Register : Jurnal Ilmiah Teknologi Sistem Informasi 3 (2) 34-41
ISSN 2502-3357 (*Online*)
ISSN 2503-0477 (*Print*)
Tersedia *online* di www.journal.unipdu.ac.id
Unipdu
Halaman jurnal di www.journal.unipdu.ac.id/index.php/register

Perubahan perilaku *Non-Player Character* (NPC) pada *Game Arabic Hunter* menggunakan Jaringan Syaraf Tiruan *Perceptron*

Syafei Karim

Teknik Informatika, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

email: syfei.karim@gmail.com

I N F O A R T I K E L

Sejarah artikel:
Menerima 1 Januari 2018
Revisi 21 Desember 2017
Diterima 21 Desember 2017
Online 21 Desember 2017

Kata kunci:
Kecerdasan Buatan
Jaringan Syaraf Tiruan
Non-Player Character
Permainan

Keywords:
Artificial Intellegence
Game
Neural Network
Non-Player Character

Style APA dalam mensitasi artikel ini:
Karim, S. (2017). Perubahan perilaku Non-*Player* Character (NPC) pada *Game* Arabic Hunter menggunakan Jaringan Syaraf Tiruan Perceptron. *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, 3(1), 34-41.

A B S T R A K

Permainan pada *smartphone* merupakan aplikasi yang banyak digunakan orang untuk menghabiskan waktu ketika orang tersebut sedang dalam keadaan menunggu atau bosan. Industri *game* merupakan perwujudan pengembangan industri kreatif yang ada pada saat ini. Perkembangan permainan pada *smartphone* juga didukung dengan kemajuan grafis yang membuat lingkungan dan jalannya permainan terlihat lebih realistis. *Game* akan terlihat lebih realistis jika memiliki *Artificial Intelligence* (AI) pada karakternya khususnya pada *Non-Player Character* (NPC). Ketika sebuah *game* sudah memiliki AI yang baik, berarti bahwa karakter permainan menunjukkan perilaku yang konsisten dan realistis, bereaksi dengan tepat kepada tindakan pemain dan karakter lain. Pada *game* AI, ada banyak metode yang bisa digunakan untuk memberikan perilaku pada NPC salah satunya adalah metode Jaringan Saraf Tiruan (JST). Makalah ini memberikan sebuah perubahan perilaku pada NPC dengan menggunakan algoritma *Perceptron*. Perubahan perilaku akan diproses menyesuaikan jumlah *mufradat* dan jumlah poin yang didapatkan pemain. Berdasarkan hasil pengujian dapat disimpulkan bahwa implementasi *Perceptron* untuk memberikan perilaku pada NPC pada *Game Arabic Hunter* dapat berjalan dengan baik. Dari hasil uji coba algoritma, pada proses *learning* dapat diketahui bahwa semakin besar nilai *learning rate*, maka semakin kecil nilai *epoch* yang didapat. Dari uji coba tersebut dihasilkan *learning rate* = 1, *threshold* = 0.6, nilai bias = -1, waktu eksekusi = 1.433 detik. Pada proses *learning* dihasilkan 93% berhasil dan 7% gagal.

A B S T R A C T

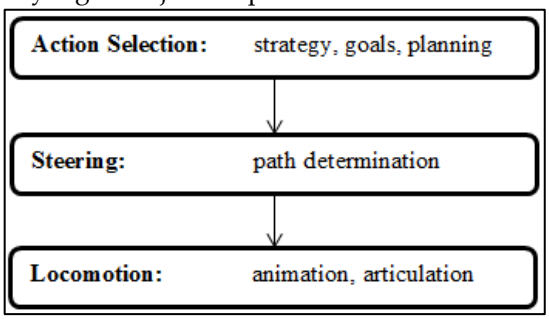
Games on the smartphone is an application that people use to pass the time when the person is in a state of waiting or bored. Game industry is a manifestation of the development of creative industries that exist at this time. Game development on the smartphone is also supported by the progress of graphics that make the environment and the game more realistic. The game will look more realistic if it has artificial intelligence (AI) on its character especially on Non-Player Character (NPC). When a game already has a good AI, it means that the game characters show consistent and realistic behavior, reacting appropriately to the actions of players and other characters. In the game AI, there are many methods that can be used to give the NPC behavior in one of which is a method of an artificial neural network. This paper provides a behavioral change in the NPC using Perceptron algorithm. Changes in behavior will be processed mufradat adjust the amount and number of points earned player. Based on the test results can be concluded that the implementation of the Perceptron to give the NPC behavior in Arabic game hunter can run well. From the test results of the algorithm, the learning process can be seen that the greater the value of learning rate, the smaller value epoch obtained. From these trials generated learning rate = 1, threshold = 0.6, the value of bias = -1, the execution time = 1.433 seconds. In the learning process produced 93% success and 7% failed.

