PREDIKSI KEMENANGAN BERDASARKAN KOMBINASI HERO PADA GAME DOTA 2 MENGGUNAKAN ALGORITMA NEURAL NETWORK BACKPROPAGATION

Jaka Septian

Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
E-mail : jksptn@gmail.com

ABSTRAK

DoTA 2 merupakan permainan yang berorientasi kerja sama yang melibatkan dua regu untuk saling bertanding, setiap regu masing-masing beranggotakan lima pemain. Setiap pemain mengendalikan satu karakter yang disebut Hero dan bersifat unik. Berdasarkan kombinasi Hero yang dipilih hasil pertandingan dapat diprediksi. Namun memprediksi hasil setiap pertandingan tidaklah mudah, karena Hero yang dapat dipilih berjumlah 110 Hero, maka terdapat sekitar 4,689 x 1013 kemungkinan kombinasi Hero yang dapat terjadi pada satu pertandingan.

Algoritma *Neural network* mempunyai kemampuan memorisasi dan generalisasi. Kemampuan memorisasi adalah kemampuan *Neural network* untuk memanggil kembali secara sempurna pola yang telah dipelajari, dengan kemampuan ini *Neural network* dapat mengenali pola pertandingan game DoTA 2 yang telah ada untuk menebak hasil pertandingan dengan pola yang serupa pada masa yang akan datang. Kemampuan generalisasi adalah kemampuan *Neural network* untuk menghasilkan respons yang bisa diterima terhadap pola-pola *input* yang serupa (namun tidak identik) dengan pola-pola sebelumnya yang telah dipelajari. Berdasarkan hal diatas maka dibangunlah simulator yang mampu memprediksi kemenangan berdasarkan kombinasi hero pada game DoTA 2 menggunakan algoritma *Neural network*.

Kata kunci : *Neural network*, Jaringan Syaraf Tiruan, *Backpropagation*, Prediksi.

1. PENDAHULUAN

Game DoTA 2 adalah permainan berjenis multilplayer online battle arena MOBA). MOBA adalah jenis permainan yang berorientasi kerja sama yang melibatkan dua regu untuk saling bertanding, setiap regu masing-masing beranggotakan lima pemain yang harus saling menghancurkan tower dan benteng utama lawan untuk memenangkan pertandingan. Setiap pemain mengendalikan satu karakter yang disebut Hero dan bersifat unik, maka antara satu pemain dan pemain lainnya harus memilih Hero yang berbeda [1].

Hero yang dimainkan memiliki kemampuan dan peran yang berbeda-beda dengan Hero lainnya. Karena setiap Hero mempunyai kemampuan dan peran yang berbeda maka kombinasi dari Hero yang dipilih oleh kedua regu akan berpengaruh pada hasil pertandingan dan memungkinkan untuk dilakukan prediksi. Dengan memprediksi hasil pertandingan berdasarkan kombinasi Hero, pemain dapat memilih Hero yang sesuai untuk melawan kombinasi dari Hero lawan dan meningkatkan peluang memenangkan pertandingan.

Selain pertandingan biasa dalam game DoTA 2 juga terdapat pertandingan turnamen, banyak turnamen yang diadakan resmi di dalam game ataupun turnamen tidak resmi yang diadakan pihak ketiga. Pemenang dari turnamen-turnamen tersebut mendapatkan hadiah yang biasanya tidak sedikit dan cukup menggiurkan [1]. Karena itu dengan memprediksi hasil kemenangan dari satu pertandingan dapat membantu pemain menentukan langkah demi mencapai kemenangan.

Pemain dalam memprediksi hasil setiap pertandingan tidaklah mudah, karena Hero yang dapat dipilih berjumlah 110 Hero, maka terdapat sekitar 4,689 x 1013 kemungkinan kombinasi Hero yang dapat terjadi pada satu pertandingan. Setiap pertandingan yang telah dilakukan akan membentuk pola kemenangan berdasarkan kombinasi Hero yang dipilih. Pola tersebut dapat dijadikan acuan untuk melakukan prediksi kemenangan, namun pola yang dihasilkan sangat banyak. Algoritma *Neural network* merupakan algoritma yang sering dipakai dalam mempelajari dan mengenali pola pada data. Algoritma *Neural network* mempunyai kemampuan memorisasi dan generalisasi. Kemampuan memorisasi adalah kemampuan *Neural network* untuk memanggil kembali secara sempurna pola yang telah dipelajari, dengan kemampuan ini *Neural network* dapat mengenali pola pertandingan game DoTA 2 yang telah ada untuk menebak hasil pertandingan dengan pola yang serupa pada masa yang akan datang. Kemampuan generalisasi adalah kemampuan *Neural network* untuk menghasilkan

