Kumpulan jurnaL Ilmu Komputer (KLIK)
Volume 05, No.02 September 2018
ISSN: 2406-7857

IMPLEMENTASI ALGORITMA BACKPROPAGATION UNTUK MEMPREDIKSI KELULUSAN MAHASISWA

[1]Muhammad Dedek Yalidhan, [2]Muhammad Faisal Amin
[1,2] Program Studi Sistem Informasi, STMIK Banjarbaru
Jl. A. Yani KM. 33,3 Loktabat, Banjarbaru, 0511 4782881
[1]dedek.southborneo@gmail.com

Abstract

Student's graduation is one kind of the college accreditation elements by BAN-PT. Because of that. Information System is one of the department in STMIK Banjarbaru, there is no application has been implemented to predict imprecisely of student's graduation time so far, which causes on time graduation percentage tend low every year. Therefore the accurate student's graduation prediction can help the committe to choose the correct decisions in order to prevent the imprecisely of student's graduation time. In this research, the backpropagation algorithm of artificial neural network will be implemented into the application with the output result as delayed and on time graduation. This reseach is using 318 data samples which the 70 % of it will be used as the training data and the other 30 % will be used as testing data. From the calculation of confusion matrix table's the percentage of the prediction accuracy is 98.97 %.

Keywords*: student's graduation, artificial neural network, backpropagation, confusion matrix*

Abstrak

Kelulusan mahasiswa merupakan salah satu elemen dalam standar akreditasi perguruan tinggi oleh BAN-PT. Sistem Informasi adalah salah satu program studi yang ada di STMIK Banjarbaru, selama ini belum ada aplikasi yang diimplementasikan untuk memprediksi ketidaktepatan waktu kelulusan mahasiswanya yang menyebabkan angka kelulusan tepat waktu cenderung rendah setiap tahunnya. Oleh sebab itu, prediksi kelulusan mahasiswa yang akurat dapat membantu pihak Program Studi dalam mengambil keputusan-keputusan yang tepat untuk mencegah ketidaktepatan waktu kelulusan mahasiswanya. Pada penelitian ini, artificial neural network algoritma backpropagation diimplementasikan pada aplikasi yang dibuat dengan output lulus terlambat dan lulus tepat waktu. Penelitian ini menggunakan sebanyak 318 sampel data yang mana 70 % data digunakan sebagai data training dan 30 % data digunakan sebagai data testing. Dari hasil perhitungan tabel confusion matrix diperoleh persentase akurasi prediksi sebesar 98.97 %.

Kata kunci*: kelulusan mahasiswa, artificial neural network, backpropagation, confusion matrix*

1. PENDAHULUAN

Ada beberapa standar akreditas perguruan tinggi oleh BAN-PT, waktu kelulusan mahasiswa menjadi salah satu elemen dalam standar ke-3 (mahasiswa

dan lulusan)[1]. Sistem Informasi adalah salah satu program studi yang ada di STMIK Banjarbaru, selama ini belum ada aplikasi yang diimplementasikan untuk memprediksi ketidaktepatan waktu kelulusan mahasiswanya.

Menurut data dari program studi sistem informasi sebanyak 169 mahasiswa baru angkatan 2011 yang lulus tepat waktu hanya berjumlah 29 orang (17%) [2], menurut BAN-PT persentase KTW 17% hanya masuk ke dalam harkat dan peringkat CUKUP, dimana KTW < 50%, maka skor = 1 + (6 x KTW) =2. (dengan deskriptor 4: Sangat Baik, 3: Baik, 2: Cukup, 1: Kurang, 0: Sangat Kurang) [1].

Selama ini pada bagian program studi Sistem Informasi STMIK Banjarbaru belum ada metode yang dapat memprediksi mahasiswa yang akan lulus tidak tepat waktu sejak dini, sehingga pihak program studi tidak dapat membantu mengarahkan mahasiswanya agar lulus tepat waktu.

Jika perguruan tinggi tidak berusaha membantu mengarahkan mahasiwa, maka akan berdampak mahasiswa yang lulus tepat waktu akan berkurang, dikarenakan kurangnya peringatan serta tindakan dini untuk mencegah hal tersebut.

