden state at time step t. This network updates

via back-propagation through time. For a more complete overview of LSTMs please refer to [17]. The network used for this project had 1 layer with 52 cells.

4 Experiment

4.1 Dataset

The dataset used for this project was obtained using the OpenDota API [29], an open source platform providing information about almost every public match played in DoTA 2. However due to the significant computational and storage cost of such an endeavour only a subset of these games have "advanced data" which is to say more information than player ids and what heroes were picked. Tournament games are part of that subset and thus for this project we downloaded all the competitive games we could find for four professional players and parsed them to get the relevant information about which heroes they picked. We were able to obtain 1556 games for player 34505203, 2995 games for player 82262664, 1448 games for player 87278757 and 1688 for player 106863163.

Figure 1 shows a graph of the frequency of the top 10 heroes picks that each player. As we can see there is a lot of variance in the heroes that are not the most picked hero. This is due to the fact that most players pick one hero to be their first option or their 'main' but professional players must also be able to play other heroes well. Therefore we expect them to pick their main when possible and when not to pick the hero that they think fits the situation best. This causes player picks to be quite unpredictable which causes a level of dynamism that allows the game to have a high skill ceiling but makes predicting player picks really difficult.

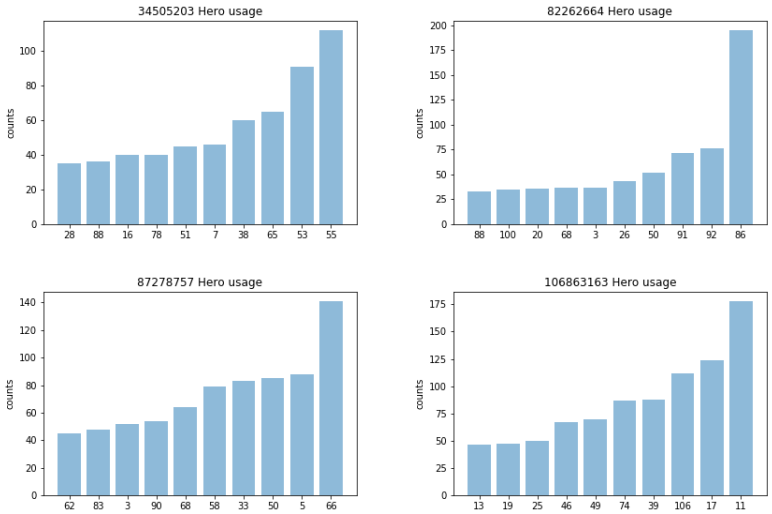


Figure 1: The 10 Most Picked Hero For each player

Since we are picking professional games, it is worth noting the game mode, Captain's Mode, that tournaments use for all of their matches. In Captain's Mode each team has a captain who makes the decision on what hero the team bans and picks. The captain does so after consultation with the team members and so their decisions reflect the desires of the team. The order of the draft is given Figure 2

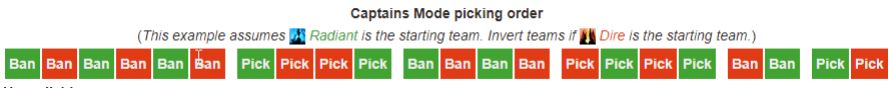


Figure 2: Draft order for Captain's mode

4.2 Feature Representation

4.2.1 Simple Representation

Initially we tried training various non-sequential models using only the hero picks of the player's team, the enemy team and the banned heroes. The representation of this was as follows:

$$x = [\texttt{game\_patch}, \texttt{ally\_hero\_1}, \texttt{ally\_hero\_2}, \ldots, \texttt{enemy\_hero\_1}, \texttt{enemy\_hero\_2}, \ldots, \texttt{banned\_hero\_1} \ldots \texttt{banned\_hero\_12}]$$
$$y = \texttt{hero\_picked}$$

