Sistem Rekomendasi Item Pada Game Dota 2 dengan Multilayer Perceptron Neural Network

Vincentius Leonardo [1], Leo Willyanto Santoso [2], Alvin Nathaniel Tjondrowiguno [3]
Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jl. Siwalankerto 121-131, Surabaya 60236
Telp (031) – 2983455, Fax. (031) - 8417658
vincentiusleo12@gmail.com[1], leow@petra.ac.id[2], alvin.tjondrowiguno@gmail.com [3]

ABSTRAK

Dota 2 adalah salah satu *online game* bergenre *Multiplayer Online Battle Arena* yang paling populer di dunia. *Dota 2* dimainkan antara dua tim berisi 5 orang, di mana masing - masing tim harus saling menghancurkan markas satu sama lain untuk menang. Pada awal permainan, tiap pemain memilih satu karakter yang disebut "*hero*" yang masing - masing memiliki kemampuan unik. Dalam permainan ini, para pemain bertujuan untuk mengumpulkan *resource* berupa *experience* dan *gold* yang akan digunakan untuk membantu menghancurkan markas dari tim lawan. *Gold* digunakan untuk membeli *item* yang berfungsi untuk memperkuat *hero* pemain masing – masing. Pada *Dota 2* versi 7.16, terdapat lebih dari 150 jenis *item* unik dengan efek yang beragam, sementara setiap pemain hanya bisa menggunakan 6 *item* secara aktif.

Kombinasi *item* yang optimal tentunya sangat berpengaruh dalam mencapai kemenangan dalam permainan *Dota 2.* Menentukan kombinasi item yang paling optimal seringkali tidak mudah, baik bagi pemula maupun pemain berpengalaman. Oleh karena itu dibuatlah sistem rekomendasi untuk membantu pemain menentukan pilihan *item.* Ada beberapa penelitian sebelumnya terkait Dota 2 seperti sistem rekomendasi *hero* dan juga sistem prediksi *hero* pada saat picking phase. Sistem rekomendasi *hero* tersebut memiliki kekurangan di mana sistem tersebut kurang diuji, sementara sistem prediksi *hero* tersebut memiliki kekurangan yaitu akurasi yang rendah.

Pada skripsi ini dibuat sebuah sistem rekomendasi pemilihan *item* dalam permainan *Dota 2.* Rekomendasi dihasilkan berdasarkan *association rule* yang dicari menggunakan algoritma *Apriori* menggunakan ribuan *match data Dota 2.* Untuk mengevaluasi kualitas rekomendasi yang dihasilkan, telah dilatih sebuah *neural network* jenis *multilayer perceptron* yang mampu menentukan persentase kemenangan berdasarkan pilihan item dengan akurasi sebesar 73,04%. Hasil dari pengujian terhadap sistem rekomendasi adalah rata – rata *win rate* sebesar 82,97%.

Kata Kunci:

Dota 2, *Apriori, Association Rule, Multilayer Perceptron*

ABSTRACT

Dota 2 is a Multiplayer Online Battle Arena (MOBA) game which is currently one of the most popular in the world. Dota 2 is played between two teams of 5 players, where each team must destroy each other's bases in order to win. At the start of the game, each player picks their character called a "hero" which has various unique abilities. In this game, the players collect resources such as experience and gold which purpose is to assist them in destroying the opponent's base. Gold is used to purchase items to make the heroes stronger. In Dota 2 7.16 version, there are more than 150 items with various unique effects available, but players are only allowed to use 6 items.

An optimal combination of items plays a huge factor in winning the game. Deciding the most optimal combination of items is often difficult, whether for beginners or experienced players. Therefore, a recommendation system to help players choose their items is made. There are several previous researches in Dota 2, such as a hero recommendation system and hero pick prediction system. The hero recommendation system lacks in the testing, meanwhile the hero pick prediction system lacks in its' accuracy.

With this thesis a recommendation system for those items is made. The recommendation is based on association rules which are found using apriori algorithm by analyzing thousands of Dota 2 matches. To evaluate the quality of the recommendation, a multilayer perceptron neural network which can predict the chance of winning through the choice of items is trained. The neural network is able to predict accurately with up to 73.04% accuracy. The result of the test on the recommendation system is an average of win rate of 82.97%.

Keywords:

Dota 2, Apriori, Association Rule, Multilayer Perceptron

1. PENDAHULUAN

Dota 2 adalah suatu *game* bergenre *Multiplayer Online Battle Arena* yang dikembangkan oleh *Valve Corporation*. *Dota 2* merupakan salah satu *game* yang terkenal susah untuk dipelajari dan memiliki sistem yang kompleks. *Dota 2* dimainkan antara dua tim berisi 5 orang, di mana masing - masing tim memiliki markas yang harus mereka lindungi pada arena permainan. Pada awal permainan, tiap pemain memilih satu karakter yang disebut "*hero*" yang masing - masing memiliki *skill* unik masing - masing. Dalam permainan ini, para pemain bertujuan untuk mengumpulkan *resource* berupa *experience* dan *gold* yang akan digunakan untuk membantu menghancurkan markas dari tim lawan. *Experience* diperlukan agar hero yang dimainkan dapat memperoleh level sehingga dapat memperkuat skillnya. *Gold* diperlukan agar pemain dapat membeli *item* yang dapat digunakan untuk memperkuat *hero*-nya. Permainan berakhir jika bangunan besar pada salah satu markas yang disebut *ancient* berhasil dihancurkan.

