DOTA 2 CHARACTER RECOMMENDATION SYSTEM BASED ON ASSOCIATION RULES USING FP-GROWTH AND MULTILAYER PERCEPTRON ALGORITHM

Erwin Tobing[1], Kania Evita Dewi[2]

[1,2] Informatics Engineering Department - Universitas Komputer Indonesia
Jl. Dipatiukur 112 – 114 Bandung
E-mail : erwintobing15@gmail.com[1], kania.evita.dewi@email.unikom.ac.id[2]

ABSTRACT

Dota 2 is a Multiplayer Online Battle Arena (MOBA) computer game that has quite a lot of users around the world. Dota 2 is played by two teams, consisting of five players each, where each player controls a character named “hero”. One of the important factors that can determine the winner of a Dota 2 match is the hero picks. The combination of 5 out of 121 total heroes results in 198,792,594 possible combinations that can be picked. Picking the best combination of heroes from the total available combinations is not easy, especially when the time given to pick each hero is only about one minute. Therefore, a recommendation system can help players in picking the best combination of heroes [1]. In this study, a recommendation system was built, which provides hero suggestions from the combination of heroes that often win when they are picked together. The recommendation system is built based on Association Rules using the FP-Growth algorithm, with a dataset of 621,064, collected from Dota 2 public matches from May to July 2021. To test the performance of the recommendation system, a Multilayer Perceptron model was trained which can predict the winning team of Dota 2 matches with an accuracy of 63.43%. The results of the recommendation system performance obtained a success rate of 81% in 1000 simulations.

Keywords: Recommendation System, Association Rules, FP-growth, Multilayer Perceptron, Dota 2.

1. INTRODUCTION

Dota 2 is played by two teams, consisting of five players each. Each player controls a hero who has different abilities (skills) and roles (roles) one from the other. Each team occupies and defends a base that has a core or a crown called "ancient". The first team that manages to destroy the enemy ancient comes out victorious. In Dota 2 there are many factors that can determine the winner of a match. One important factor that is very influential in the selection of the right hero. Choosing the right combination of heroes will increase a team's winning percentage. Choosing 5 combinations hero from a total of 121 heroes is not easy, especially to pick each hero only given 30-60 seconds. For this reason, a hero recommendation system will certainly help players in choosing the best combination of heroes [1].

In a study conducted by K. Conley and D. Perry [2], a Dota 2 character recommendation system has been developed based on clustering with the K-Nearest Neighbors and Logistic Regression algorithms. While in a study conducted by Z. Chen et al. [3] the character recommendation system uses the Monte Carlo Search Tree algorithm. In other work L. Hanke and L. Chaimowicz [1] developed a recommendation system based on Association Rules, using the Apriori algorithm and Multilayer Perceptron (MLP) with a dataset of 70,000. It is reported the recommendation system built by L. Hanke and L. Chaimowicz [1] has a success rate of 74.9% from 1000 trials. Apriori algorithm is a traditional algorithm that is easy to use in building a recommendation system. However, the use of the Apriori algorithm in extracting frequent patterns is still inefficient, because the data scanning process is repeated in each iteration and the presence of candidate generation, a process that requires large memory space and takes a long executing time [4]. This makes the Apriori algorithm less suitable for extracting association rules with relatively large datasets measuring hundreds of thousands to millions.

From the description above, an association rule algorithm that is more efficient in space and time is needed, so that it is suitable for large datasets, for building a recommendation system. One of the traditional association rule algorithms with better efficiency than the Apriori algorithm is the FP-Growth algorithm. The FP-Growth algorithm scans data only twice, and there is no candidate generation process. An example of the application of the FP-Growth algorithm as in the study conducted by H. Li et al [5] which develops a query recommendation using parallel FP-Growth. In this study, they succeeded in applying effective and fast data mining techniques to 802,939 web data and 1,021,107 tag data.

Therefore, in this study we intend to build a recommendation system based on Association

Rules, which was extracted from a dataset of 621,064 in size, using traditional but efficient algorithm FP-growth and Multilayer Perceptron model for match prediction.