respons yang bisa diterima terhadap pola-pola *input* yang serupa (namun tidak identik) dengan pola-pola sebelumnya yang telah dipelajari [2].

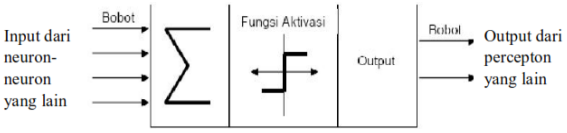
Backpropagation merupakan salah satu arsitektur *Neural network* yang memiliki proses pembelajaran maju dan koreksi kesalahan secara mundur. Model ini banyak digunakan baik itu untuk proses pengenalan maupun prediksi dengan tingkat akurasi yang cukup baik [3]. Sandro Radovanović and Milan Radojičić melakukan penelitian menggunakan metode *Backpropagation* untuk memprediksi kemenangan pertandingan sepakbola berdasarkan pemain yang dimainkan dalam pertandingan tersebut. Dari penelitian tersebut dapat disimpulkan metode *Backpropagation* dapat digunakan untuk memprediksi hasil suatu pertandingan berdasarkan kombinasi variabel yang dipilih [4]. Pada tugas akhir ini akan dibangun simulator prediksi kemenangan berdasarkan kombinasi hero pada gamei DoTA 2 menggunakan algoritma *Neural network Backpropagation*.

Tujuan dari penelitian ini adalah untuk membantu pemain untuk memprediksi hasil dari satu pertandingan berdasarkan kombinasi hero yang dipilih oleh kedua regu dan untuk mengetahui performansi berupa akurasi dan parameter yang optimal metode *Backpropagation* untuk memprediksi kemenangan.

1.1 *Neural network*

Neural network atau Jaringan syaraf tiruan adalah salah satu kajian pada kecerdasan buatan. *Neural network* digunakan untuk menyelesaikan masalah-masalah yang kompleks dan sulit dipahami, dimanan sejumlah besar data mengenai masalah tersebut telah dikumpulkan. *Neural network* mencari pola dan hubungan dalam data yang sangat besar yang terlalu rumit dan sulit dianalisis manusia menggunakan perangkat keras dan piranti lunak yang menyerupai pola-pola pemrosesan dalam otak manusia.

Komponen *Neural network* terdiri dari *neuron-neuron* yang saling berhubungan. *Neuron – neuron* ini akan mentransformasikan informasi yang diterima melalui sambungan keluarannya menuju *neuron-neuron* yang lain. Pada *Neural network* hubungan *neuron-neuron* tersebut dikenal dengan nama bobot [5]. Komponen *Neural network* seperti terlihar pada gambar 1 terdiri dari *input* (informasi) bobot (nilai- nilai tertentu), fungsi aktivasi berfungsi apabila *input* yang dimasukan sesuai dengan nilai ambang (*threshold*) yang ditentukan jika tidak sesuai maka fungsi aktivasi tidak diaktifkan, dan apabila *neuron – neuron* tersebut diaktifkan maka *neuron* akan mengirimkan *output* melalui bobot – bobot.