Dota 2 merupakan salah satu *online game* dengan *playerbase* terbanyak di dunia. Diambil dari *steamdb.info* pada 29 Mei 2018, *Dota 2* memiliki rata - rata pemain tiap harinya mencapai sejumlah 480.000 pemain. *Dota 2* tentu saja sebagai salah satu *online game* terbesar pada *platform PC* dikompetisikan secara profesional. Ajang kompetisi terbesar dari *Dota 2* adalah *The International* yang diadakan setiap bulan Agustus. Hadiah yang

diberikan bagi peserta *The International* tidaklah sedikit. Total hadiah *The International 7* mencapai sebesar $24.787.916.

Kompleksitas *Dota 2* dimulai dari awal permainan, yaitu pemilihan *hero* (*picking phase*). Setiap pemain secara bergiliran harus memilih masing - masing *hero* untuk dimainkan dalam kurun waktu yang sudah ditentukan. Namun, untuk dapat mencapai kemenangan dengan lebih mudah, *player* harus memahami kegunaan dan karakteristik masing - masing *hero* sehingga *player* dapat mengerti kapan situasi yang tepat untuk memilih suatu *hero* tertentu. Permasalahannya adalah sejauh ini, pada Dota 2 terdapat 115 *hero* berbeda dengan kemampuan yang unik. Dari sebab itu tidak mudah untuk memilih *hero* yang benar - benar ideal dalam permainan ini. Ada banyak sekali faktor yang harus dipertimbangkan dalam memilih *hero*, antara lain adalah peran, sinergi dengan tim, tingkat kesulitan penggunaan, keunggulan dan kelemahan dibandingkan *hero* yang dipilih oleh lawan, dan sebagainya. Begitu banyak faktor yang membuat permainan ini rumit padahal permainan ini baru saja dimulai.

Kompleksitas berikutnya yang akan dihadapi adalah pemilihan *item* yang akan dibeli oleh *player*. Setiap *player* dapat memiliki 9 *item slot* di mana terbagi menjadi dua macam penyimpanan yaitu *inventory* dan *backpack*. *Inventory* adalah *item* yang efeknya sedang diaktifkan untuk *hero* sementara *backpack* adalah sekedar tempat penyimpanan di mana efek dari *item* dinon-aktifkan. *Inventory* terdiri dari 6 *item slot* sementara *backpack* terdiri dari 3 *item slot*. Pada Dota 2 versi 7.16, terdapat lebih dari 150 jenis *item* berbeda.

Selain dengan cara membeli, ada beberapa *item* yang hanya dapat diperoleh dengan cara menggabungkan beberapa *item* tertentu. *Item* - *item* tersebut tentunya memiliki efek yang berbeda - beda dan unik dengan fungsinya masing - masing. Akibatnya, memilih *item* yang optimal dalam satu permainan tentunya tidak akan kalah sulit dengan memilih *hero*. Terdapat juga banyak faktor yang dipertimbangkan dalam memilih *item* seperti kegunaannya, kecocokan dengan *hero* yang dimainkan, apakah situasi dan *timing* tepat untuk membeli *item* tersebut, dan sebagainya. Pemilihan *item* dapat menentukan kemenangan atau kekalahan dalam sebuah permainan Dota 2 [2].

Skripsi ini akan berfokus pada proses pemilihan *hero* dan *item* tersebut. Pada skripsi ini akan dibuat suatu sistem yang dapat merekomendasikan *hero* dan *item* apa saja yang sebaiknya diperoleh oleh *hero* tersebut berdasarkan preferensi *player*. Hal ini dilakukan dengan cara menginputkan parameter seperti *hero* yang dipilih dalam tim lawan.

Metode yang akan digunakan dalam skripsi ini adalah *association rule mining* dengan algoritma Apriori untuk melakukan *data mining* dan *Multilayer Perceptron Neural Network* untuk sistem prediksi kemenangan yang berfungsi untuk menilai apakah *hero* dan *item* yang dipilih oleh user dengan bantuan rekomendasi oleh sistem.

Association rule digunakan karena dianggap sesuai untuk mencari informasi asosiasi antar *hero* dan *item* yang ada dalam data pertandingan yang berjumlah besar. *Multilayer Perceptron Neural Network* digunakan karena pada penelitian yang sebelumnya tentang sistem rekomendasi pemilihan *hero* juga menyertakan sistem untuk memprediksi kemenangan menggunakan *Multilayer Perceptron* berdasarkan hero yang direkomendasikan dengan akurasi yang sudah baik, yaitu sebesar 88,63% [3]. Pada penelitan tersebut, peneliti menyatakan kekurangan terletak pada pengujian. Sebaiknya dilakukan pengujian secara kualitatif untuk pengguna dengan cara mengisi form dan sebagainya.

2. LANDASAN TEORI

2.1 Association Rule

Association Rule atau sering disebut juga *Market Basket Analysis* (Analisa Keranjang Pasar) adalah suatu metode *data mining* yang bertujuan untuk mencari sekumpulan *items* yang sering muncul bersamaan. Pada umumnya, *association rule* ini dianalogikan dengan keranjang belanjaan. Dari keranjang belanjaan para pengunjung supermarket akan dapat diketahui, barang apa saja yang sering dibeli bersamaan dan barang mana saja yang tidak. *Association rule* umumnya mengambil bentuk IF-THEN yang menggabungkan beberapa items menjadi satu, misalnya IF A and B THEN C.

Secara teori, beberapa hal yang digunakan untuk mengukur apakah sekumpulan *items* (*an item set*) sering muncul bersamaan atau tidak, adalah *support of an item set*, *confidence of an association rule*, dan beberapa r*ule selection methods* [1].