2. RESEARCH CONTENTS

2.1 Data Collection

The data used is collected from the web API www.opendota.com. The data collected is in the form of Dota 2 public match data from April 2021 - July 2021 patch 2.79 (Dota 2 version in this study). The data is collected under conditions :

1. Completed match. The match ends normally and is not canceled, usually due to a player experiencing internet problems.
2. Matches with average MMR between 2000-7000. This MMR range is quite competitive to ensure players choose a good understanding hero.
3. All picked ranked match mode. The most used match mode of public matches today.

The results of the raw data collected from the web API can be seen in Figure 1.

match_id	match_seq_num	radiant_win	start_time	duration	avg_mmr	num_mmr	lobby_type	game_mode	avg_rank_tier	num_rank_tier	cluster	radiant_team	dire_team
5937680909	4978110542	FALSE	1618056811	947	5384	2	7	22	80	5	155	19,120,107,79,94	31,49,70,110,25
5937680202	4978115238	TRUE	1618056560	1116	3938	3	7	22	64	8	171	128,5,8,59,112	85,6,56,42,45
5937677706	4978115412	TRUE	1618056484	1245	4343	2	7	22	52	5	151	135,70,104,59,31	58,44,107,46,87
5937677414	4978110742	FALSE	1618056474	1019	3626	3	7	22	57	7	274	20,71,82,114,66	26,46,89,129,14
5937677106	4978110923	TRUE	1618056464	1082	3216	3	7	22	43	7	227	26,8,104,87,11	73,70,135,42,67
5937676309	4978118016	TRUE	1618056434	1371	3338	1	7	22	48	2	274	7,87,8,71,59	101,40,67,106,74
5937671711	4978114369	FALSE	1618056284	1389	4114	2	7	22	66	7	183	11,98,35,18,87	46,85,10,27,99
5937670916	4978117744	TRUE	1618056274	1568	4129	4	7	22	71	4	152	110,42,76,107,114	60,74,70,26,21
5937670210	4978116932	FALSE	1618056250	1505	4116	3	7	22	53	3	186	7,35,26,135,54	44,14,87,16,62
5937669906	4978117879	TRUE	1618056248	1582	4186	1	7	22	63	2	274	48,28,41,121,96	6,104,25,17,43
5937669416	4978114978	TRUE	1618056190	1463	4174	2	7	22	67	4	182	70,101,5,28,74	93,7,110,76,68
5937668217	4978119072	TRUE	1618056186	1688	3047	1	7	22	33	3	274	7,82,59,16,9	97,101,105,75,13
5937667503	4978117208	TRUE	1618056174	1545	3430	3	7	22	72	4	274	62,93,96,121,45	135,54,106,108,69
5937667209	4978116286	TRUE	1618056154	1598	3863	2	7	22	55	5	182	81,75,49,45,121	135,41,85,112,77
5937668217	4978116892	TRUE	1618056135	1394	3672	4	7	22	59	8	236	21,37,32,71,99	14,12,36,128,104
5937666108	4978119775	TRUE	1618056114	1792	3856	5	7	22	64	6	274	58,39,54,74,28	21,93,2,76,23
5937665905	4978119296	FALSE	1618056124	1705	3856	4	7	22	78	5	155	98,13,100,53,1	67,106,21,27,42
5937665717	4978110442	TRUE	1618056109	1381	4012	3	7	22	65	5	184	59,87,62,57,89	44,72,20,49,51
5937665212	4978110932	FALSE	1618056104	1418	3457	1	7	22	49	3	151	72,62,135,35,121	13,111,8,53,84
5937665001	4978104220	TRUE	1618056081	1024	3810	2	7	22	58	4	251	16,15,128,123,44	32,60,7,39,13
5937664000	4978119373	TRUE	1618056055	1708	3945	4	7	22	51	5	191	112,47,26,28,89	50,60,105,129,16
5937663804	4978109796	FALSE	1618056043	1373	3228	3	7	22	53	5	274	48,38,126,23,107	31,61,17,27,42
5937663300	4978114872	FALSE	1618056054	1854	3678	3	7	22	61	7	187	12,88,84,101,120	135,1,96,104,64
5937661502	4978117430	TRUE	1618055994	1785	3305	1	7	22	28	4	193	80,2,16,48,5	42,6,135,105,25
5937661203	4978116844	TRUE	1618055969	1783	3921	2	7	22	55	6	154	14,26,75,54,19	126,62,135,95,120