Gambar 1. Komponen *Neural network*

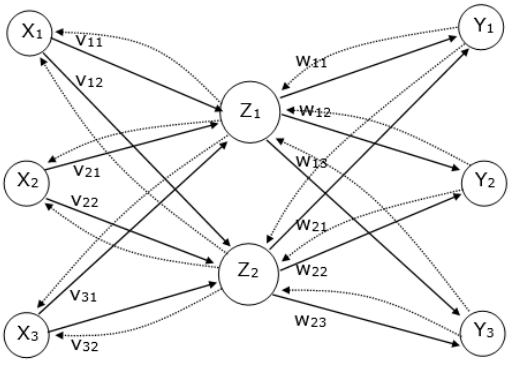
1.1 *Backpropagation*

Backpropagation neural network merupakan tipe jaringan saraf tiruan yang menggunakan metode pembelajaran terawasi (supervised learning). Pada supervised learning terdapat pasangan data *input* dan *output* yang dipakai untuk melatih *Neural network* hingga diperoleh bobot yang diinginkan. Metode *Backpropagation* menggunakan error *output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (backward). Untuk mendapatkan error ini, tahap perambatan maju (forward) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan menggunakan fungsi aktivasi sigmoid, seperti terlihat pada persamaan 1.

Fungsi aktivasi yang mentranformasikan total input Neural Network pada suatu neuron untuk menghasilkan sinyal keluaran (outgoing activity). Fungsi aktivasi yang digunakan pada penelitian ini adalah fungsi sigmoid. Fungsi ini digunakan untuk Neural Network yang dilatih dengan menggunakan metode backpropagation. Fungsi sigmoid memiliki nilai pada range 0 sampai 1. Fungsi ini sering digunakan untuk jaringan yang membutuhkan nilai output yang terletak pada interval 0 sampai 1. Namun, fungsi ini bisa juga digunakan oleh jaringan syaraf yang nilai outputnya 0 atau 1.

$$f(x) = \frac{1}{1+e^{-x}} \qquad (1)$$

Arsitektur *Backpropagation Neural network* terdiri atas tiga lapis (layer) yaitu lapis masukan (*input* layer) xi, lapis tersembunyi (*hidden* layer) zj, dan lapis keluaran (*output* keluaran) yk. Lapis masukan dan lapis tersembunyi dihubungkan dengan bobot vij dan antara lapis tersembunyi dan lapis keluaran dihubungkan oleh bobot w'jk.. Arsitektur dapat dilihat pada gambar 2.



Gambar 2. Arsitektur *Backpropagation*

Jumlah lapis tersembunyi dan jumlah *neuron* tiap lapisan tidak ada rumus tertentu untuk menghitungnya, namun terdapat aturan yang sering dipakai baik untuk menentukan jumlah lapis tersembunyi maupun jumlah *neuron* tersembunyi didalamnya. Permasalahan yang membutuhkan dua lapis tersembunyi jarang ditemukan, meskipun *neural network* yang terdiri dari dua lapis tersembunyi dapat merepresentasikan segala jenis bentuk fungsi. Tidak terdapat alasan teoritis untuk menggunakan *neural network* lebih dari dua lapis tersembunyi. Faktanya untuk berbagai macam permasalahan tidak terdapat alasan untuk menggunakan lebih dari satu lapis tersembunyi [7]. Kemudian untuk menentukan jumlah *neuron* pada lapis tersembunyi sangat penting dalam membangun arsitektur jaringan. Menggunakan terlalu sedikit *neuron* pada lapis tersembunyi kemungkinan akan menghasilkan underfittin, sebaliknya menggunakan terlalu banyak *neuron* pada lapis tersembunyi mungkin akan menghasilkan overfitting dan apabila data latih sangat banyak akan sangat lama untuk melakukan pelatihan [7].
Aturan yang sering dipakai dalam menentukan jumlah *neuron* pada lapis tersembunyi adalah sebagai berikut:

1. Jumlah *neuron* pada lapis tersembunyi adalah antara jumlah *neuron* lapis masukan dan *neuron* lapis keluaran [7].
2. Jumlah *neuron* pada lapis tersembunyi adalah 2/3 jumlah *neuron* lapis masukan, ditambah jumlah *neuron* lapis keluaran [7].
3. Jumlah *neuron* pada lapis tersembunyi harus lebih sedikit dari dua kali lipat jumlah *neuron* lapis tersembunyi [7].
4. Jumlah *neuron* pada lapis tersembunyi dipengaruhi jumlah data latih [8].