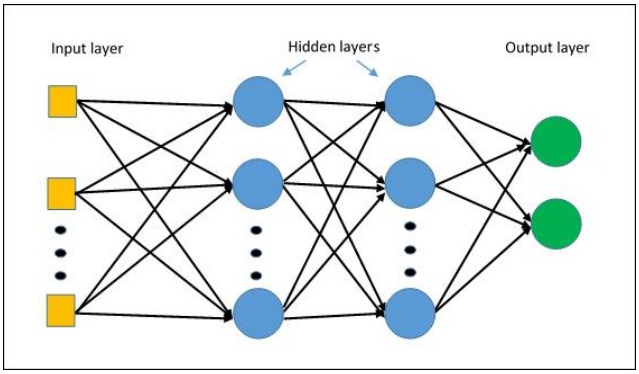
2.2 Apriori Algorithm

Algoritma *Apriori* digunakan untuk mengurangi jumlah pencarian *frequent itemset* yang dilakukan untuk menemukan *association rules*. Algoritma ini melakukan pencarian pada *database* untuk mencari *itemset* yang sering muncul di mana k-*itemsets* digunakan untuk generasi k+1-*itemsets* [11].

Setiap k-*itemset* harus sama atau lebih besar dari *support threshold* minimum untuk menjadi frekuensi. Selain itu, disebut *candidate itemsets*. Awalnya, algoritma ini memindai *database* untuk mencari frekuensi dari 1-*itemsets* yang hanya berisi satu *item* dengan menghitung setiap *item* dalam *database*. Frekuensi dari 1-*itemsets* digunakan unutk mencari *itemsets* dalam 2-*itemsets* yang nantinya akan digunakan untuk mencari 3-*itemsets* dan seterusnya. Jika sebuah *itemset* tidak termasuk *frequent*, subset sebesar apapun juga tidak termasuk *frequent* [4].

2.3 Multilayer Perceptron Neural Network

Sebuah *Multilayer Perceptron Neural Network* (MLP) adalah suatu klasifikasi dari *Feedforward Artificial Neural Network*. Sebuah MLP terdiri dari minimal 3 *layer node*, yaitu satu *input layer,* minimal satu *hidden layer,* dan satu *output layer.* Gambaran sebuah *multilayer perceptron neural network* dapat dilihat pada Gambar 1.



Gambar 1. Sebuah gambaran Multilayer Perceptron Neural Network

MLP menggunakan sebuah teknik *supervised learning* yang disebut *backpropagation* untuk *training*-nya. *Backpropagation* tersebut digunakan untuk mencari seberapa besar *error gradient* yang terjadi sehingga MLP dapat menyesuaikan weight antar

neuron-nya sesuai dengan *error gradient* tersebut. *Layer*-nya yang banyak dan aktivasi non-linearnya membedakan MLP dari *perceptron* linear. Lapisan input menerima sinyal dari luar, kemudian melewatkannya ke lapisan tersembunyi pertama, yang akan diteruskan sehingga akhirnya mencapai lapisan output [6].

2.4 MLXtend

MLXtend adalah suatu *library Python* yang berisi ekstensi dan modul bantuan untuk analisis data dan *library machine learning* lainnya [5]. Pada skripsi ini modul dari *MLXtend* yang digunakan adalah modul *frequent patterns* yang berfungsi untuk menerapkan algoritma *apriori* dan *association rules*.

2.5 TensorFlow

TensorFlow adalah suatu *library open source* yang berfungsi untuk *high performance numerical computation*. Arsitekturnya yang fleksibel memungkinkan komputasi di berbagai *platform*, dari *desktop*, *server*, hingga *mobile device*. Awalnya *TensorFlow* dikembangkan oleh peneliti dari *Google*. *TensorFlow* menyediakan banyak bantuan untuk *machine learning*, *deep learning*, dan berbagai fungsi komputasi lainnya yang digunakan di bidang penelitian sains [9].

2.6 Keras

Keras adalah suatu *API* untuk membuat dan melatih model *deep learning* dan *neural network*. *Keras* digunakan untuk eksperimentasi cepat, penelitian tingkat lanjut, dan produksi dengan beberapa keunggulan seperti *user-friendly*, bersifat modular, dan mudah untuk dikembangkan. *Keras* sangat sederhana, memiliki *interface* yang cukup umum. *Keras* menyediakan *feedback* yang jelas apabila terjadi *error* [10].

2.7 Tinjauan Studi

Sebagai salah satu tren baru dalam penelitian, sudah ada banyak penelitian yang dilakukan berkaitan dengan Dota 2. Ada penelitian yang telah dilakukan dengan judul "*Draft-Analysis of the Ancients: Predicting Draft Picks in DotA2 Using Machine Learning*". Penelitian ini membahas tentang sebuah sistem untuk memprediksi pemilihan *hero* (*hero drafting*) dan mengevaluasinya dengan cara membandingkan kinerja sistem tersebut dengan pemain ahli. Metode yang digunakan adalah *Bayes Nets* dan *Long Short-Term Memory Recurrent Neural Networks* (LSTM RNNs). Dari hasil penelitian tersebut diperoleh akurasi sebesar 11,94% dalam memprediksi pilihan *hero* dari seluruh *test data* [8].