Figure 1. Dota 2 Match Dataset

Dota 2 match dataset is collected by a data collector program that fetches 500 new rows of data every hour from the web API. It can collect up to a maximum of 12,000 per day or 360,000 per month. Every new data retrieved from the web API is combined with the old data that has been previously collected. Data collection process starts with fetching raw data from web API as JSON format, processed with python pandas library into CSV data. Data collection process can be seen in Figure 2.

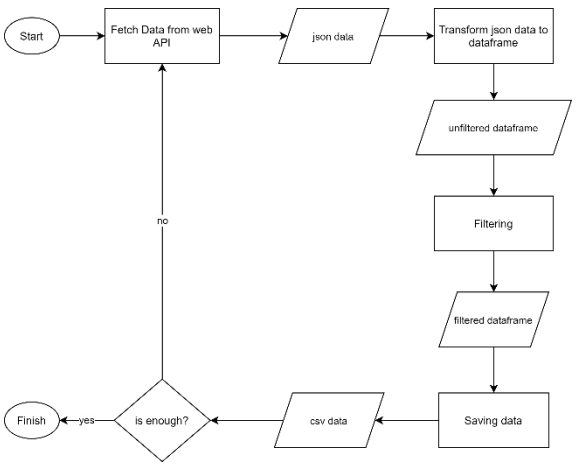

Figure 2. Data Collection

Data collection is carried out for four months every day from April to July 2021. The final result of the data collection process is concluded with a total of 621,064 datasets obtained.

2.2 Preprocessing

Preprocessing is the process of preparing raw data into data that is ready to be used, in this case, it is used in extracting association rules and training multilayer perceptron model. The preprocessing stage consists of overcoming data redundancy, selecting the attributes to be used, then from here the stages are divided into two processes. The first is data processing for extracting association rules and the second is data processing for MLP learning. The preprocessing stages can be seen as shown in.

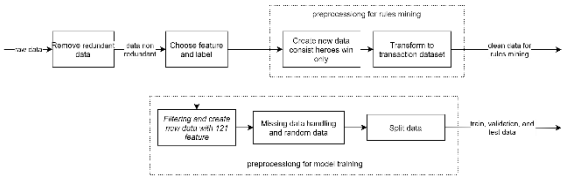

Figure 3. Preprocessing

1. Remove Redundant Data

The raw data are investigated to check if there are duplicated data or not. Data redundancy is checked based on the match_id column, if there is the same match_id then the duplicate data is removed.

2. Choosing Feature and Label

In the raw data the attributes that will be used, both for extracting association rules and training the classification model, are radiant_team, and dire_team, and using radiant_win as data labels. The data is selected by three-column mentions before, so resulting in a new data, in the form of hero selection data by both teams and the winner of the match as shown in the Table 1.

Table 1. Hero Selection and Match Winners Data

radiant_team	**dire_team**	**radiant_ win**
19,120,107,79,94	31,49,70,110,25	*False*
128,5,8,59,112	85,6,56,42,45	*True*
135,70,104,59,31	58,44,107,46,87	*True*
...	...	...
5,21,8,34,23	18,30,40,74,56	*True*
12,26,7,54,119	67,101,106,15,84	*True*
87,114,71,13,16	26,21,104,8,25	*False*

3. Preprocessing For Rules Mining

For association rules mining, the data that will be used is only the list of winning heroes data. Therefore, the data in Table 1 is processed into a list of winning heroes based on the radiant_win label into new data as shown in Table 2.