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

1. Pendahuluan

Permainan pada *smartphone* merupakan aplikasi yang banyak digunakan orang untuk menghabiskan waktu ketika orang tersebut sedang dalam keadaan menunggu atau bosan. Menurut seorang analis, lebih dari 23% dari seluruh pengguna ponsel selama 13 tahun di Amerika Serikat bermain *game* di ponsel mereka dan prosentase terus meningkat, terutama untuk 60 juta lebih pengguna *smartphone* (comScore, 2010). Menurut Nielsen (2010), 65% pengguna *smartphone* telah memainkan *mobile game* pada ponsel mereka di beberapa tempat. Dalam perhitungannya, yang berarti sekitar 40 juta orang saat ini bermain *game* di *smartphone* mereka. Teknologi *game* memiliki banyak genre seperti petualangan, olahraga, *puzzle*, strategi, bertarung, dan tak terkecuali *game* edukasi. Dengan *game* edukasi seorang *player* dapat bermain dan belajar. Sehingga *player* tidak hanya mendapatkan kesenangan dalam bermain, tetapi juga mendapatkan ilmu pengetahuan.

Pada khususnya *game* memiliki beberapa komponen yang penting yaitu skenario (alur cerita), level (tingkatan), skor (nilai), karakter, dan *obstacle* (rintangan) (Karim, 2014) (Atmaja, Siahaan, & Kuswardayan, 2016). Industri *game* merupakan perwujudan pengembangan industri kreatif yang ada pada saat ini. Perkembangan permainan pada *smartphone* juga didukung dengan kemajuan grafis yang membuat lingkungan dan jalannya permainan terlihat lebih realistis. *Game* akan terlihat lebih realistis jika memiliki Kecerdasan Buatan (KB) atau *Artificial Intelligence* (AI) pada karakternya khususnya pada *Non-Player Character* (NPC). Tetapi, grafis yang telah dibangun tidak akan menghasilkan sebuah *game* yang realistis jika tidak didukung oleh implementasi perilaku NPC dalam *game* (Thurau, Bauckhage, & Sagerer, 2002). Penelitian tentang AI pada NPC dalam *game*, hingga saat ini masih terus dikembangkan (Hong & Cho, 2005). AI tersebut dikembangkan untuk memberikan perilaku NPC. Ketika sebuah *game* sudah memiliki AI yang baik, berarti bahwa karakter permainan menunjukkan perilaku yang konsisten dan realistis, bergerak dengan tepat kepada tindakan pemain dan karakter lain. Reynolds (1999) membagi perilaku NPC menjadi tiga lapisan: seleksi tindakan (*action selection*), kendali (*steering*), dan penggerak (*locomotion*) seperti yang ditunjukkan pada Gambar 1.



Gambar 1. Hirarki gerak perilaku (Reynolds, 1999)