Penelitian lainnya adalah penelitian yang berjudul "*A Recommender System for Hero Line-Ups in MOBA Games*". Penelitian ini membahas tentang sebuah sistem untuk merekomendasi pilihan *hero* dalam Dota 2. Metode yang digunakan adalah Multilayer Perceptron Neural Network untuk memprediksi hasil pertandingan berdasarkan *hero* yang telah dipilih. Hasil penelitian menghasilkan sebuah sistem rekomendasi di mana telah teruji dari 1000 kali percobaan, sebesar 74,9% pertandingan dimenangkan apabila mengikuti rekomendasi yang dimunculkan. Selain itu juga dihasilkan sistem prediksi pertandingan berdasarkan *hero* dengan akurasi sebesar 88,63% [3].

Penelitian lain berjudul "*Esports Analytics Through Encounter Detection*" di mana dilakukan pendeteksian terhadap titik temu dua tim (*encounters*), kemudian dilakukan prediksi kemenangan berdasarkan data tersebut. Metode yang digunakan adalah *logistic regression*. Hasil penelitian ini menyatakan bahwa analisis berdasarkan *encounter* dapat menguraikan dinamik permainan Dota yang kompleks menjadi komponen - komponen teratur. Teknik analisis ini dapat diterapkan pada permainan *esport* kompetitif berbasis tim lainnya [7].

3. ANALISIS dan DESAIN SISTEM

3.1 Asumsi Dasar

Pada skripsi ini diasumsikan bahwa kemenangan individu sama dengan kemenangan tim. Hal ini berarti dengan melakukan pengolahan data terhadap *hero* secara individu dalam sebuah tim dianggap apabila *hero* tersebut menang, berarti tim yang diikuti oleh *hero* tersebut juga menang.

3.2 Kriteria Data

Berikut adalah kriteria data yang digunakan dalam skripsi ini:

1. Data yang digunakan adalah *match* data yang diambil dari *Web API Dota 2* yang disediakan oleh *Valve* dengan ketentuan sebagai berikut:
   a. Data pertandingan di mana tidak ada pemain yang meninggalkan permainan sebelum berakhir (*abandon*).
   b. Data yang diambil dari setiap *match* adalah data *hero* dan *item* yang dimiliki masing – masing *hero* tersebut setelah permainan berakhir.
2. Data *item* yang dapat diambil dalam skripsi ini adalah item yang pernah menjadi *item* tingkat paling akhir atau yang sekarang menjadi *item* tingkat paling akhir, yaitu item yang tidak dapat di-*upgrade* lebih lanjut dan tidak bersifat *consumable* (dapat habis ketika digunakan). Dengan kriteria di atas, telah diambil sejumlah 76 *item*

3.3 Desain Aplikasi

Aplikasi ini bertujuan untuk membantu *player Dota 2* dalam memilih *hero* dan *item.* Hal ini dilakukan dengan cara mengolah *match data* yang sudah dikumpulkan untuk memperoleh *association rule* yang akan dijadikan dasar dari rekomendasi. Aplikasi ini dibuat dengan beberapa fitur utama diantaranya adalah:

1. Rekomendasi *Hero* berdasarkan data *history* permainan *user*.
2. *Generate Association Rule* dari *frequent itemsets* yang disimpan dalam format *CSV* secara terpisah untuk tiap *hero*.
3. Rekomendasi *Item* berdasarkan *Assocation Rule* yang sudah dihasilkan.
4. Memprediksi persentase kemenangan berdasarkan pilihan *hero* dan *item*.

Diharapkan *player* yang menggunakan aplikasi ini baik pemula ataupun sudah berpengalaman dalam *Dota 2* dapat mempelajari dan mengeksplorasi lebih lanjut mengenai pemilihan *item* dalam *Dota 2.*

3.4 Desain Sistem Rekomendasi

Sistem ini dibuat dengan salah satu fungsi utamanya adalah rekomendasi *item* dengan ketentuan – ketentuan sebagai berikut:

1. *Item* yang sudah dipilih tidak direkomendasikan lagi, tetapi tetap bisa dipilih. Hal ini dikarenakan pada umumnya, di *Dota 2* memiliki item akhir yang sama kurang disarankan karena sebagian besar efek dari item yang sama tidak menumpuk, hanya salah satu saja yang aktif.
2. *Rule* yang diambil sebagai dasar rekomendasi adalah *rule* dengan *consequent* berupa item atau sekumpulan item.
3. Saat suatu *item* direkomendasikan, *rules* yang menghasilkan rekomendasi tersebut akan ditampilkan apabila *user* meng-*hover cursor*-nya pada *button* untuk *item* tersebut. *Metric* yang ditampilkan adalah *antecedent* dan *confidence* dari suatu *rule*.

3.5 Desain Neural Network

Neural Network yang akan digunakan adalah jenis *Multilayer Perceptron* dengan *layer* sejumlah 3, satu *input layer*, satu *hidden layer*, dan satu *output layer*. *Optimization function* yang digunakan adalah *RMSprop optimizer algorithm* dan *loss function* yang digunakan adalah *binary crossentropy*. *Metric* yang akan diukur adalah *accuracy*.

Jumlah *neuron* pada *input layer* adalah 198, di mana 122 *neuron* pertama merupakan representasi untuk *hero* sesuai dengan IDnya dan 76 sisanya merupakan representasi untuk *item*. *Neural Network Keras Tensorflow* menerima data dengan tipe *numpy array* sebagai inputnya, dengan ketentuan satu *neuron* sama dengan satu indeks *array*.

Untuk 122 *neuron* pertama, ketentuan inputnya adalah *array* indeks ke sekian akan bernilai 1 apabila indeks tersebut merupakan ID *hero* pilihan *user*. *Array* indeks ke sekian akan bernilai -1 apabila indeks tersebut merupakan ID *hero* di tim lawan. Selain itu nilainya adalah 0.