Table 2. Winning Heroes List

win_heroes
31,49,70,110,25
128,5,8,59,112
135,70,104,59,31
…
5,21,8,34,23
12,26,7,54,119
26,21,104,8,25

4. Preprocessing For Model Training

Data used for Multilayer Perceptron training is preprocessed with some preprocessing techniques such as filtering, remove missing data, randomizing the data and split the data into training data, validation data, and testing data.

Filtering is done to transform data into new data with 121 attributes and 1 label. Each hero represents an attribute, while the new label uses the old label radiant_win. The value of each new data attribute is filled in based on the previous data with the conditions as in equation (1). Meanwhile, for label values, replace True with 1 and False with 0.

$$att\_value = \begin{cases} 1, & if\ hero_i \in radiant\_team \\ -1, & if\ hero_i \in dire\_team \\ 0, & others\ (not\ picked) \end{cases} \quad (1)$$

Dataset were randomized to eliminate the data ordering bias based on the time of collection. Dataset then split into training data, validation data, validation data, and test data with the ratio of 70: 15: 15. The final form of the preprocessing data can be seen in Figure 1.

hero_1	hero_2	hero_3	hero_4	hero_5	…	hero_126	hero_128	hero_129	hero_135	radiant_win
0	0	0	0	0	…	0	0	0	0	0
0	0	0	0	1	…	0	1	0	0	1
0	0	0	0	0	…	0	0	0	1	1
0	0	0	0	0	…	0	0	-1	0	0
0	0	0	0	0	…	0	0	0	-1	1
0	0	0	0	0	…	0	0	0	0	1
0	0	0	0	0	…	0	0	0	0	0
0	0	0	0	0	…	0	0	0	0	1
0	0	0	0	0	…	0	0	0	1	0
0	0	0	0	0	…	0	0	0	0	1
0	0	0	0	1	…	0	0	0	0	1
0	0	0	0	0	…	0	0	0	0	1
…	…	…	…	…	…	…	…	…	…	…
0	1	0	0	0	…	0	0	0	0	1
1	0	0	0	0	…	0	0	0	0	0
0	0	0	0	0	…	0	0	0	0	0

Figure 1. Dataset For Model Training

2.3 Association Rules Mining with FP-Growth

Recommendation system provides suggestions for hero combinations based on a set of rules in the form of rules [6]. The set of rules is called Association Rules which is obtained from the rules mining from dataset winning hero list. The process of extracting Association Rules broadly consists of two stages which include :

1. find all hero combination frequent patterns and
2. generate strong association rules from frequent patterns.

Frequent hero combinations are obtained by determining the minimum support value. From a set of frequent hero combinations obtained, a set of association rules is generated. The Association Rules are filtered by a minimum confidence set so that they can be said to be feasible or often referred to as strong [7]. The support value is calculated using equation (2) and the confidence value is calculated using equation (3).

$$support(A => B) = P(A \cup B) \quad (2)$$

$$confidence(A => B) = P(B|A) \quad (3)$$

The process of extracting association rules with the FP-Growth algorithm can be seen in Figure 2.

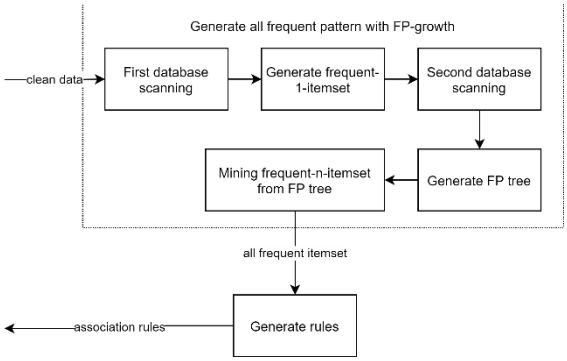

Figure 2. Association Rules Mining

The result of association rules mining is a set of rules consisting of antecedent and consequent (antecedent → consequent) with the support and confidence values meeting the specified minimum value. An example of the results of a list of rules is shown in Table 1.