Pada *game* AI ada banyak metode yang bisa digunakan untuk memberikan perilaku pada NPC salah satunya adalah metode Jaringan Saraf Tiruan (JST) atau *Artificial Neural Network* (ANN). JST merupakan bagian ilmu dari AI. JST dibangun bertujuan untuk meniru kerja otak makhluk hidup dalam menyimpan, belajar, dan mengambil kembali pengetahuan yang telah tersimpan dalam sel saraf atau neuron. JST memiliki beberapa metode pelatihan terbimbing yang telah diciptakan oleh peneliti, diantaranya Hebbian, ADALINE, Hopfield, Backpropagation dan tak terkecuali Perceptron. Perceptron merupakan salah satu metode pembelajaran terawasi (*supervise learning method*) dalam JST. Umumnya, Perceptron digunakan untuk menyelesaikan permasalahan tentang pengenalan pola, pengenalan suara, pengenalan sinyal, dan pengolahan citra maupun permasalahan lainnya. Seperti yang dilakukan lucky pada penelitiannya Van FC (2016), peneliti menggunakan algoritma Perceptron untuk memetakan pola gaya belajar mahasiswa. Dari algoritma tersebut menghasilkan suatu pembelajaran yang mampu memberikan hasil yang optimal terhadap hasil gaya belajar mahasiswa, sehingga untuk tenaga pengajar di kampus dapat menentukan metode yang tepat dalam proses pengajaran dan pembelajaran. Sama halnya dengan Azmi, Saripurna, dan Anwar (2013), pada penelitiannya menggunakan algoritma Perceptron untuk mengetahui pola pembukaan permainan catur. Hasil tersebut dapat meningkatkan kinerja program catur yang sudah ada sekaligus meningkatkan pengetahuan para pecatur untuk mempersiapkannya dalam pertandingan.

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

Berdasarkan dari penelitian sebelumnya, algoritma Perceptron dianggap mampu untuk memberikan sebuah pembelajaran untuk pengenalan pola. Dalam hal ini, pada penelitian ini diusulkan untuk menggunakan algoritma Perceptron ke dalam *game* untuk pengenalan pola pada perilaku NPC sehingga bisa menghasilkan perilaku NPC sesuai dengan kondisi yang ada.

2. *State of the Art*

Wardhana (2009) dalam penelitiannya mengimplementasikan AI dalam *game*, di mana salah satu elemen *game* yaitu NPC mampu mengenali emosi dari teks bahasa Indonesia sekaligus merespon dengan perilaku sesuai dengan jenis emosi NPC. Wardhana (2009) menggunakan klasifikasi teks untuk menentukan jenis emosi dan Logika *Fuzzy* sebagai penentu perilaku dari NPC. Dari hasil yang sudah dilakukan terdapat beberapa kelemahan yaitu memiliki keakuratan klasifikasi yang masih rendah dan respon emosi dari NPC yang tidak bersifat manusiawi.

Arif, Wicaksono, dan Kurniawan (2012) membuat aksi dan reaksi pada NPC sehingga NPC memiliki perilaku strategi menyerang terhadap musuh. Peneliti menggunakan Logika *Fuzzy* untuk menentukan respon perilaku NPC terhadap kondisi yang dihadapi. Dengan harapan *game* yang digunakan dapat terlihat lebih hidup dengan kondisi NPC yang memiliki AI. Arif, Wicaksono, dan Kurniawan (2012) melakukan 10 kali percobaan untuk menguji tingkat kemenangan ketika melawan NPC jahat. Hasil uji coba menunjukkan nilai 80% untuk tingkat kemenangan strategi menyerang jika melawan musuh yang memiliki perilaku menyerang dan menghindar. Perilaku NPC sudah dapat menghasilkan perilaku yang bervariasi sesuai dengan variabel yang dimiliki.

3. Metode Penelitian

3.1. *Perancangan AI pada Perilaku NPC*

Game yang dibangun adalah *game single player* yang bergenre *side-scrolling*. Terdapat beberapa karakter pada *game* ini yang didesain agar *game* sangat menarik dimainkan. Karakter tersebut berupa pemain utama dan beberapa karakter NPC. NPC akan diberikan sebuah perilaku yang cerdas agar *game* yang dimainkan jadi lebih menarik. Perilaku NPC dapat berubah berdasarkan dengan parameter yang telah diatur yaitu jumlah poin dan jumlah *mufradat*.

AI dalam penelitian ini dikonsentrasikan pada perilaku NPC. Untuk memberikan perilaku pada NPC dibutuhkan beberapa data yang digunakan. Dalam mengolah data ini dibutuhkan sebuah variabel yang digunakan sebagai *input*. Variabel yang digunakan jumlah poin dan jumlah *mufradat*. Variabel yang digunakan setiap tingakatan akan selalu berubah. Pada level pertama koin penuh yang disediakan berjumlah 20 koin, sedangkan jumlah *mufradat* yang disediakan berjumlah 8 *mufradat*. Untuk level kedua jumlah poin semakin meningkat menjadi 30 koin, sedangkan jumlah *mufradat* menjadi 12 *mufradat*. Pada level terakhir, jumlah koin menjadi 40, sedangkan jumlah *mufradat* menjadi 15 *mufradat*. 1 koin dihargai dengan 10 poin. Semakin banyak koin yang didapat, maka semakin banyak poin yang didapat.