Berdasarkan kondisi tersebut diperlukan penelitian untuk mengolah data yang dimiliki oleh bagian akademik STMIK Banjarbaru. Data yang akan dimanfaatkan di sini adalah data nilai akademik mahasiswa baik yang sudah lulus (untuk data traning dan data testing) maupun yang belum lulus ataupun yang masih menjalani studi yang akan dimanfaatkan untuk memprediksi masa studi masing-masing mahasiswa. Penelitian ini diperlukan karena jika masa studi mahasiswa dapat diprediksi lebih dini, maka pihak program studi maupun mahasiswa yang bersangkutan dapat melakukan tindakan-tindakan yang diperlukan untuk mencegah keterlambatan kelulusan mahasiswa yang bersangkutan sekaligus meningkatkan kualitas program studi itu sendiri.

Ada beberapa penelitian mengusulkan penerapan algoritma backpropagation untuk memprediksi, diantaranya adalah pada penelitian yang dilakukan oleh Rudy Ansari, algoritma *backpropagation* dengan pengaturan atau parameter awal 12 *input*, 9 *hidden node*(neuron) dan 3 *output*, digunakan untuk memprediksi kelulusan mahasiswa STMIK Indonesia Banjarmasin dan menghasilkan akurasi sebesar 92,49%. [3] Sedangkan pada penelitian ini akan dicoba pengaturan parameter lainnya guna menghasilkan akurasi yang lebih akurat, karena menurut penelitian yang dilakukan oleh Lestari Handayani dkk. [4] dan Redjeki [5], variabel atau parameter seperti *learning rate*, *goal* (total error), fungsi aktivasi dan *hidden layer* sangat mempengaruhi hasil dari perhitungan algoritma *backpropagation*.

2. METODOLOGI PENELITIAN

2.1. Kelulusan Tepat Waktu

Untuk memenuhi capaian pembelajaran lulusan Program Sarjana sesuai dengan Standar Nasional Pendidikan Tinggi, bahwa mahasiswa wajib menempuh beban belajar paling sedikit untuk Program Studi Sistem Informasi sebanyak 145 sks (136 sks mata kuliah wajib dan 9 sks mata kuliah pilihan) yang dijadwalkan

untuk sekurang-kurangnya 7 (tujuh) semester bagi mahasiswa yang memiliki prestasi akademik tinggi atau 8 (delapan) semester bagi mahasiswa normal. [6]

2.2. *Artificial Neural Network* (Jaringan Syaraf Tiruan)

Artificial Neural Network (Jaringan Syaraf Tiruan) adalah model non-linear yang kompleks, dibangun dari komponen yang secara individu berperilaku mirip dengan model regresi. Jaringan syaraf tiruan dapat divisualisasikan sebagai grafik, dan beberapa sub-grafik mungkin ada perilaku yang sama dengan gerbang logika. Meskipun struktur dari jaringan saraf secara eksplisit dirancang terlebih dahulu, pengolahan bahwa jaringan tidak untuk menghasilkan hipotesis (berbagai gerbang logika dan pengolahan lainnya terstruktur dalam jaringan) berkembang selama proses pembelajaran. Hal ini memungkinkan neuron yang membentuk jaringan akan digunakan sebagai pemecahan masalah dari "program itu sendiri". [7]

2.3. Algoritma *Backpropagation*

Backpropagation merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh perceptron dengan banyak lapisan untuk mengubah bobot- bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyinya. Algoritma *backpropagation* menggunakan *error output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini, tahap perambatan maju (*feedforward*) harus dikerjakan terlebih dahulu. [8]

2.4. Proses Training *Backpropagation*

Menurut Fausett [9] berikut ini adalah algoritma pelatihan jaringan propagasi balik untuk satu *hidden layer*:

a. Inisialisasi bobot dengan memberikan nilai acak (nilai acak kecil - 0.5 s/d 0.5).
b. Selama kondisi berhenti false, lakukan langkah 3-9.
c. Untuk setiap pasangan data pelatihan (x_setb, tb) dimana b=1,…l, lakukan 4-8.
d. Memulai proses *forward*, setiap unit *input* (Xi, i = 1,…, n) menerima sinyal *input* xi dan melanjutkannya ke *hidden layer*, setiap unit tersembunyi (Zj ,j= 1,…p) menjumlahkan sinyal-sinyal *input* terbobot,

$$Z_{\_in_j} = vo_j + \sum_{i=1}^{n} x_i v_{ij}$$

gunakan fungsi aktivasi untuk menghitung sinyal *output*-nya,

$$Z_i = f(Z\ in_i)$$

dan lanjutkan sinyal ke semua unit di lapisan atasnya (*output layer*).