Untuk 76 *neuron* berikutnya, ketentuan inputnya adalah *array* indeks ke sekian nilainya akan ditambah satu apabila *user* memilih *item* dengan ID sekian ditambah 122. Apabila *user* memilih *item* yang sama maka nilai *array* dengan indeks tersebut menjadi 2, dan seterusnya. Selain itu nilainya adalah 0.

Activation function yang digunakan pada *input layer* adalah *activation function* jenis *sigmoid*.

Pada skripsi ini, jumlah *hidden layer* yang digunakan adalah 1, dengan jumlah *neuron*-nya merupakan salah satu parameter yang diujikan. *Activation function* yang digunakan juga merupakan salah satu parameter yang diujikan.

Output layer terdiri dari satu *neuron*, berarti *neural network* ini mengoutputkan satu angka, yaitu persentase kemenangan berdasarkan input yang sudah dimasukkan dengan *range* antara 0 – 1. *Activation function* yang digunakan adalah *sigmoid*.

4. PENGUJIAN SISTEM

4.1 Pengujian Neural Network

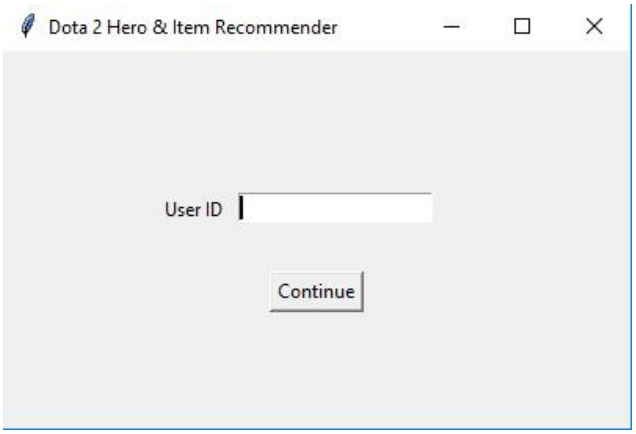
Untuk menghasilkan *neural network* dengan performa terbaik, maka dilakukan berbagai pengujian agar dapat menemukan *neural network* yang memiliki *accuracy* setinggi mungkin. Oleh karena itu dilakukan *training* dan *testing* dengan berbagai parameter. Data *match* yang digunakan berjumlah 2441. Karena di dalam setiap *match* terdapat 10 *hero*, maka setiap *hero* tersebut dapat dijadikan input untuk *neural network*. Dari situ didapat total data berjumlah 24410 input. Kemudian data tersebut dibagi untuk *training* dan *testing*. Pada skripsi ini digunakan 90% dari keseluruhan data untuk *training* dan 10% sisanya untuk *testing*.

Untuk pengujian tiap parameter, dilakukan *training* dan *testing* sejumlah 150 *epoch*, dengan pengecualian kepada beberapa pengujian dengan parameter *learning rate* 0.0001 dikhususkan *epoch* berjumlah 500 karena hasil *testing* belum menunjukkan tren menurun pada *epoch* 150. Ada 16 arsitektur *neural network* yang diujikan.

Dari pengujian yang telah dilakukan, diperoleh bahwa *testing neural network* dengan akurasi tertinggi terjadi pada pengujian 15 dengan akurasi 73,04% pada *epoch* ke 49 dengan parameter 1024 *hidden layer neuron*, *activation function hidden layer ReLU* dan *learning rate* 0.0001. Maka model tersebutlah yang akan digunakan untuk aplikasi pada skripsi ini.Tetapi, dapat juga diperoleh bahwa akurasi tertinggi tiap pengujian tidak berbeda terlalu jauh. Hal lain yang dapat ditafsirkan dari tabel di atas adalah *neural network* dengan *activation function ReLU* rata – rata lebih cepat mencapai akurasi puncak.

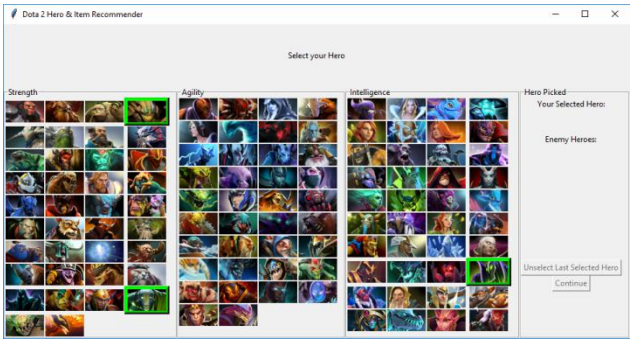
4.2 Simulasi Pengujian Aplikasi

Pada bagian ini dilakukan sebuah simulasi untuk menguji berjalannya aplikasi. Hal ini dilakukan untuk menguji apakah aplikasi sudah berjalan dengan seharusnya. Langkah pertama yang dilakukan oleh *user* adalah memasukkan sebuah nomor ID *Dota 2* pada *window* yang dapat dilihat pada Gambar 2. Di sini terdapat beberapa *error handling*, diantaranya adalah *error handling* untuk *user* ID tidak ditemukan, *error handling* terlalu banyak *request* atau *API* sedang *down*, dan juga *error handling* untuk *API key* yang tidak valid.