Table 1. List of Association Rules Sample

Rules	Confidence
95, 69 → 64	100%
95 → 69, 64	66%
69 → 64	66%
69 → 36	66%
69 → 62	66%

95 → 36	66%
95 → 64	66%
95 → 69	66%

2.4 Model Training

The hero recommendation system based on Association Rules will suggest a combination of 5 heroes against 5 enemy heroes. One way to evaluate whether the heroes recommendation will win or not is using model classification. Therefore, in this study, one of the supervised machine learning algorithms is used, namely Multilayer Perceptron (MLP) [8] to predict the winning rate of the recommended hero combination.

2.4.1 Architecture and Hyperparameter

Multilayer perceptron consists of three main layers, that is input layer, hidden layer and the output layer [9]. The input layer consists of one neuron for each hero which is a total of 121 heroes or in other words, consists of 121 inputs. The value of each input is 1, -1, or 0 according to the explanation in the preprocessing. The hidden layer used is one, while the number of neurons in the hidden layer is 300. The activation function used in the hidden layer is the GeLu activation function. The output layer consists of one neuron with binary values 1 and 0. A value of 1 indicates the recommended hero is the winner, while a value of 0 indicates otherwise. The activation function used is the binary activation function Sigmoid.

In MLP training to get the model with the best performance, hyperparameter tuning is performed. This process is carried out by following the rules based on Heaton [10] and by trial-and-error. In addition, the experiment was conducted to change architecture and lastly evaluation with k-fold cross validation. The final result of the MLP architecture used is as shown in Figure 3.

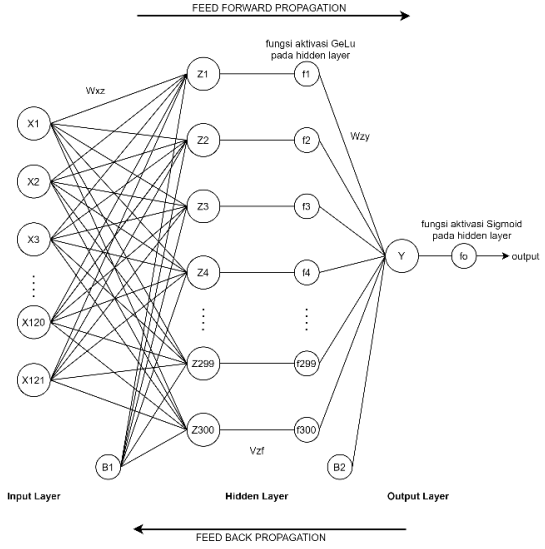

Figure 3. Multilayer Perceptron Architecture

Meanwhile hyperparameter used in multilayer perceptron is shown in Table 2.

Table 2. Multilayer Perceptron Hyperparameter

Hyperparameter	**Value Used**
Input Neuron	121
Hidden Layer	1
Neuron pada Hidden Layer	300
Hidden Layer Activation	GeLu
Output neuron	1
Output Layer Activation	Sigmoid
Learning Rate (α)	0,001
Momentum	0,0
Epoch	10
Batch	80
Optimizer	SGD
Weight Initialization	random_uniform
Weight Constrain	3
Dropout	0,4
Loss Function	Binary cross entropy

2.4.2 Backpropagation

Multilayer perceptron was trained using Backpropagation algorithm. Multilayer perceptron training is carried out in two step first a forward manner (feed-forward) to calculate the comparison of the output value and the expected target value. Then the second step is to calculate the error propagation backward (feed-back) in order to improve the synaptic weights on all existing neurons.

2.5 Build Hero Website Recommendation

The recommendation website development is carried out using the Django website development framework [11]. Association rules and models obtained from the previous process are imported into the main directory of the Django website, then used as hero recommendation and match prediction functions that can be called on the website. The hero recommendation function is built with the same logic as in Algorithm 1.