Dari variabel yang sudah ditentukan tadi, maka dapat dibuat aturan untuk *input* pada perhitungan JST Perceptron sebagai $f(x)$. Persamaan 1 adalah aturan *input* $f(x)$ yang digunakan pada setiap level,

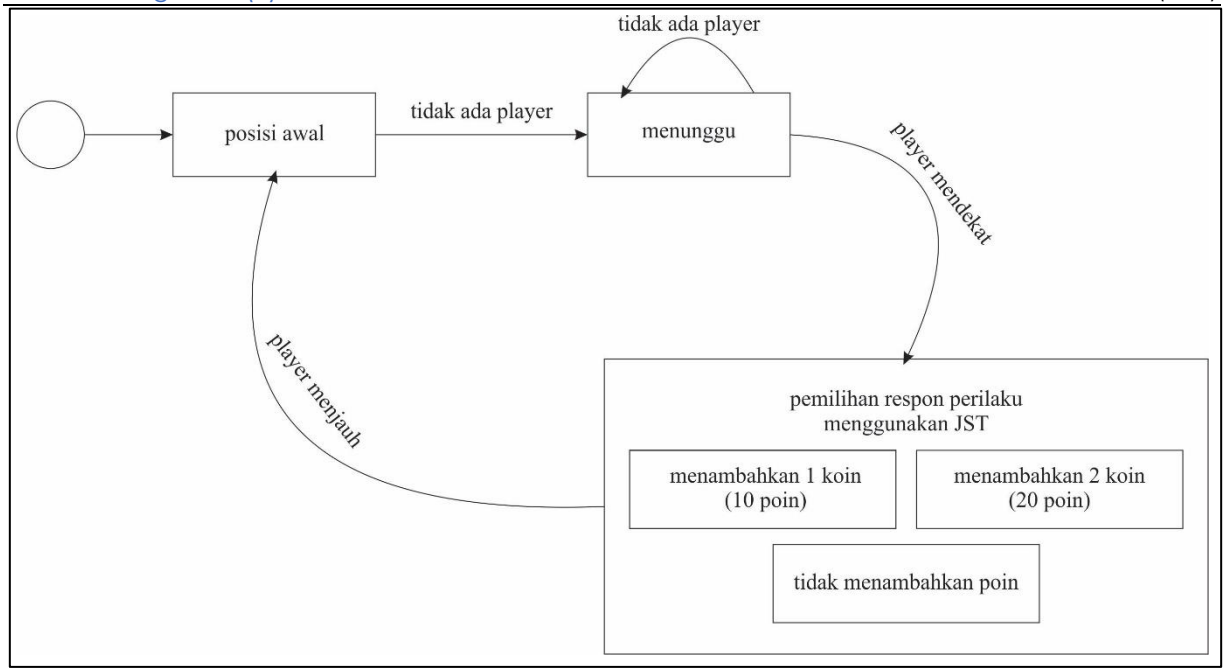
- Level 1: $f(x) = \begin{cases} 0, & koin < 10 \\ 1, & koin \geq 10 \end{cases}$ dan $f(x) = \begin{cases} 0, & mufradat < 5 \\ 1, & mufradat \geq 5 \end{cases}$
- Level 2: $f(x) = \begin{cases} 0, & koin < 18 \\ 1, & koin \geq 18 \end{cases}$ dan $f(x) = \begin{cases} 0, & mufradat < 8 \\ 1, & mufradat \geq 8 \end{cases}$
- Level 3: $f(x) = \begin{cases} 0, & koin < 26 \\ 1, & koin \geq 26 \end{cases}$ dan $f(x) = \begin{cases} 0, & mufradat < 12 \\ 1, & mufradat \geq 12 \end{cases}$ (1)

Aturan tersebut berlaku pada NPC jahat dan NPC penolong. Berikut hasil *output* dari *input* yang digunakan:

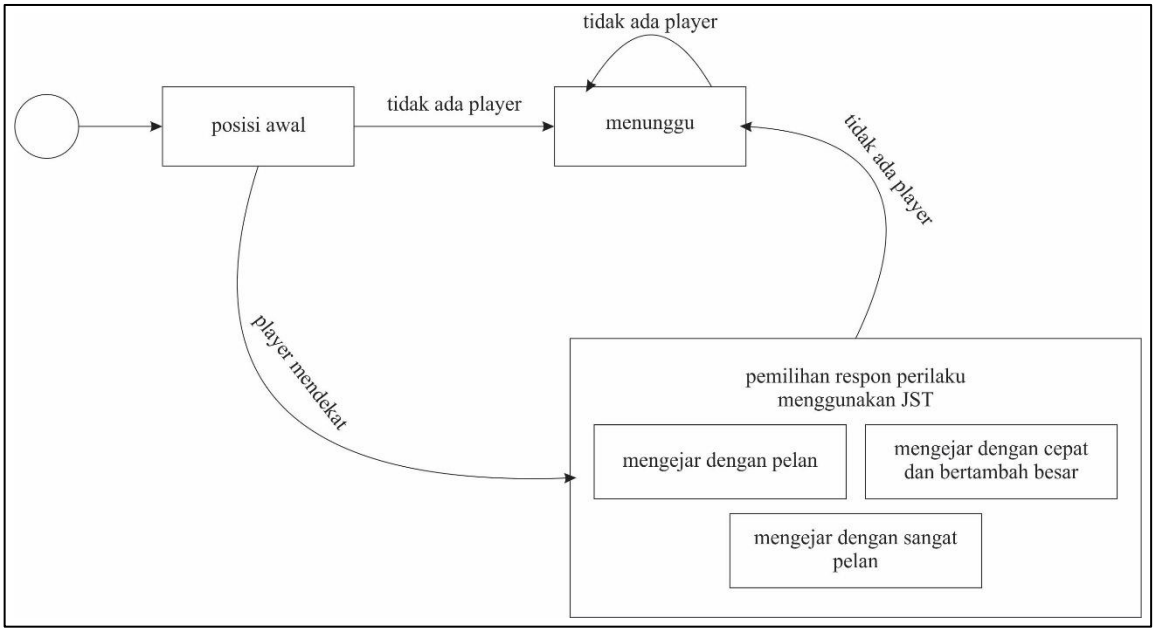
- NPC jahat : *Output* 1 maka mengejar dengan cepat dan bertambah besar, *output* 0 maka mengejar dengan lambat, *output* -1 maka mengejar dengan sangat lambat
- NPC penolong: *Output* 1 maka tidak menambahkan poin dan *mufradat*, *output* 0 maka menambahkan 1 koin (10 poin), *output* -1 maka menambahkan 2 koin (20 poin).

3.2. *Finite State Machine (FSM) NPC Penolong*

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)



Gambar 1. FSM NPC penolong



Gambar 2. FSM NPC jahat

Saat pemain *start* memainkan *game*, NPC ini akan berada pada posisinya menunggu pemain datang. NPC penolong akan tetap menunggu hingga pemain mendekat, jika pemain mendekat maka akan dilakukan pemilihan respon perilaku. Ada tiga respon perilaku yang dimiliki, yaitu menambahkan 2 koin (20 poin), menambahkan 1 koin (10 poin), dan tidak menambahkan koin. Pemberian perilaku berdasarkan aturan yang sudah dibuat, seperti pada Gambar 2 menggambarkan FSM NPC Penolong.

3.3. Finite State Machine (FSM) NPC Jahat

Saat pemain *start* memainkan *game* NPC ini akan berada pada posisinya menunggu pemain datang. NPC jahat akan jalan-jalan menunggu hingga pemain mendekat, jika pemain mendekat maka akan dilakukan pemilihan respon perilaku. Ada tiga respon perilaku yang dimiliki, mengejar dengan cepat dan bertambah besar, mengejar dengan pelan, dan mengejar dengan sangat pelan. Pemberian perilaku berdasarkan aturan yang sudah dibuat pada Gambar 3 menggambarkan FSM NPC Jahat.