Gambar 2. *Window Input ID*

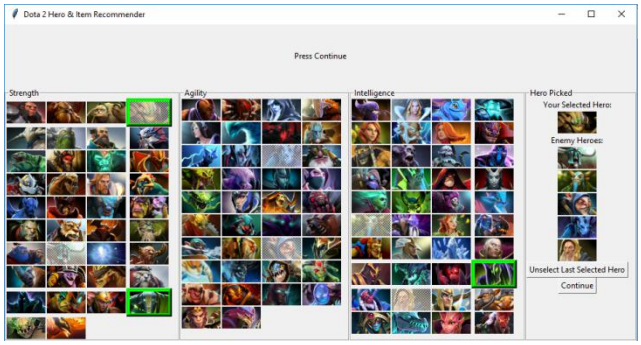
Apabila *user* sudah memasukkan sebuah *ID* dengan benar, maka selanjutnya akan ditampilkan *window* pemilihan *hero* seperti pada Gambar 3. Pada *window* ini dikeluarkan sebuah rekomendasi *hero* berdasarkan data yang sudah dimasukkan oleh user pada *window* sebelumnya. Aplikasi akan mengakses *API* dan mengakses *history match* dari akun dengan *ID* yang sudah dimasukkan dan mencari 3 *hero* yang paling banyak menang dalam 100 *match* terbaru yang dimainkan oleh akun tersebut. *Hero* yang direkomendasikan oleh sistem diberi tanda *border* berwarna hijau. Apabila *user* tidak memasukkan *ID* pada *window* sebelumnya, aplikasi akan tetap berjalan, namun tidak mengeluarkan rekomendasi *hero* apapun.



Gambar 3. *Window* pemilihan *hero* dengan adanya rekomendasi

Pada *window* ini, *user* harus memilih 6 *hero* di mana *hero* pertama adalah *hero* yang akan disimulasikan pemilihan *item*-

nya dan 5 sisanya adalah *hero* pada tim musuh. *Hero* yang sama tidak dapat dipilih kembali dan *button* untuk *hero* tersebut akan menjadi *disabled* apabila suatu *hero* dipilih. *User* dapat melihat nama *hero* dengan cara meng-*hover cursor* pada *button* sebuah *hero*. Apabila user ingin membatalkan pilihan hero, user dapat menekan *button* "*Unselect Last Selected Hero*" yang terletak di bagian kanan bawah untuk membatalkan pilihan terakhir yang dilakukan. Setelah *user* selesai memilih 6 *hero* seperti pada Gambar 4, maka *user* dapat menekan tombol *continue* untuk lanjut pada tahap berikutnya. Button *continue* akan tetap berada dalam status *disabled* selama *user* belum memilih 6 *hero*.



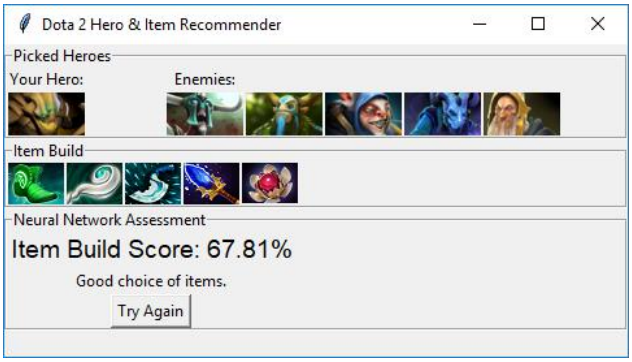
Gambar 4. *Window* pemilihan *hero* dengan 6 *hero* sudah dipilih

Setelah *button continue* ditekan, maka aplikasi akan memasuki tahap berikutnya, yaitu tahap pemilihan *item*. Pada tahap ini user dapat memilih *item* untuk *hero* yang sudah dipilih pada tahap sebelumnya. *Item* yang dipilih dapat berjumlah maksimal 6 *item*. Item yang direkomendasikan oleh sistem akan diberikan *border* berwarna hijau di sekitar *button item* tersebut. Setelah user memilih sebuah *item,* aplikasi akan mencari *rule* yang berkaitan dengan *item* - *item* yang sudah dipilih oleh user. Karena *rule* ditemukan maka sistem menampilkan beberapa rekomendasi

Gambar 5. Aplikasi menampilkan *rules* terkait *item* yang direkomendasikan

Apabila *user* meng-*hover cursor* pada item yang direkomendasikan, maka nama *item* dan *rule* yang menghasilkan rekomendasi tersebut akan ditampilkan. Informasi mengenai *rule* yang ditampilkan adalah nomor rule, antecedent, dan persentase confidence dari rule tersebut, seperti yang dapat dilihat pada Gambar 5. Apabila *user* meng-*hover cursor* pada item yang tidak direkomendasikan, maka hanya nama *item* tersebut saja yang ditampilkan.

Setelah *user* selesai memilih *item* sesuai keinginannya, langkah selanjutnya adalah menekan *button continue* di bagian sebelah kanan *window*. Tahap selanjutnya adalah penilaian atau prediksi persentase kemenangan oleh *neural network* berdasarkan pilihan - pilihan *hero* dan *item* yang sudah dimasukkan oleh *user.* Aplikasi akan menampilkan sebuah angka persentase yang menyatakan kemungkinan menang menurut *neural network* yang sudah diimplementasikan. Apabila *user* ingin mencoba memasukkan input kembali dari awal, maka *user* dapat menekan tombol *try again. Window* ini dapat dilihat pada Gambar 6.



Gambar 6. Tampilan *window* prediksi oleh *neural network*

Secara keseluruhan, aplikasi sudah berjalan dengan seharusnya. Aplikasi dapat mengakses data dari *API* dan menghasilkan rekomendasi *hero*. Aplikasi juga sudah dapat menghasilkan rekomendasi *item* berdasarkan pilihan *hero* dan *item* yang dilakukan oleh *user* dengan cara mencari *rules* terkait pilihan - pilihan tersebut. *Neural network* yang dimplementasikan pada aplikasi dapat berfungsi dengan seharusnya dan mengeluarkan sebuah persentase. Pada aplikasi tidak ditemukan *bug* apapun.