where game_patch $\in 0, \ldots, 22$ is an identifier of the game version that this match was played in, ally_hero_X $\in 1, \ldots, 115$ are identifiers of which hero the player's team picked, enemy_hero_X $\in 1, \ldots, 115$ are identifiers of which hero the player's enemy team picked, banned_hero_X $\in 0, \ldots, 115$ are identifiers of which hero were banned and hero_picked $\in 1, \ldots, 115$ are identifiers of which hero the player picked. The banned hero ids can be 0 due to the fact in some games there are 10 bans while there are 12 in others. So in instances where there are only 10 heroes banned, banned_hero_11 and banned_hero_12 are both 0. For the Feed-Forward Neural Network we used one-hot encoding to represent all of the features. To reduce the feature space for the NN we collapsed all of the allied heroes into one feature vector of size 115, similarly to the enemy heroes and banned heroes. Thus we feed into the network three feature vectors of size 115 where 1 or 0 in the vector indicates the inclusion or exclusion of the hero whose id matches the index.

4.2.2 Role Based Representation

This representation did not perform well, results will be shown in a later section, and thus we had to think of other feature representations that capture more information about the latent factors not visible from the raw data. Valve, the developers of DoTA2, have assigned various roles to each hero that help players understand where that hero fits within the roster and how they expect each hero to be played. The community also have assigned a different set of roles to each hero which focuses more on the position of where the hero aught to be played. So we revised our representation with this information in mind and came up with the following:

$$x = [\texttt{game\_patch}, \texttt{num\_ally\_role\_1}, \texttt{num\_ally\_role\_2}, \ldots, \texttt{num\_enemy\_role\_1}, \texttt{num\_enemy\_role\_2}, \texttt{banned\_hero\_1} \ldots \texttt{banned\_hero\_12}]$$
$$y = \texttt{hero\_picked}$$

where game_patch is defined as above, num_ally_role_X $\in 0 \ldots 5$ is the number of heroes in the ally team that fulfill that role,num_enemy_role_X $\in 0 \ldots 5$ is the number of heroes in the enemy team that fulfill that role, banned_hero_1 and hero_picked are defined as above.

The main idea behind this representation is that players often pick complementary heroes to the ones that their team picked, and counters to the heroes that an enemy picks. If we know what roles are yet to be filled and roles that might be over-represented in our team and the enemy team then we have more information about what the player is likely to pick. We also added the banned heroes to inform the models about what heroes are not available for selection.

4.2.3 Sequence Representation

The sequential representation is fairly straight forward. We extracted the draft sequence from the games and parsed them into the following:

$$x = [\texttt{TEAM\_PICK/BAN\_heroID}, \texttt{TEAM\_PICK/BAN\_heroID}, \ldots]$$
$$y = \texttt{hero\_picked}$$

where TEAM indicates if the team deciding is the ally team or the enemy team, PICK/BAN indicates if the next step is a ban or a pick,hero_ID $\in 1, \ldots, 115$ is an identifier of which hero was being picked/banned and hero_picked is defined as above. Here is an example of such a sequence:

```
BAN_ENEMY_TEAM_3,BAN_ALLY_TEAM_7,BAN_ENEMY_TEAM_107,BAN_ALLY_TEAM_112,
BAN_ENEMY_TEAM_38,BAN_ALLY_TEAM_41,PICK_ENEMY_TEAM_51,PICK_ALLY_TEAM_60
```

We truncate the sequence before the player we are interested in picks a hero, and thus simulate the sequence of events that occur before the player makes their decision. For the LSTM model we used word embedding via a simple look up table with an embedding dimension of 32 before feeding the sequence into the network.

4.3 Baseline

As far as we are aware no one has attempted to do player imitation for DoTA2 player picks. And thus there are no state-of-the-art methods to compare our method against. Therefore we use a somewhat qualitative baseline of picking the player's most picked hero for every game in the test set. Despite its simplicity it does reflect some assumptions made by actual players where they predict their opponent to play their main whenever possible.