e. Setiap unit output (Yk, k = 1,…, m) menjumlahkan sinyal-sinyal *input* terbobot,

$$y_{\_in_k} = wo_k + \sum_{j=1}^{p} z_j w_{jk}$$

gunakan fungsi aktivasi untuk menghitung sinyal *output*-nya,

$$y_k = f(y\_in_k$$

dan lanjutkan ke proses *backward*.

f. Tiap-tiap unit output (yk, k=1,..., m) menerima pola target yang berhubungan dengan pola *input* pembelajaran, hitung informasi *error* k,

$$\delta_k = (t_k - y_k) f'(\mathrm{y\_in_k})$$

hitung koreksi bobot wjk,

$$\Delta w_{jk} = \alpha \delta_k z_j$$

hitung koreksi bias w0k,

$$\Delta w o_k = \alpha \delta_k$$

dan kirimkan nilai informasi error ke lapisan bawahnya.

g. Tiap–tiap *hidden unit* menjumlahkan hasil kali informasi error dengan *Weight*.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

hitung informasi *error* j,

$$\delta_j = \delta_{in_j} f'(z\_in_j)$$

koreksi bobot vij,

$$\Delta v_{ij} = \alpha \delta_j x_t$$

dan koreksi bias v0j,

$$\Delta v_{oj} = \alpha \delta_j$$

lanjutkan ke tahap peng-*update*-an *weight*.

h. Setiap unit *output* (Yk, k = 1,…, m) memperbaiki bobot dan biasnya (j = 0,…, p),

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

setiap unit tersembunyi (Zj, j = 1,…, p) memperbaiki bobot dan biasnya (i= 0,…,n),

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

lanjutkan ke tes kondisi.

i. Tes kondisi, jika *true* maka *training* berhenti.

2.5. Momentum

Pada standard *backpropagation*, perubahan bobot didasarkan atas *gradient* yang terjadi untuk pola yang dimasukkan pada saat itu. Modifikasi yang dapat dilakukan adalah dengan menggunakan momentum yaitu dengan melakukan perubahan bobot yang didasarkan atas arah *gradient* pola terakhir dan pola sebelumnya yang dimasukkan. Penambahan momentum dimaksudkan untuk menghindari perubahan bobot yang mencolok yang diakibatkan oleh adanya data yang sangat berbeda dengan yang lain. Variabel momentum dapat meningkatkan waktu pelatihan dan stabilitas dari proses pelatihan [10]. Berikut merupakan rumus momentum menurut Fausett [9]:

$$w_{jk}(t+1) = w_{jk}(t) + \alpha\delta_k z_j + \mu[w_{jk}(t) - w_{jk}(t-1)]$$

Dan

$$v_{ij}(t+1) = v_{ij}(t) + \alpha\delta_j x_i + \mu[v_{ij}(t) - v_{ij}(t-1)]$$

Keterangan:

$w_{jk}(t)$ = Bobot mula-mula pola kedua
$\alpha\delta_k z_j$ = Hasil dari perhitungan langkah ke-6 (Δw_{jk}).
$w_{jk}(t-1)$ = Bobot mula-mula pada data ke-1 iterasi pertama.
μ = Nilai momentum
$v_{ij}(t)$ = Bobot mula-mula pola kedua
$\alpha\delta_j x_i$ = Hasil dari perhitungan langkah ke-7 (Δv_{ij}).
$v_{ij}(t-1)$ = Bobot mula-mula pada data ke-1 iterasi pertama.

3. HASIL DAN PEMBAHASAN

3.1. Pra Pengolahan

Sampel data yang akan digunakan sebagai data *training* adalah 70 % dari total data keseluruhan mahasiswa program studi Sistem Informasi STMIK Banjarbaru tahun angkatan 2011, 2012, dan 2013, terdiri dari data gender, index prestasi semester 1, jumlah pengambilan satuan kredit semester 2, index prestasi semester 2, jumlah pengambilan satuan kredit semester 3, jumlah pengambilan satuan kredit semester 4, index prestasi semester 4, jumlah pengambilan satuan kredit semester 5, dan index prestasi semester 5. Sedangkan sampel data yang digunakan sebagai data *testing* 30 % dari total data keseluruhan mahasiswa program studi Sistem Informasi STMIK Banjarbaru tahun angkatan 2011, 2012, dan 2013.