4.3 Pengujian Sistem Rekomendasi Secara Teori

4.3.1 Teknis Pengujian

Pengujian terhadap rekomendasi dilakukan dengan cara membandingkan antara pemilihan *item* menggunakan rekomendasi dan pemilihan *item* secara *random* sepenuhnya. Langkah pengujian adalah sebagai berikut:

1. 6 *hero* dipilih secara *random*.
2. Cari *rule* yang memenuhi syarat, kemudian *consequent* dimasukkan ke dalam sebuah *array* yang berisi *item* rekomendasi.
3. Kemudian dari *array* berisi *item* yang direkomendasikan tersebut dipilih satu *item* secara random. Apabila array kosong, maka dipilih *item random*.
4. Langkah 2 dan 3 diulang hingga diperoleh 6 *item*.
5. Kemudian hasil pilihan *hero* dan *item* dikonversi agar sesuai dengan input *neural network* dan *neural network* mengeluarkan outputnya.

6. Selanjutnya dengan *hero* yang sama, dipilih 6 item secara *random* sepenuhnya tanpa mengikuti rekomendasi, dan diinputkan pada *neural network*.

Langkah pengujian di atas diiterasikan sebanyak 1000 kali untuk setiap parameter yang diujikan sehingga dihasilkan perbandingan dua jenis output, yaitu output *neural network* dengan input menggunakan rekomendasi dan input tanpa rekomendasi. Dilakukan pengujian rekomendasi dengan generasi *rule* dengan *confidence threshold* 20%, 50%, 80%, dan 100%.

4.3.2 Hasil Pengujian

Hasil pengujian sistem rekomendasi dapat dilihat pada Tabel 1.

Tabel 1. Hasil Pengujian Sistem Rekomendasi

Rule Confidence Threshold	Rules Found	Tests	Average Win Rate	Difference
20%	184517	With Recom.	0.8071	-0.0341
		Without Recom.	0.8412	
50%	122011	With Recom.	0.8099	-0.0344
		Without Recom.	0.8444	
80%	47785	**With Recom.**	**0.8297**	-0.0159
		Without Recom.	0.8456	
100%	34340	With Recom.	0.8291	**-0.0070**
		Without Recom.	0.8360	

Dari Tabel 1 telah diperoleh bahwa rata - rata *win rate* tertinggi dicapai dengan *confidence threshold* sebesar 80% sebesar 82,97%. Namun jika melihat selisih antara hasil pengujian menggunakan rekomendasi dan tanpa rekomendasi, hasil terbaik adalah dengan *confidence threshold* 100% dengan rata - rata *win rate* yang tidak jauh berbeda dengan *win rate* tertinggi, yaitu 82,91% dengan selisih 0,6952% dibandingkan pengujian tanpa rekomendasi. Sementara *threshold* 80% memiliki selisih -1,5882% dibandingkan pengujian tanpa rekomendasi.

4.3.3 Sebab Akibat

Melalui tabel hasil pengujian rekomendasi *item*, dapat diperoleh bahwa secara keseluruhan *win rate* dengan rekomendasi lebih rendah dibandingkan secara *random*. Secara keseluruhan semakin rendah *threshold confidence* maka *win rate* dengan rekomendasi semakin rendah. Hal ini dikarenakan adanya *rule* dengan *confidence* yang rendah. Karena *rule* berasal dari *itemset* di mana suatu *hero* menang, *confidence rule* yang rendah berarti kemenangan di mana *rule* tersebut berlaku relatif lebih sedikit terjadi.

Hal lain yang diperoleh dari tabel di atas adalah semakin tinggi *threshold confidence* maka *rule* yang dihasilkan semakin sedikit. *Rule* yang tersisa adalah rule dengan kualitas yang baik (nilai *confidence* tinggi) sehingga kemungkinan menang lebih tinggi. Mungkin *win rate* dengan rekomendasi akan melebihi *win rate* tanpa rekomendasi apabila jumlah iterasi pengujian ditambah, namun pada pengujian ini akan sulit untuk melebihi *win rate* tanpa rekomendasi karena apabila *rule* sedikit, pemilihan *item* secara *random* akan lebih sering terjadi. Akibatnya, perbedaan *win rate* tidak akan terlalu jauh karena sebagian besar dari proses pemilihan *item* adalah *random*.

Faktor lain terkait *rule* adalah kurangnya data yang diujikan sehingga *rule* yang dihasilkan kurang optimal. Pembagian *dataset* untuk tiap *hero* tidak merata karena diambil secara random, sehingga mengakibatkan kualitas rule untuk beberapa *hero* tidak optimal. Akibatnya rekomendasi yang dihasilkan tidak mengarahkan kepada kemungkinan menang yang lebih tinggi.

Faktor lain yang mungkin mempengaruhi hasil pengujian di atas adalah performa *neural network* itu sendiri. *Neural network* yang digunakan hanya memiliki *testing accuracy* sebesar 73,04%. Hal ini menunjukkan bahwa prediksi oleh *neural network* tidak jauh dari kesalahan, sehingga menjadi salah satu faktor yang mengakibatkan hasil yang demikian.