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

3.4. Algoritma Perceptron

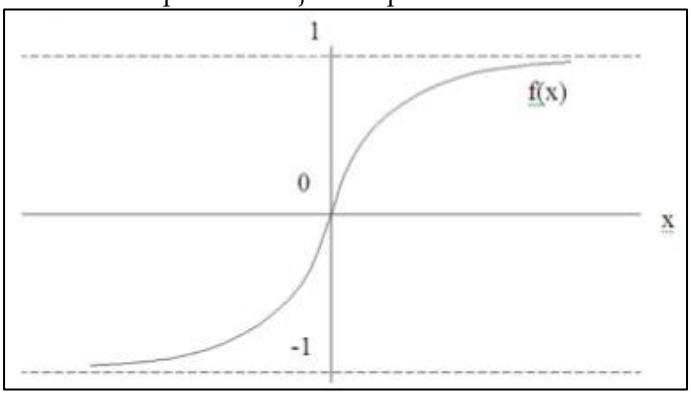
Algoritma Perceptron diimplementasikan dengan blok diagram pada Gambar 4. *Input* diambil dari jumlah *score* dan *mufradat*. Terdapat dua nilai *input* yang digunakan yaitu 0 dan 1. *Input* yang didapat akan dihitung dengan bobot dan bias yang sudah diketahui untuk mendapatkan nilai *output*. Fungsi aktivasi yang digunakan adalah fungsi sigmoid bipolar yang ditunjukkan pada Gambar 4, sehingga ada tiga nilai *output* yang digunakan yaitu 1, 0, dan -1. Adapun rumus untuk mencari nilai *output* yang ditunjukkan pada Persamaan 4, di mana sebelum mendapatkan nilai *output* harus didapatkan terlebih dahulu nilai aktivasi seperti pada Persamaan 2 dan nilai *yOut* pada Persamaan 3.

$$aktivasi = (input1 \times bobot1) + (input2 \times bobot2) \quad (2)$$

$$yOut = bias + aktivasi \quad (3)$$

$$Output = \begin{cases} 1, jika\ yOut > threshold \\ 0, jika - \theta \leq yOut \leq threshold \\ -1, jika\ yOut < -threshold \end{cases} \quad (4)$$

Sedangkan alur implementasi Perceptron ditunjukkan pada Gambar 5.



Gambar 3. Fungsi sigmoid bipolar

Gambar 4. Blok diagram implementasi Perceptron

4. Hasil dan Pembahasan

Pengujian diawali dengan pengujian Perceptron yang digunakan untuk menentukan variasi perilaku NPC berdasarkan variabel *input* yang dimiliki.

4.1. Pengujian Algoritma Perceptron

Uji coba algoritma Perceptron ini dilakukan untuk mengetahui pola data yang akan dikenali. Proses ini melakukan uji coba pada *learning rate*, *threshold*, dan mencari nilai *epoch* yang dicapai. Percobaan dilakukan untuk mendapatkan nilai *learning rate* dan nilai *threshold* terbaik. Pada Tabel 1 menunjukkan hasil uji coba nilai *epoch* yang gagal. Terdapat tujuh data yang tidak bisa menghasilkan target yang diinginkan dikarenakan nilai bobot dan bias yang tidak stabil sehingga nilai *epoch* mencapai batas maksimal.

Nilai *epoch* merupakan iterasi yang dilakukan untuk mendapatkan nilai bobot dan bias yang sesuai dengan target yang diinginkan. Untuk menghasilkan proses pembelajaran yang maksimal maka iterasi dilakukan hingga 1.000 iterasi. Pada Tabel 2 dapat diketahui nilai *learning rate* dan *threshold* berpengaruh pada performansi jaringan. Semakin besar nilai *learning rate*, maka semakin sedikit iterasi

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

yang diperlukan untuk mendapatkan nilai bobot dan bias mencapai stabil. Pada saat uji coba, dihasilkan 93% keberhasilan dalam melakukan proses *learning* dan terdapat 7% yang gagal.