4.4 Results

4.4.1 Non-Sequential Problem

Figure 3 shows the results of running Linear SVM,Random Forest with the Deep Feed-Forward Neural Network on the simple representation.We see that the results do outperform the baseline with the Neural Network doing better than all the other model for all the players. However since the model has very little data and external information outside of the draft picks themselves, it still does rather poorly. As such we developed the representation mentioned in the last section. We ran the same models on this data and Figure 4 shows the results

Method	Player 87278757	Player 34505203	Player 82262664	Player 106863163	Average
SVM	0.092	0.085	0.136	0.105	0.105
Random Forest	0.116	0.104	0.121	0.114	0.114
NN	**0.197**	**0.136**	**0.161**	**0.152**	**0.162**
Baseline	0.083	0.057	0.061	0.081	0.071

Figure 3: Test accuracy on the original simple features

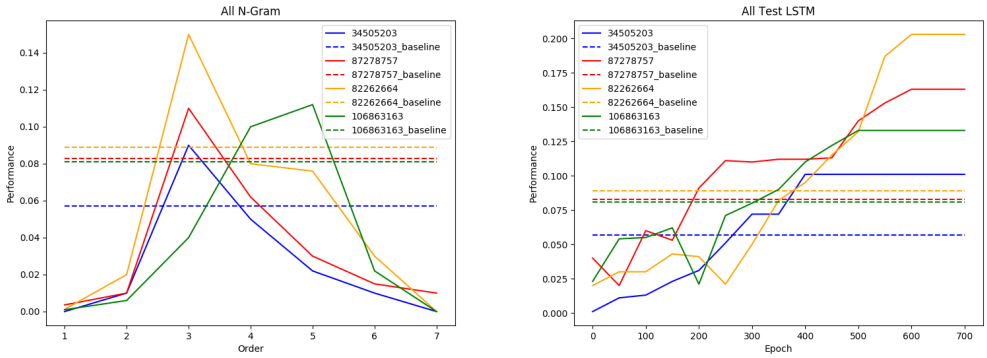
Method	Player 87278757	Player 34505203	Player 82262664	Player 106863163	Average
SVM	0.151	0.214	0.237	0.163	0.191
Random Forest	0.234	0.182	0.226	0.164	0.202
NN	**0.276**	**0.223**	**0.324**	**0.259**	**0.271**
Baseline	0.083	0.057	0.089	0.081	0.071

Figure 4: Test accuracy on the proposed feature representation

We see a significant improvement over the previous results, which is what we expect as we embedded more domain knowledge in the representation. Again we see that the Neural Network doing the best, with Random Forest and SVM being very close in some instances.

4.4.2 Sequential Problem

We generated n-grams of order $1 \ldots 7$ and ran the model on each order for each player. Figure 5a shows the test error across the orders. As we might expect performance goes up as we increase the number of events we look back on until we reach a peak and then the patterns we look for become too specific and performance goes down. Figure 5b shows the LSTM test error for every 50 epochs.



(a) N-Gram test error

(b) LSTM test error

Figure 5: Plots of test error of the sequential models

The LSTM network surpassed the N-Grams model as we expected but not by much. We suspect this is due to the fact that LSTM requires a lot of data per player and we did not have that many data points. They do both outperform our qualitative baseline, however the non-sequential models do fair a bit better. This may be due to the improved feature representation or the fact that the rest of the draft after the player's pick is not given to this model. We think it is a mixture of both as the sequential models do not have enough data to capture the latent factors that are more explicit in the proposed representation. And it is likely that team members communicate together before the draft starts and thus an earlier pick could be based on a later pick, which gives the non-sequential models an advantage.

5 Conclusion and Future Work

In this project we attempted to model a player's hero picks. We described two problems that this endeavour entails and presented two relatively successful deep learning methods that were able to predict what a specific player will pick given the other picks of the match. Though we did outperform the baseline, our models can still be greatly improved. Infusing more explicit information about the current state of the meta-game, the player's understanding of that meta-game and their hero preferences in the state representation is likely to greatly improve performance. There are also other game modes that non-professionals play which have a lot more data, though we were unable to retrieve that from OpenDota due to various technical complications. There are other platforms for gathering DoTA 2 data out there and so this is approach is not out of the question.Given the time constraints of this project we were unable to follow up on these avenues, but we hope we can further develop them in the future.