Untuk melatih *Backpropagation*, *Neural network* diberi pola masukan sebagai pola pelatihan kemudian pola tersebut akan menuju ke unit pada lapis tersembunyi untuk diterusan pada unit yang berada pada lapis keluaran. Keluaran sementara pada lapis tersembunyi akan diteruskan pada lapis keluaran dan lapis keluaran akan memberikan tanggapan yang disebut sebagai keluaran sementara yk. ketika yk ≠ tk dimana tk adalah keluaran yang diharapkan, maka selisih (error) keluaran sementara u'k akan disebarkan mundur (backward) pada lapis tersembunyi dan diteruskan ke unit pada lapis masukan. Oleh karena itu proses tersebut disebut propagasi balik (*backpropagation*) [6].

2. ISI PENELITIAN

2.1 Metode yang digunakan.

Pada penelitian ini metode yang digunakan adalah Algoritma *Neural network Backpropagation*.

2.1.1 Metode *Backpropagation*

Pada metode *Backpropagation* terdapat dua tahapan proses, yaitu :

1. Tahapan pelatihan, dimana pada tahap ini metode *backpropagation* diberikan sejumlah data pelatihan dan target.
2. Tahap pengujian atau penggunaan, pengujian dilakukan setelah melakukan tahap belajar atau tahap pelatihan

Kondisi pelatihan berhenti atau *stopping condition* dapat menggunakan dua cara yang biasa dipakai, yaitu:

1. Membatasi iterasi yang ingin dilakukan.
   Misalnya jaringan akan dilatih sampai iterasi yang ke-500.
2. Membatasi *error*.
   Misalnya menentukan besar *Mean Square Error*.

Secara umum langkah dalam pelatihan *Neural network* menggunakan *Backpropagation* adalah sebagai berikut :

1. Menentukan jumlah *input* (pola masukan), *hidden* layer, dan *output* (target pelatihan).
2. Memberi nilai awal secara random bagi seluruh bobot antara *input-hidden* layer dan *hidden-output* layer.
3. Melakukan feedforward
   a. Tiap unit *input* (xi) menerima sinyal *input* dan sinyal tersebut dikirimkan pada seluruh unit pada *hidden* layer.
   b. Tiap unit pada *hidden* layer (Z_inj) menjumlahkan sinyal-sinyal *input* terbobot:

$$z_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \qquad (2)$$

   gunakan fungsi aktivasi untuk menghitung sinyal *output*nya:

$$z_j = f(z_in_j) \qquad (3)$$

   c. Tiap unit pada *output* layer (Y_ink) menjumlahkan sinyal-sinya *input* terbobot:

$$y_in_k = w_{0j} + \sum_{i=1}^{n} z_i w_{jk} \qquad (4)$$

   gunakan fungsi aktivasi untuk menghitung sinyal *output*nya:

$$y_k = f(y_in_k) \qquad (5)$$

4. Melakukan *backpropagation*
   a. Tiap unit pada *output* layer (yk) menerima target pola yang dengan berhubungan

dengan pola *input* pembelajaran, hitung informasi errornya :

$$\delta_k = (t_k - y_k)\, f'(y_in_k) \quad (6)$$

Kemudian hitung koreksi bobot (yang nantinya akan digunakan untuk memperbaiki nilai w_{jk}):

$$\Delta w_{jk} = \alpha\, \delta_k\, z_j \quad (7)$$

Hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai w_{0k})

$$\Delta w_{0k} = \alpha\, \delta_k \quad (8)$$

Kirimkan δ_k ke unit yang berada dilapisan bawahnya.

b. Tiap unit pada *hidden layer* (Z_in_j) menjumlahkan delta *input*nya (dari unit yang berada pada lapisan atasnya):

$$\delta_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (9)$$

kalikan nilai ini dengan turunan dari fungsi aktivasinya untu menghitung informasi error:

$$\delta_j = \delta_in_j f'(z_in_j) \quad (10)$$

kemudian hitung koreksi bobot (yang nantinya akan digunakan untuk memperbaiki nilai v_{ij}):

$$\Delta v_{ij} = \alpha\, \delta_j\, x_i \quad (11)$$

Hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai v_{0j}):

$$\Delta v_{0j} = \alpha\, \delta_j \quad (12)$$

c. Tiap unit *output* (y_in_k) memperbaiki bias dan bobotnya:

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (13)$$

tiap unit tersembunyi (z_in_j) memperbaiki bias dan bobotnya:

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (14)$$

5. Tes kondisi berhenti
   a. Apakah batas *error* minimum telah tercapai.
   b. Apakah batas iterasi maksimum telah tercapai.