Tabel 1. Hasil uji coba nilai *epoch* yang gagal

No.	*Learning rate*	*Threshold*	*Max epoch*	*Epoch* yang dicapai
1	0.4	0.1	1000	1000
2	0.6	0.1	1000	1000
3	0.6	0.1	1000	1000
4	0.7	0.1	1000	1000
5	0.8	0.1	1000	1000
6	0.7	0.2	1000	1000
7	0.7	0.3	1000	1000

Tabel 2. Uji coba nilai *epoch* yang berhasil

No.	*Learning rate*	*Threshold*	*Max epoch*	*Epoch* yang dicapai
1	0.1	0.1	1000	15
2	1	0.1	1000	4
3	0.1	0.2	1000	17
4	1	0.2	1000	4
5	0.1	0.3	1000	19
6	1	0.3	1000	4
7	0.1	0.4	1000	21
8	1	0.4	1000	4
9	0.1	0.5	1000	23
10	1	0.5	1000	4
11	0.1	0.6	1000	25
12	1	0.6	1000	4
13	0.1	0.7	1000	27
14	1	0.7	1000	4
15	0.1	0.8	1000	29
16	1	0.8	1000	4
17	0.1	0.9	1000	31
18	1	0.9	1000	4
19	0.1	1	1000	33
20	1	1	1000	6

Setelah diketahui nilai *epoch* maksimal maka dilakukan uji coba kembali untuk mengetahui waktu eksekusi. Hasil uji coba waktu eksekusi dapat dilihat pada Tabel 3. Hasil yang didapatkan yaitu *learning rate* = 1, *threshold* = 0.6, waktu eksekusi = 1.433 detik, dan maksimum *epoch* = 1.000. Dari hasil pembelajaran yang sudah dilakukan didapatkan nilai untuk kedua bobot adalah 1 dan nilai bias adalah -1.

Tabel 3. Hasil uji coba waktu eksekusi

Learning rate	*Threshold*	*Max epoch*	Waktu eksekusi (detik)
1	0.1	4	1.497
1	0.2	4	1.665
1	0.3	4	1.474
1	0.4	4	1.566
1	0.5	4	1.802
1	0.6	4	1.433
1	0.7	4	1.582
1	0.8	4	1.695
1	0.9	4	1.573
1	1	6	1.628

4.2. Pengujian NPC Penolong

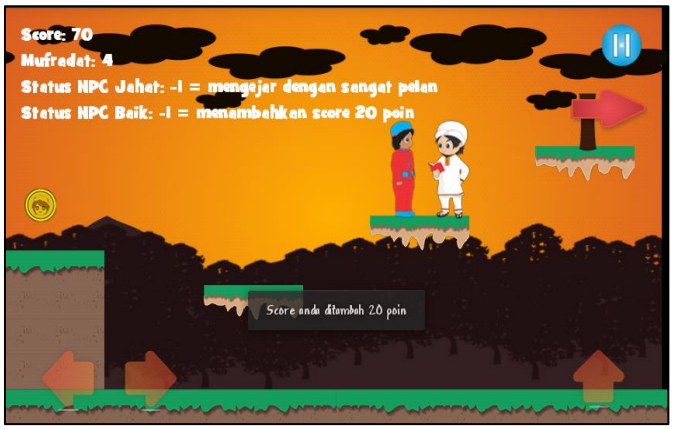
Setelah melakukan hasil uji coba algoritma Perceptron maka dihasilkan nilai *learning rate, threshold*, nilai bobot dan nilai bias yang terbaik. Data tersebut akan dilakukan pengenalan terhadap data yang sudah ditentukan seperti pada Tabel 4. Setelah dilakukan pengenalan maka dapat dihasilkan target yang diinginkan. Pada Gambar 6 merupakan hasil pengujian perilaku NPC penolong yang berdasarkan

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

dengan variabel *input* yang sudah ditentukan. Nilai *input* didapatkan berdasarkan jumlah *score* dan jumlah *mufradat* yang didapatkan oleh *player* sesuai dengan aturan yang sudah dibuat. Perilaku yang dilakukan NPC tersebut sesuai dengan target yang diinginkan.

Tabel 4. Data NPC penolong

No.	Input 1 (x1)	Input 2 (x2)	Target
1	0	0	-1
2	0	1	0
3	1	0	0
4	1	1	1



Gambar 5. Hasil pengujian NPC penolong

4.3. Pengujian NPC Jahat

Sama halnya dengan NPC penolong, data yang ada pada Tabel 5 dilakukan pengenalan dengan data Perceptron yang sudah diujikan. Pengujian ini dilakukan untuk menguji perilaku NPC jahat berdasarkan variabel *input* yang didapatkan oleh *player*. Di mana nilai *input* diambil berdasarkan jumlah *score* dan jumlah *mufradat* yang kemudian dikonversi menjadi nilai 0 dan 1 sesuai aturan yang sudah ditentukan. Pada Gambar 7 terlihat bahwa perilaku NPC jahat sesuai dengan target yang telah dirancang sebelumnya. Tabel 5 merupakan data *input* dan target NPC jahat.