Alur penelitian pada penelitian ini dimulai dengan proses *input dataset* yang dilanjutkan dengan pra pengolahan data, yang terdiri dari *set role* dan *normalize. Set Role* adalah operator pada *tool* Rapid Miner untuk mengatur peran yang akan digunakan. Kebanyakan operator klasifikasi tidak akan bekerja jika tidak ada atau bahkan jika ada lebih dari satu atribut dengan peran label di dalam *dataset* [11] Setelah melakukan *set role*, kemudian dilakukan normalisasi *dataset* untuk mengubah semua nilai atribut menjadi kisaran 0-1 dengan menggunakan operator *normalize* pada *tool* Rapid Miner.

3.2 Hasil Uji Coba

Uji coba dilakukan dengan menggunakan *tool* Rapid Minner versi 7.6 untuk menentukan parameter *Neural Network* berupa *training cycles*, *learning rate*, momentum, dan *hidden layer* terbaik bagi dataset pada penelitian ini.

Pada uji coba pertama, percobaan dilakukan untuk menentukan nilai *training cycles*, *nilai training cycles* (*epoch*) dalam penelitian ini ditentukan dengan cara melakukan uji coba memasukkan nilai dengan *range* 200 sampai dengan 2000 untuk *training cycles* (*epoch)*, serta nilai 0.1 untuk *learning rate* dan 0.1 untuk momentum, hasil dari uji coba dapat dilihat pada Tabel 1.

Tabel 1. Percobaan Penentuan Nilai Training Cycles

Training Cycles	*Learning Rate*	Momentum	Akurasi
200	0.1	0.1	97.94 %
400	0.1	0.1	97.94 %
600	0.1	0.1	97.94 %
800	0.1	0.1	97.94 %
1000	0.1	0.1	97.94 %
1200	0.1	0.1	98.97 %
1400	0.1	0.1	96.91 %
1600	0.1	0.1	96.91 %
1800	0.1	0.1	97.94 %
2000	0.1	0.1	97.94 %

Training cycles (*epoch*) dipilih berdasarkan nilai akurasi tertinggi yang dihasilkan. Berdasarkan hasil percobaan di atas, maka dipilih nilai *training cycles* (*epoch*) sebesar 1200. Nilai *training cycles* (*epoch*) ini selanjutnya dipakai untuk percobaan dalam menentukan *learning rate*.

Nilai *learning rate* ditentukan dengan cara melakukan uji coba memasukkan nilai dengan *range* 0.1 sampai dengan 1. Nilai *training cycles* (*epoch*) dipilih dari percobaan sebelumnya yaitu 1200, sedangkan nilai 0.1 digunakan untuk nilai momentum, hasil dari uji coba dapat dilihat pada Tabel 2.

Tabel 2. Percobaan Penentuan Learning Rate

Training Cycles (epoch)	*Learning Rate*	Momentum	Akurasi
1200	0.1	0.1	98.97 %
1200	0.2	0.1	97.94 %
1200	0.3	0.1	97.94 %
1200	0.4	0.1	97.94 %
1200	0.5	0.1	96.91 %
1200	0.6	0.1	97.94 %
1200	0.7	0.1	97.94 %
1200	0.8	0.1	97.94 %
1200	0.9	0.1	97.94 %
1200	1	0.1	97.94 %

Nilai *learning rate* dipilih berdasarkan nilai akurasi tertinggi yang dihasilkan. Berdasarkan hasil percobaan di atas, dipilih nilai *learning rate* sebesar 0.1, nilai ini akan digunakan untuk percobaan dalam menentukan nilai momentum.

Nilai momentum ditentukan dengan cara melakukan uji coba memasukkan nilai dengan *range* 0 sampai dengan 0.9. Nilai *training cycles* (*epoch*) dan *learning rate* dipilih dari percobaan sebelumnya yaitu 1200 dan 0.1, hasil uji dari uji coba dapat dilihat pada Tabel 3.