2.2 Analisis Data

2.2.1 Sumber Data

Data yang digunakan diambil menggunakan *Steam Web API* yang berbentuk file .xml yang selanjutnya dikonversi dan disimpan kedalam sebuah basis data.

2.2.2 Data Masukan

Data yang digunakan dalam memprediksi kemenangan adalah data seluruh data hero yang berjumlah 110 hero. Data Hero tersebut disusun menjadi variabel X1 hingga X110 yang menjadi data masukan pada sistem. Masing - masing variabel dapat memiliki nilai -1 jika dipilih oleh regu lawan, 0 jika tidak dipilih kedua regu dan 1 jika dipilih oleh regu kawan seperti yang terlihat pada tabel 1

Tabel 1. Data Masukan.

Nilai	Keterangan
-1	Variabel Hero yang dipilih regu lawan
0	Variabel Hero yang tidak dipilih
1	Variabel Hero yang dpilih regu kawan

2.2.3 Data Keluaran

Data keluaran atau *output* yang dihasilkan oleh sistem berupa nilai representasi dari dua keadaan yang mungkin terjadi sebagai hasil akhir dari suatu pertandingan. Data keluaran dapat dilihat pada tabel 2 berikut.

Tabel 2. Data Keluaran.

Nilai	Keterangan
1	Menang
0	Tidak Menang

2.3 Pelatihan Metode *Backpropagation*

Masukan pada proses pelatihan adalah *history* pertandingan dengan melihat data *hero* yang dipilih dan regu pemenang dari pertandingan tersebut. Bentuk masukan berupa signal-signal *input* berdasar *hero* yang dipilih kedua regu seperti pada tabel 3.

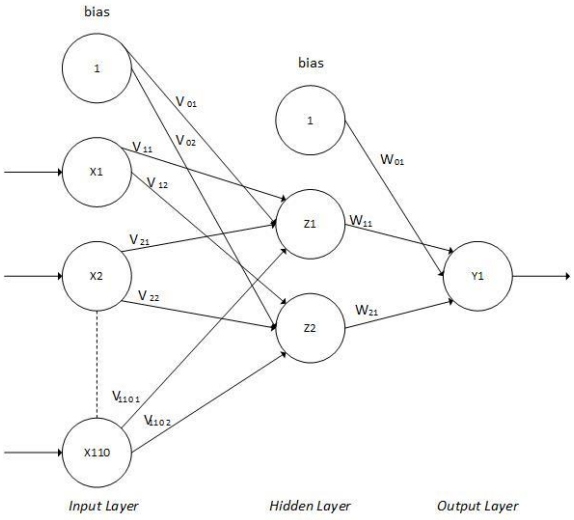
Tabel 3. Hero yang dipilih

No	Hero Regu Kawan	Hero Regu Lawan
1	Juggernaut (X8)	Anti-Mage (X1)
2	Kunkka (X60)	Dark Seer (X55)
3	Slardar (X65)	Weaver (X88)
4	Ancient Apparation (X68)	Gyrocopter (X72)
5	Nyx Assasin (X88)	Troll Warlord (X95)

Dengan target *output* adalah 1 (regu kawan menang)

Data tersebut akan digunakan sebagai pelatihan terhadap arsitektur jaringan yang sudah dibangun. Jaringan yang dibangun mempunyai tiga lapisan yaitu lapisan masukan, lapisan tersembunyi

dan lapisan keluaran. Pada kasus ini layer masukan mempunyai 110 *neuron* dan 1 *neuron* bias, layer tersembunyi mempunyai 2 *neuron* dan 1 node bias, sedangkan pada lapisan keluara mempunyai 1 node. Arsitektur jaringan dapat dilihat pada gambar 3.