Tabel 5. Data NPC jahat

No.	Input 1 (x1)	Input 2 (x2)	Target
1	0	0	-1
2	0	1	0
3	1	0	0
4	1	1	1

Gambar 6. Hasil pengujian npc jahat

ISSN 2502-3357 (*Online*)
S. Karim./Register 3 (1) 34-41 ISSN 2503-0477 (*Print*)

5. Kesimpulan

Dari hasil implementasi dan uji coba, algoritma ini dapat digunakan untuk menentukan perilaku NPC sesuai dengan keahlian *player* dalam bermain game ini. Perilaku NPC penolong dan jahat dapat bergerak dengan sesuai dengan aturan yang dibuat berdasarkan dari jumlah mufradat dan *score* yang didapat oleh *player*. Pada proses *training*, besarnya nilai *learning* berpengaruh terhadap nilai *epoch* yang dicapai. Semakin besar nilai *learning rate* semakin kecil nilai *epoch* yang dicapai. Dari rata-rata hasil uji waktu eksekusi, untuk menyelesaikan proses *learning* menggunakan algoritma ini membutuhkan waktu 1.591 detik. Pada proses *learning* dapat diketahui 93% yang berhasil dan 7% yang gagal.

6. Referensi

Arif, Y. M., Wicaksono, A., & Kurniawan, F. (2012). Pergantian Senjata NPC pada Game FPS Menggunakan Fuzzy Sugeno. *Prosiding Seminas Competitive Advantage. 1.* Jombang: Unipdu.

Atmaja, P. W., Siahaan, D. O., & Kuswardayan, I. (2016). Game design document format for video games with passive dynamic difficulty adjustment. *Register: Jurnal Ilmiah Teknologi Sistem Informasi, 2*(2), 86-97.

Azmi, Z., Saripurna, D., & Anwar, B. (2013). Aplikasi Jaringan Syaraf Tiruan Untuk Pengenalan Pola Pembukaan Permainan Catur. *Jurnal SAINTIKOM, 12*(2), 139-152.

comScore. (2010, December 3). *Press Release*. Retrieved from comScore: http://www.comscore.com/Insights/Press-Releases/2010/12/comScore-Reports-October-2010-US-Mobile-Subscriber-Market-Share?cs_edgescape_cc=ID

Hong, J. H., & Cho, S. B. (2005). Evolving Reactive NPCs for the Real-Time Simulation Game. *IEEE Symposium on Computational Intelligence and Games.* Colchester: IEEE.

Karim, S. (2014). *Pada khususnya game memiliki beberapa komponen yang penting yaitu skenario (alur cerita), level (tingkatan), skor (nilai), karakter, dan obstacle (rintangan).* Malang: UIN Maulana Malik Ibrahim.

Nielsen. (2010). *Digital*. Retrieved from Nielsen: http://www.nielsen.com/us/en/insights/news/2010/the-state-of-mobile-apps.html

Reynolds, C. W. (1999). Steering Behaviors For Autonomous Characters. *Game Developers Conference*, (pp. 763-782).

Thurau, C., Bauckhage, C., & Sagerer, G. (2002). Learning human-like Movement Behavior for Computer Games. *Proceeding of the Seventh International Conference on Simulation of Adaptive Behavior* (pp. 315-323). London: MIT.

Van FC, L. L. (2016). Klasifikasi gaya belajar Visual-Audiotory-Kinesthetic (V-A-K) mahasiswa berbasis JST menggunakan algoritma Perceptron. *Jurnal Teknologi Informasi & Komunikasi Digital Zone, 7*(1), 26-30.

Wardhana, M. I. (2009). *Kecerdasan Buatan dalam game untuk merespon emosi dari teks berbahasa Indonesia menggunakan klasifikasi teks dan Logika Fuzzy.* Surabaya: Institut Teknologi Sepuluh Nopember.