Algorithm 1 The algorithm is used to recommend heroes by utilizing association rules in the form of, initializing at least one hero. The input is an array of integer in the form of a list of radiant and dire team

heroes, a list of antecedents and consequents from association rules, a list of hero recommendations that have been initialized. The output is a list of the latest recommended heroes as a result.

```
result ← []
function get_recommendation(radiant_heroes,
                            dire_heroes,
                            antecendents,
                            consequents,
                            recommend_heroes)

 total_rules ← antecedents.size
 total_radiant ← radiant_heroes.size

 for i ← 1 to total_radiant do
  for i ← 1 to total_rules do
    if radiant_heroes[i] in antecendents then
     if radiant_heroes[i] not in recommend_heroes then
      if radiant_heroes[i] not in radiant_heroes then
       if radiant_heroes[i] not in dire_heroes then
        result.insert(radiant_heroes[i])
       endif
      endif
     endif
    endif
  endfor
 endfor

 return result
endfunction
```

The final implementation of hero recommendation system and match predictions as a web application can be seen as shown below Figure 4.

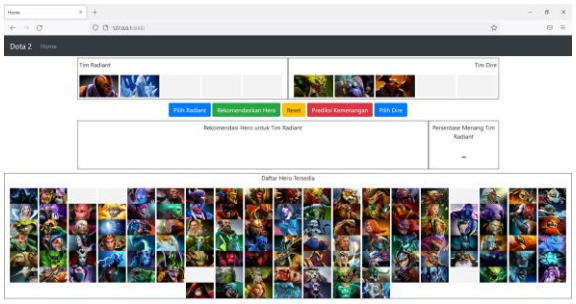

Figure 4. Hero Recommendation Website Interface

2.6 Experiments and Results

2.6.1 Association Rules

1. Experiments Plan

Association rules mining is carried out by setting the support and confidence values to get strong association rules that include the highest number of heroes. The support value is initialized as 0.1 and then adjusted by lowering the support value from a minimum. The confidence value is initialized at 10%, then adjustments are made by increasing the confidence value as much as possible [6]. Testing of association rules is also carried out to calculate the execution time of the excavation process for each specified support and confidence value.

2. Mining Result

Association rules mining result following experimental plan mention before can be seen as in Table 3.

Table 3. Association Rules Mining Result

Support	*Confidence*	Execution Time	*Rule Total*	Id Hero Total
0,1	10%	4 detik	0	0
0,01	10%	38 detik	13	8
0,01	20%	34 detik	0	0
0,01	15%	36 detik	0	0
0,01	11%	37 detik	12	8
0,01	12%	34 detik	6	5
0,001	10%	34 detik	273	112
0,0009	10%	36 detik	297	113
0,0008	10%	32 detik	317	115
0,0007	10%	27 detik	354	115
0,0006	10%	29 detik	429	115
0,0005	10%	30 detik	576	116
0,0004	10%	31 detik	810	117
0,0003	10%	31 detik	1297	118
0,0002	10%	40 detik	2322	119
0,0001	10%	45 detik	4550	119

2.6.2 Multilayer Perceptron

1. Experiments Plan

MLP training is carried out by initializing the architecture and initial hyperparameters [1]. Furthermore, to get the model with the best performance, hyperparameter tuning was carried out using RandomizedSearchCV, setting the model architecture and evaluating the model with kfold. The hyperparameters and model architecture that will be tuned include epoch, batch_size, learning_rate, momentum, weight initialization, activation function, dropout regulation, neurons in the hidden layer, and the number of hidden layers.

2. MLP Performance

MLP training and evaluation with architectural settings and hyperparameter tuning obtained the best results as shown in Table 4.

Table 4. MLP Performance with Hyperparameter Tuning

Hidden Layer	***Hidden Neuron***	***Epoch***	***Batch***	**Akurasi (%)**

1	300	10	80	63,26

MLP training and evaluation with kfold data selection k=10 as shown in Table 5.

Table 5. Hasil Pengujian MLP dengan *Kfold*

k	Accuracy (%)
1	62,89
2	63,25
3	62,81
4	63,11
5	63,47
6	63,07
7	63,17
8	63,12
9	63,18
10	63,34
Average	**63,14**

2.6.3 Recommendation System Performance

1. Experiments Plan

The recommendation system performance test is carried out by match simulation [1][12]. The match simulation is carried out by predicting the victory of the selected hero based on a recommendation system against enemy heroes that are randomly selected from a list of all heroes or from the 10 most frequently selected heroes in patch 2.79. The simulation is carried out 1000 times and then the percentage of the team's winnings is calculated using a recommendation system. The match simulation runs with logic as in Algorithm 2.