Gambar 3. Arsitektur Jaringan Metode *Backpropagation*

Langkah-langkah proses pelatihan adalah sebagai berikut:

1. Tentukan nilai iterasi maksimum dan *learning rate*. Pada contoh perhitungan ini digunakan batasan sebagai berikut:
   Iterasi maksimum = 1000
   Learning rate = 0.05
2. Memberi nilai awal secara random bagi seluruh bobot antara *input-hidden* layer dan *hidden-output* layer.
3. Selama iterasi maksimum belum tercapai lakukan :
   a. *feedforward*
   b. *backpropagation*

Pada data kedua, juga dilakukan operasi-operasi yang sama dengan menggunakan bobot-bobot akhir hasil pengolahan data pertama sebagai bobot awalnya. Proses ini dilakukan secara berulang sampai pada iterasi maksimum.

2.4 Pengujian Metode *Backpropagation*

Pengujian dilakukan untuk menguji jaringan yang bobot-bobotnya sudah diberikan proses pelatihan menggunakan data latih. Proses pengujian hanya melakukan *feedforward* dan membandingkan hasilnya dengan nilai ambang (*threshold*) yang ditentukan.

Hasil dari fungsi aktivasi pada lapis keluaran dibandingkan dengan nilai ambang tertentu. Misalkan diambil nilai ambang (*threshold*) = 0.5, artinya jika nilai yk 0.5 maka *output* yang diberikan adalah 1, namun jika nilai yk < 0.5 *output* yang diberikan adalah 0.

2.5 Hasil Penelitian

Berikut adalah hasil penelitian yang terlebih dahulu dilakukan pengujian terhadap hasil akhir.

2.5.1 Performansi

Pengujian performansi bertujuan untuk mengetahui performansi dan parameter optimal dari metode *backpropagation* yang diimplementasikan pada simulator prediksi kemenangan game DoTA 2. Pengujian metode ini menggunakan cross validation dengan 10 fold, data keseluruhan dibagi menjadi 10 subset dimana setiap subset memiliki data yang berbeda. Data yang digunakan dalam pengujian ini merupakan data pertandingan selama bulan Maret sampai dengan April 2015. Jumlah keseluruhan data yang digunakan adalah 1440 data, data ini dibagi menjadi 10 subset data yang terdiri dari 140 data pertandingan untuk setiap subset. Setiap subset akan diuji dengan menggunakan subset lainnya sebagai data pelatihan, sehingga setiap pengujian dilakukan sebanyak 10 kali. Pengujian ini menghasilkan besaran akurasi dari metode *backpropagation* berdasarkan jumlah data keluaran yang sangat mendekati atau sesuai dengan target keluaran. Pengujian ini dilakukan menggunakan simulator prediksi kemenangan game DoTA 2 yang sudah dibuat.

Pada metode *backpropagation* yang diimplementasikan terdapat tiga buah variabel yang dapat dirubah yaitu jumlah *hidden* layer, learning rate, dan iterasi maksimum. Perubahan nilai iterasi maksimum dilakukan untuk membatasi proses pelatihan metode *backpropagation*, sedangkan perubahan dari jumlah *hidden* layer dan learning rate mempengaruhi laju pembelajaran dan akurasi dari metode *backpropagation*.

Akurasi dari metode ini dapat dilihat dari banyaknya hasil prediksi simulator yang mendekati atau sesuai dengan hasil aslinya. Jumlah hasil prediksi yang akurat nilai persen untuk memudahkan dalam mengetahui nilai akurasi dari sistem.

1. Skenario pengujian 1

Pada pengujian skenario 1, pengujian metode *backpropagation* dilakukan dengan mengubah batas epoch dari 100 hingga 1000, sedangkan kedua variabel lainnya diisi dengan nilai yang sama. Hasil dari skenario pengujian 1 dapat dilihat pada tabel 4.

Tabel 4. Hasil skenario pengujian 2

Epoch	Learning Rate	Jumlah *Hidden Neuron*	Akurasi
100	0.05	5	49.93 %
200	0.05	5	48.82 %
300	0.05	5	49.52 %
400	0.05	5	50.76 %

500	0.05	5	50.14 %
600	0.05	5	50.14 %
700	0.05	5	50.35 %
800	0.05	5	50.56 %
900	**0.05**	**5**	**50.97 %**
1000	0.05	5	49.17 %

Batas epoch yang dipilih berdasarkan nilai akurasi yang terbesar. Berdasarkan pengujian diatas, dipilih batas epoch sebesar 900 karena memiliki akurasi terbesar yaitu 50.97 %. Nilai 900 ini selanjutnya dipakai untuk melakukan pengujian dalam menentukan nilai learning rate. Hasil dari skenario pengujian 2 dapat dilihat pada tabel 5.