References

[1] Rakesh Agrawal et al. Fast algorithms for mining association rules. In *In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, pages 487–499.

[2] Anonymous. Player imitation for build actions in a real-time strategy game. *Unpublished*, May 2018.

[3] Bobby D. Bryant and Risto Miikkulainen. Acquiring Visibly Intelligent Behavior with Example-guided Neuroevolution. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, pages 801–808, Vancouver, British Columbia, Canada, 2007. AAAI Press.

[4] Luigi Cardamone, Daniele Loiacono, and Pier Luca Lanzi. Learning drivers for TORCS through imitation using supervised methods. pages 148–155, October 2009.

[5] Zhengxing Chen, Truong-Huy D. Nguyen, Yuyu Xu, Chris Amato, Seth Cooper, Yizhou Sun, and Magy Seif El-Nasr. The Art of Drafting: A Team-Oriented Hero Recommendation System for Multiplayer Online Battle Arena Games. *arXiv:1806.10130 [cs]*, June 2018. arXiv: 1806.10130.

[6] H. C. Cho, K. J. Kim, and S. B. Cho. Replay-based strategy prediction and build order adaptation for StarCraft AI bots. In *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–7, August 2013.

[7] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.

[8] Steve Dahlskog, Julian Togelius, and Mark J. Nelson. Linear levels through n-grams. In *Proceedings of the 18th International Academic MindTrek Conference on Media Business, Management, Content & Services - AcademicMindTrek '14*, pages 200–206, Tampere, Finland, 2014. ACM Press.

[9] Ethan W. Dereszynski, Jesse Hostetler, Alan Fern, Thomas Dietterich, Thao-Trang Hoang, and Mark Udarbe. Learning Probabilistic Behavior Models in Real-Time Strategy Games. January 2011.

[10] G. M. Farouk, I. F. Moawad, and M. M. Aref. A machine learning based system for mostly automating opponent modeling in real-time strategy games. In *2017 12th International Conference on Computer Engineering and Systems (ICCES)*, pages 337–346, December 2017.

[11] Georgia Frantzeskou, Efstathios Stamatatos, Stefanos Gritzalis, Carole E Chaski, and Blake Stephen Howald. Identifying Authorship by Byte-Level N-Grams: The Source Code Author Profile (SCAP) Method. 6(1):18, 2007.

[12] Benjamin Geisler. *An Empirical Study of Machine Learning Algorithms Applied to Modeling Player Behavior in a 'First Person Shooter' Video Game*. PhD thesis, University of Wisconsin - Madison, 2002.

[13] Lucas Hanke and Luiz Chaimowicz. A Recommender System for Hero Line-Ups in MOBA Games. page 7.

[14] Brent Harrison and David L Roberts. Analytics-Driven Dynamic Game Adaption for Player Retention in a 2-Dimensional Adventure Game. page 7.

[15] G Hinton and T Sejnowski. A theoretical framework for back-propagation.

[16] Tin Kam Ho. Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE, 1995.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] Christoffer Holmgård, Michael Cerny Green, Antonios Liapis, and Julian Togelius. Automated Playtesting with Procedural Personas through MCTS with Evolved Heuristics. *arXiv:1802.06881 [cs]*, February 2018. arXiv: 1802.06881.

[19] J. L. Hsieh and C. T. Sun. Building a player strategy model by analyzing replays of real-time strategy games. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 3106–3111, June 2008.

[20] Niels Justesen and Sebastian Risi. Learning Macromanagement in StarCraft from Replays using Deep Learning. *arXiv:1707.03743 [cs]*, July 2017. arXiv: 1707.03743.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] John E. Laird and John C. Duchi. Creating human-like synthetic characters with multiple skill-levels : A case study using the Soar quakebot. In *AAAI Fall Symposium Technical Report*, 2000.