Algorithm 2 Dota 2 match simulation algorithm. Team A chooses a hero based on the recommendation system and Team B randomly. Then returns the result that is the proportion of team A's wins against team B.

```
function simulation()
 team_A, team_B ← []
 while team_A.size ≠ 5 and team_B.size ≠ 5 do
  team_B.insert(random enemy)
  if team_A.size = 0 then
   team_A.insert(random available hero)
  else
   recommendation ← get_recommendation(team_A,
                                       team_B,
                                       antecendents,
                                       consequents,
                                       recommend_heroes)
   team_A.insert(random recommendation)
  endif
 endwhile
 result ← get_prediction(team_A,team_B)
 return result
endfunction
```

2. Recommendation System Success Rate

The results of the recommendation system with 1000 times match simulation can be seen in Table 6.

Table 6. Recommended System Performance Results

Support	Confidence (%)	Hero Total	Success Rate (%)
0,001	10	112	76
0,0009	10	113	77
0,0008	10	115	78
0,0007	10	115	77
0,0006	10	115	81
0,0005	10	116	81
0,0004	10	117	77
0,0003	10	118	71
0,0002	10	119	75
0,0001	10	119	78

2.7 Study

2.7.1 Association Rules Result

Association rules mining begins with an initial support value of 0.1 and a confidence value of 10%. The result is 0 association rules consisting of 0 hero id. The support value was lowered to 0.01. The results obtained 13 association rules consisting of 8 hero ids. In the 7th experiment with a support value of 0.001 and a confidence of 10%, it has resulted in quite a lot of association rules, 273 in total consisting of 112 hero ids. Furthermore, the support value is reduced by a multiple of 1/10 starting from 0.0009 to 0.0001, the result is that the most association rules are 4550 consisting of 119 hero ids. When we try increasing the confidence value from 10% to 11%, 12%, 15%, up to 20%, the result is that the number of association rules is getting smaller so that the confidence value is set at 10%.

Meanwhile, in the execution time column, the first try with a support value of 0.4 has an execution time of 4 seconds, while for support values of 0.1 to 0.0001 it has an execution time of 27-45 seconds. So the execution time of association rules mining on the data is 621,064, the fastest is 4 seconds and the longest is 45 seconds. It is concluded that the algorithm used is quite time efficient and is able to extract association rules on quite large data.

2.7.2 Multilayer Perceptron Performance

The accuracy of the MLP following the architecture and hyperparameter tuning and architectural settings obtained a result of 63.62% as

shown in Table 6. Next, the model was evaluated using 10-fold cross validation. The results obtained an average accuracy of 63.14% with the largest accuracy of 63.47 at k=5. From quite a number of methods that have been used to increase the accuracy of the model, the best accuracy result that can be obtained is 63.44%. This accuracy can be said to be worthy of acceptance. However, considering that the model used is a neural network, especially with large enough data, it is felt that the accuracy can still be improved.

The hyperparameter tuning and model architecture has been improved a best a possible, so it is assumed that the cause of the accuracy can not be higher than 63,44% is not from the model. We assume that the cause is data patterns that can only produce such accuracy. This is in accordance with Taskin's study [13] which states that the accuracy of supervised classification models, especially neural networks, is highly dependent on training data. Therefore, Taskin suggests conducting an investigation and selection of training data, in order to obtain a representation of training data with the best size and quality.

2.7.3 Recommendation System Performance

The results of the recommendation system performance simulations show that the lowest success rate is 71%, the average is 77.1%, and the highest is 81%. The highest results were obtained from testing the recommendation system using AR with support value = 0.0006 and confidence = 10% and also AR with support value = 0.0005 and confidence = 10%. Then the association rules used for the recommendation website are AR with support value = 0.0005 and confidence = 10%. which has more rules and total heroes.