Tabel 5. Hasil skenario pengujian 1

Epoch	Learning Rate	Jumlah *Hidden Neuron*	Akurasi
900	**0.05**	**5**	**50.97 %**
900	0.1	5	47.92 %
900	0.15	5	50.59 %
900	0.2	5	50.42 %
900	0.25	5	50.28 %
900	0.35	5	48.54 %
900	0.3	5	49.24 %
900	0.4	5	48.54 %
900	0.45	5	50 %
900	0.5	5	49.52 %

.

2. Skenario pengujian 2

Pada pengujian skenario 2, pengujian metode *backpropagation* dilakukan dengan mengubah nilai learning rate dari 0.05 hingga 0.5 dan batas epoch sebesar 900 sesuai dengan hasil skenario pengujian 1, sedangkan variabel lainnya diisi dengan nilai yang sama.

3. Skenario pengujian 3

Pada pengujian skenario 3, pengujian metode *backpropagation* dilakukan dengan mengubah jumlah *neuron* pada *hidden* layer dari 5 hingga 50 *neuron*, batas epoch sebesar 900 sesuai dengan hasil skenario pengujian 1, dan nilai learning rate 0.05 sesuai dengan hasil skenario pengujian 2. Hasil dari skenario pengujian 3 dapat dilihat pada tabel 6.

Tabel 6. Hasil skenario pengujian 3

Epoch	Learning Rate	Jumlah *Hidden Neuron*	Akurasi
900	0.05	5	50.97 %
900	0.05	10	51.81 %
900	0.05	15	49.44 %
900	**0.05**	**20**	**52.01 %**
900	0.05	25	50.14 %
900	0.05	30	51.53 %
900	0.05	35	50.35 %
900	0.05	40	50.56 %
900	0.05	45	50.97 %
900	0.05	50	49.17 %

Jumlah *neuron* pada *hidden* layer yang dipilih berdasarkan nilai akurasi yang terbesar. Berdasarkan pengujian diatas, dipilih jumlah *neuron* sebanyak 20 karena memiliki akurasi terbesar yaitu 52.01 %.

Pengujian skenario ini merupakan pengujian yang terakhir untuk menentukan parameter optimal Algoritma *Neural network Backpropagation*. Batas epoch yang optimal adalah sebesar 900, nilai learning rate yang optimal adalah 0.05 dan jumlah *neuron* pada *hidden* layer yang optimal adalah sebanyak 20 *neuron*. Akurasi terbesar yang dicapai dari pengujian performansi ini adalah sebesar 52.01%.

2.5.2 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif. Metode yang digunakan dalam pengujian beta ini adalah dengan cara memberikan kuesioner kepada para pemain game DoTA 2. Sampel yang diambil sebanyak 30 orang pemain, dan hasil kuisoner tersebutakan dilakukan perhitungan untuk nantinya dapat diambil kesimpulan penilaian.

Pengujian ini bertujuan untuk melihat apakah simulator yang dibangun dapat mencapai tujuan penelitian. Diberikan pertanyaan “Apakah simulator ini membantu anda dalam bermain game DoTA 2?” dengan menggunakan skala likert sari skala 1 sampai 5.

Berikut ini adalah skor penilaian yang diberikan dengan menggunakan skala likert pada setiap pernyataan nya, yang ditunjukan oleh tabel 7.