4.4 Pengujian Sistem Rekomendasi Secara *Real*

Untuk menguji sistem rekomendasi lebih lanjut, maka dilakukan pengujian secara praktek atau *real*. Pada pengujian ini, dimainkan antara dua tim yang berisi hero - hero dengan nilai prediksi oleh neural network tinggi dan hero - hero dengan nilai prediksi oleh neural network rendah. Semua hero memilih itemnya mengikuti rekomendasi. Diambil 10 *hero* secara random yang masing - masing dimasukkan pada aplikasi dengan musuh random dan dipilih 6 item secara random sepenuhnya mengikuti rekomendasi. Kemudian 10 *hero* tersebut diurutkan berdasarkan hasil prediksi oleh *neural network*. Selanjutnya, 10 *hero* tersebut dibagi menjadi dua tim, 5 tertinggi dan 5 terendah. Tim pertama diberi nama Tim A, tim kedua Tim B.

Selanjutnya dengan hero yang sudah dipilih dimasukkan kembali pada aplikasi untuk diprediksi, kali ini dengan musuh sesuai dengan pembagian yang telah dilakukan. Kemudian kedua kelompok tersebut ditandingkan. Pengujian ini dilakukan dengan beberapa asumsi, yaitu:

1. Semua pemain dianggap setara dan konstan sehingga diasumsikan faktor pemain tidak mempengaruhi pengujian.

2. Faktor *in-game* lainnya dianggap tidak mempengaruhi hasil pengujian.

Hasil dari pengujian ini adalah dari 10 kali bertanding, tim A menang sebanyak 7 kali.

5. KESIMPULAN

Dari hasil pengujian dan pembuatan sistem dapat diambil beberapa kesimpulan antara lain:

- Akurasi tertinggi yang dicapai oleh *Multilayer Perceptron Neural Network* dalam memprediksi kemenangan sesuai dengan *input* dari *user* adalah sebesar 73,04%.
- Model *Neural Network* dengan akurasi tertinggi dicapai pada pengujian 15 dengan parameter *neuron hidden layer* 1024, *learning rate* 0,0001, dan *activation function hidden layer ReLU* pada *epoch* 49.
- Pengujian terhadap sistem rekomendasi dilakukan dengan cara membandingkan hasil prediksi antara pemilihan *item* mengikuti rekomendasi dan pemilihan *item* secara *random*

menggunakan pilihan *hero* yang sama. Parameter yang diujikan adalah *confidence threshold*. Rata - rata *win rate* sebesar 82,97% adalah nilai tertinggi yang tercapai dengan *confidence threshold* 80%. Sementara itu selisih terbaik dicapai dengan *confidence threshold* 100%, sebesar -0,6952% sementara rata - rata *win rate*-nya adalah 82,91%.

- *Neural Network* dengan *activation function* jenis *ReLU* pada *hidden layer*-nya cenderung lebih cepat mempelajari data dibandingkan dengan *activation function sigmoid*.
- Parameter yang diujikan untuk *neural network* dalam skripsi ini seperti jumlah *neuron*, *learning rate*, dan *activation function* kurang mempengaruhi dalam optimalisasi *testing accuracy,* namun sangat mempengaruhi seberapa cepat *peak accuracy* tercapai.

Saran untuk menyempurnakan dan mengembangkan sistem ini lebih lanjut:

- Penambahan jumlah data, baik untuk *neural network* maupun untuk *association rule*.
- Pengembangan *interface* aplikasi agar lebih interaktif dengan *user*.
- Pengujian dengan cara lain untuk sistem rekomendasi *item*.
- Menambahkan faktor - faktor lain untuk rekomendasi seperti harga *item*, efek *item*, dan lain - lain.
- Pengujian parameter lain pada *neural network* agar hasil lebih optimal.
- Membandingkan pengujian dengan metode lain seperti analisis korelasi.

6. DAFTAR PUSTAKA

[1] Agusta, Y. 2008, August 4. *Association Rules.* URI = https://yudiagusta.wordpress.com/2008/08/04/association-rules/

[2] Aklimovich. 2014, December 21. *In Depth Dota 2 Itemization Guide*. URI = https://steamcommunity.com/sharedfiles/filedetails/?id=356383992

[3] Hanke, L. & Chaimowicz, L. 2017. *A Recommender System for Hero Line-Ups in MOBA Games*. Proceedings of The Thirteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIIDE-17).

[4] Rao, S. & Gupta, R. 2012. *Implementing Improved Algorithm Over APRIORI Data Mining Association Rule Algorithm*. Proceedings of International Journal of Computer Science and Technology, 489-493.

[5] Raschka, S. 2018. *MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack*. The Journal of Open Source Software, 3(24). URI = http://joss.theoj.org/papers/10.21105/joss.00638.

[6] Riedmiller, M. 1994. *Advanced Supervised Learning in Multi-Layer Perceptrons – From Backpropagation to Adaptive Learning Algorithms*. International Journal of Computer Standards and Interfaces, 16.

[7] Schubert, M., Drachen, A., & Mahlmann, T. 2016. *Esports Analytics Through Encounter Detection.* Proceedings of the MIT Sloan Sports Analytics Conference.

[8] Summervile, A., Cook, M., & Steenhuisen, B. 2016. *Draft-Analysis of the Ancients: Predicting Draft Picks in DotA2 Using Machine Learning.* Proceedings of Experimental AI in Games: Papers from the AIIDE Workshop

[9] TensorFlow. 2018. *About TensorFlow*. URI = https://www.tensorflow.org/

[10] TensorFlow. 2018. *Keras*. URI = https://www.tensorflow.org/guide/keras

[11] Wu, X. et al. 2007. *Top 10 Algorithms in Data Mining. Knowledge and Information Systems*, 14(1), 1-37.