In Hanke's study [1], a recommendation system built using the Apriori algorithm with 70,000 data from patch 7.06, was reported to have a success rate of 74.9%. In this study, the recommendation system built with the FP growth algorithm, using a dataset of size 621,064 from patch 7.29, obtained a better success rate of 81%.

3. CLOSING

In this study, there are three conclusions were drawn based on the test results obtained. First, FP growth algorithm is very efficient used for Association Rules mining with the fastest execution time of 4 seconds and the longest 45 seconds, on a dataset of 621,064 rows. Second, Multilayer Perceptron models that have been built have worse accuracy than the previous study. Even though hyperparameter tuning, architectural changes, the use of training data based on MMR, and evaluation with k-fold cross validation have been carried out, the accuracy of the model obtained cannot be higher than 63.64%. Lastly, the recommendation system built using the FP growth algorithm on a dataset of 621,064 in size, obtained a very good success rate of 81%.

Suggestions given for the development of the Multilayer Perceptron model are to investigate the dataset used in this study, do a training data selection to produce training data with better size and quality, or try to use Dota 2 match data from patches other than patch 7.29 if possible using older patches. The recommendation system built in this study is only based on the relationship between heroes, not taking other factors into account. Therefore, adding other factors such as player data in the form of player roles can improve the accuracy of the recommendation system in recommending heroes.

REFERENCES

[1] L. Hanke and L. Chaimowicz, "A recommender system for hero line-ups in MOBA games," *Proc. 13th AAAI Conf. Artif. Intell. Interact. Digit. Entertain. AIIDE 2017*, pp. 43–49, 2017.

[2] K. Conley and D. Perry, "How Does He Saw Me ? A Recommendation Engine for Picking Heroes in Dota 2," *Stanford Univ.*, 2013, [Online]. Available: http://www.youtube.com/watch?v=BAC9B9z%0Ahttp://static.squarespace.com/static/53c5ec88e4b0b121aee5783a/t/53c63905e4b013ba8748794b/1405499653351/Dota2.pdf.

[3] Z. Chen *et al.*, "The Art of Drafting: A Team-Oriented Hero Recommendation System for Multiplayer Online Battle Arena Games," 2018, [Online]. Available: http://arxiv.org/abs/1806.10130.

[4] E. Widiati , S.Kom and K. E. Dewi, S.Pd., M.Si, "Implementasi Association Rule Terhadap Penyusunan Layout Makanan Dan Penentuan Paket Makanan Hemat Di Rm Roso Echo Dengan Algoritma Apriori," *Komputa J. Ilm. Komput. dan Inform.*, vol. 3, no. 2, pp. 96–101, 2014, doi: 10.34010/komputa.v3i2.2398.

[5] H. Li, Y. Wang, D. Zhang, M. Zhang, and E. Y. Chang, "PFP: Parallel FP-growth for query recommendation," *RecSys'08 Proc. 2008 ACM Conf. Recomm. Syst.*, pp. 107–114, 2008, doi: 10.1145/1454008.1454027.

[6] P. Resnick and H. R. Varian, *Recommender Systems*, vol. 40, no. 3. 1997.

[7] S. Agarwal, *Data mining: Data mining concepts and techniques*. 2014.

[8] J. W. G. Putra, "Pengenalan Konsep Pembelajaran Mesin dan Deep Learning," *Comput. Linguist. Nat. Lang. Process. Lab.*, vol. 4, pp. 1–235, 2019, [Online]. Available: https://www.researchgate.net/publication/323700644.

[9] B. Boehmke and B. Greenwell, *Hands-On*

Machine Learning with R. 2019.
[10] J. Heaton, *Introduction to Neural Networks for Java*, vol. 99. 2008.
[11] S. Paul and A. Singh, *Hands-On Python Deep Learning for the Web*. 2020.
[12] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, and F. Ricci, *Recommender Systems Handbook*. 2011.
[13] T. Kavzoglu, "Increasing the accuracy of neural network classification using refined training data," *Environ. Model. Softw.*, vol. 24, no. 7, pp. 850–858, 2009, doi: 10.1016/j.envsoft.2008.11.012.