Tabel 7. Skor Jawaban

Skala Jawaban	Keterangan	Skor
SS	Sangat Setuju	5
S	Setuju	4
KS	Kurang Setuju	3
TS	Tidak Setuju	2
STS	Sangat Tidak Setuju	1

Hasil dari pengujian beta dapat dilihat pada tabel 8

Tabel 8. Hasil Pengujian

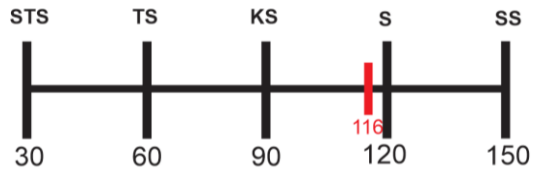
Keterangan	Skor	Responden	Jumlah Skor
Sangat Setuju	5	4	20
Setuju	4	20	80
Kurang Setuju	3	5	15
Tidak Setuju	2	-	-
Sangat Tidak Setuju	1	1	1
Jumlah		**30**	**116**

Untuk mencari nilai persentase dari jawaban digunakan rumus skala likert sebagai berikut :

$$\boldsymbol{P = \frac{S}{Skor\ Ideal} \times 100}$$

$$Y = \frac{116}{150} \text{ X } 100\% = 77\%$$

Berdasarkan nilai persentase tersebut, jumlah skor yang diperoleh adalah 116. Jadi berdasarkan data tersebut, maka dapat disimpulkan bahwa simulator prediksi kemenangan game DoTA 2 membantu pemain secara umum adalah 77% dengan responden 30 pemain dari yang diharapkan 100% dan terletak pada daerah Kurang Setuju. Berikut skala hasil secara kontinum yang ditunjukan pada Gambar 4



Tabel 8. Skala Hasil Penilaian

3. PENUTUP

Berdasarkan dari hasil penelitian, analisis, perancangan sistem, dan implementasi serta pengujian, maka diperoleh kesimpulan bahwa Simulator prediksi kemenangan bersarkan kombinasi hero pada game DoTA 2 menggunakan algoritma *Neural network Backpropagation* :

1. Simulator kurang membatu pemain dalam memprediksi kemenangan game DoTA 2.
2. Dapat mengetahui performansi dan parameter optimal algoritma *Neural network Backpropagation* dalam mengolah data pertandingan untuk dijadikan prediksi.

Berdasarkan hasil penelitian dan implementasi akurasi dari algoritma *Neural network Backpropagation* pada penelitian ini belum mencapai keakuratan yang tinggi. Untuk meningkatkan kinerja dan hasil yang lebih baik, maka diusulkan beberapa saran sebagai berikut :

1. Data pertandingan yang dikumpulkan harus banyak agar algoritma *Neural network* dapat mempelajari pola lebih banyak mengingat kombinasi hero yang mungkin terjadi sangat besar.
2. Perlu dipertimbangkan untuk menggunakan algoritma atau metode lain untuk mencapai akurasi yang tinggi.

DAFTAR PUSTAKA

[1] Valve Corporation, *DoTA 2 - The International*, Februari 2015. [Online]. Tersedia: http://www.dota2.com/international/overview/. [Diakses 12 Februari 2015]

[2] D. Puspitaningrum, *Pengantar Jaringan Saraf Tiruan*, Yogyakarta: Andi, 2006.

[3] C. Dewi dan M. Muslikh, *Perbandingan Akurasi Backpropagation Neural network dan ANFIS Untuk Memprediksi Cuaca*, Journal of Scientific Modeling & Computation, vol. 1, no. 1, pp. 7 - 13, 2013.

[4] S. Radovanović and M. Radojičić, *Premiere League Prediction Using Neural network*, Neuroph Java *Neural network* Framework, 2013. [Online]. Tersedia: http://neuroph.sourceforge.net/tutorials/Sports Prediction/Premier%20League%20Prediction. html. [Diakses 20 Juni 2015].

[5] S. Kusumadewi, *Artificial Intellegence (Teknik dan Aplikasinya)*, Yogyakarta: Graha Ilmu, 2003.

[6] W. Anggraeni, *Aplikasi Jaringan Syaraf Tiruan Untuk Peramalan Permintaan Barang*, *JUTI*, vol. 5, no. 2, pp. 99 - 105, 2006.

[7] J. Heaton, *Introduction to Neural networks for C#, 2nd Edition*, Chesterfield: Heaton Research, Inc, 2008.

[8] M. Leo, "Automatic Inspection of Aircraft Components Using Thermographic and Ultrasonic Techniques," in *Recent Advances in Aircraft Technology*, Intech, 2012, pp. 384-398.