[23] Joan Marc Llargues Asensio, Juan Peralta, Raul Arrabales, Manuel Gonzalez Bedia, Paulo Cortez, and Antonio Lopez Peña. Artificial Intelligence approaches for the generation and assessment of believable human-like behaviour in virtual characters. *Expert Systems with Applications*, 41(16):7281–7290, November 2014.

[24] Adam M Smith, Chris Lewis, Kenneth Hullett, Gillian Smith, and Anne Sullivan. An Inclusive Taxonomy of Player Modeling. April 2011.

[25] Maximiliano Miranda, Antonio A. Sánchez-Ruiz, and Federico Peinado. A Neuroevolution Approach to Imitating Human-Like Play in Ms. Pac-Man Video Game. In David Camacho, Marco Antonio Gómez-Martín, and Pedro Antonio González-Calero, editors, *Proceedings of the 3rd Congreso de la Sociedad Española para las Ciencias del Videojuego, Barcelona, Spain, June 29, 2016*, volume 1682 of *CEUR Workshop Proceedings*, pages 113–124. CEUR-WS.org, 2016.

[26] J. Muñoz, G. Gutierrez, and A. Sanchis. A human-like TORCS controller for the Simulated Car Racing Championship. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, pages 473–480, August 2010.

[27] Derek Neal and Bruce Hayles. Designing AI for Competitive Games, 2016.

[28] Santiago Ontañón, Kinshuk Mishra, Neha Sugandh, and Ashwin Ram. Case-Based Planning and Execution for Real-Time Strategy Games. In *Case-Based Reasoning Research and Development*, Lecture Notes in Computer Science, pages 164–178. Springer, Berlin, Heidelberg, 2007.

[29] OpenDota. Opendota - dota 2 statistics, 2018.

[30] Steffen Priesterjahn, Oliver Kramer, Alexander Weimer, Er Weimer, and Andreas Goebels. Evolution of Reactive Rules in Multi Player Computer Games Based on Imitation. In *Proceedings of the International Conference on Natural Computation*, 2005.

[31] G. Robertson and I. Watson. Building behavior trees from observations in real-time strategy games. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–7, September 2015.

[32] Jacob Schrum, Igor V. Karpov, and Risto Miikkulainen. UT^2: Human-like Behavior via Neuroevolution of Combat Behavior and Replay of Human Traces. In *Computational Intelligence in Games*, 2011.

[33] Aleksandr Semenov, Peter Romov, Sergey Korolev, Daniil Yashkov, and Kirill Neklyudov. Performance of Machine Learning Algorithms in Predicting Game Outcome from Drafts in Dota 2. In Dmitry I. Ignatov, Mikhail Yu. Khachay, Valeri G. Labunets, Natalia Loukachevitch, Sergey I. Nikolenko, Alexander Panchenko, Andrey V. Savchenko, and Konstantin Vorontsov, editors, *Analysis of Images, Social Networks and Texts*, pages 26–37, Cham, 2017. Springer International Publishing.

[34] Marius Stanescu, Nicolas A. Barriga, Andy Hess, and Michael Buro. Evaluating Real-Time Strategy Game States Using Convolutional Neural Networks. In *Computation Intelligence and Games*, September 2016.

[35] Adam Summerville, Michael Cook, and Ben Steenhuisen. Draft-Analysis of the Ancients. page 7.

[36] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[37] Gabriel Synnaeve and Pierre Bessiere. A Bayesian Model for Plan Recognition in RTS Games applied to StarCraft. In *Proceedings of the 7th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2011*, November 2011.

[38] Valve. Dota2 the international, 2018.

[39] Niels van Hoorn, Julian Togelius, Daan Wierstra, and Juergen Schmidhuber. Robust player imitation using multiobjective evolution. pages 652–659, June 2009.

[40] Joseph Vasquez II. Implementing n-grams for player prediction, procedural generation, and stylized ai. *COLLECTED WISDOM OF GAME AI PROFESSIONALS*, page 567.

[41] Pu Yang, Brent Harrison, and David L Roberts. Identifying Patterns in Combat that are Predictive of Success in MOBA Games. page 8.