Tabel 3. Percobaan Penentuan Nilai Momentum

Training Cycles (epoch)	*Learning Rate*	Momentum	Akurasi
1200	0.1	0	97.94 %
1200	0.1	0.1	98.97 %
1200	0.1	0.2	96.91 %
1200	0.1	0.3	96.91 %
1200	0.1	0.4	96.91 %
1200	0.1	0.5	97.94 %
1200	0.1	0.6	97.94 %
1200	0.1	0.7	97.94 %
1200	0.1	0.8	97.94 %
1200	0.1	0.9	96.91 %

Berdasarkan hasil percobaan di atas, maka untuk parameter *neural network* dalam penelitian ini digunakan nilai 1200 untuk *training cycles* (*epoch*), 0.1 untuk *learning rate*, dan 0.1 untuk momentum.

Untuk penentuan jumlah hidden layer dan hidden node, dilakukan uji coba dengan menggunakan satu dan dua *hidden layer*, sedangkan untuk *hidden node* digunakan angka dengan *range* 1 sampai dengan 24. Hasil uji coba untuk satu hidden layer dapat dilihat pada Tabel 4.

Tabel 4. Percobaan Dengan Satu Hidden Layer

Jumlah Hidden Node	Akurasi	Jumlah Hidden Node	Akurasi
3	98.97 %	14	97.94 %
4	97.94 %	15	97.94 %
5	96.91 %	16	97.94 %
6	96.91 %	17	97.94 %
7	97.94 %	18	97.94 %
8	97.94 %	19	97.94 %
9	97.94 %	20	97.94 %
10	97.94 %	21	97.94 %
11	97.94 %	22	97.94 %
12	97.94 %	23	97.94 %
13	97.94 %	24	97.94 %

Hasil terbaik pada percobaan satu *hidden layer* yaitu *hidden layer* dengan jumlah *hidden node* 3 yang menghasilkan akurasi sebesar 98.97 %.

Untuk dua *hidden layer*, dilakukan sebanyak 27 arsitektur, hasil percobaan dapat dilihat pada tabel 5.

Tabel 5 Percobaan Dengan Dua Hidden Layer

Jumlah *Hidden Node Layer* 1	Jumlah *Hidden Node Layer* 2	Akurasi
1	1	97.94 %
1	2	97.94 %
1	3	97.94 %
1	4	96.91 %
1	5	97.94 %
1	6	97.94 %
1	7	97.94 %
1	8	97.94 %
1	9	97.94 %
2	1	96.91%
2	2	98.97 %
2	3	97.94 %
2	4	97.94 %
2	5	97.94 %
2	6	97.94 %
2	7	97.94 %
2	8	97.94 %
2	9	98.97 %
3	1	96.91 %
3	2	97.94%
3	3	97.94%
3	4	97.94%
3	5	97.94%
3	6	97.94%
3	7	97.94%
3	8	97.94%
3	9	97.94%

Percobaan untuk dua *hidden layer* tidak menghasilkan akurasi yang lebih tinggi dari satu *hidden layer* dengan jumlah *hidden node* 3.

Penentuan jumlah *input node* dalam penelitian ini ditentukan dengan cara melakukan percobaan sebanyak 3 kali, yang pertama menggunakan 6 *input nodes* (Gender, IPS1, SKS2, IPS2, SKS3, IPS3), yang kedua menggunakan 8 *input nodes* (Gender, IPS1, SKS2, IPS2, SKS3, IPS3, SKS4, IPS4), dan yang ketiga menggunakan 10 *input nodes* (Gender, IPS1, SKS2, IPS2, SKS3, IPS3, SKS4, IPS4, SKS5, IPS5).

Ketiga percobaan tersebut menggunakan parameter nilai 1200 untuk *training cycles* (*epoch*), 0.1 untuk *learning rate*, *hidden layer 1, hidden node 3,* dan 0.1 untuk momentum didapatkan hasil akurasi seperti pada Tabel 6.

Tabel 6. Percobaan Jumlah Input Node

Jumlah *Input Node*	*Training Cycles (epoch)*	*Learning Rate*	Momentum	Akurasi
6	1200	0.1	0.1	96.91 %
8	1200	0.1	0.1	97.94 %
10	1200	0.1	0.1	98.97 %

Berdasarkan percobaan di atas jumlah *input nodes* 10 menghasilkan akurasi paling besar yaitu 98.97, maka pada penelitian digunakan 10 *input nodes*.

Hasil penerapan *neural network* ke dalam aplikasi yang dibuat untuk memprediksi kelulusan mahasiswa STMIK Banjarbaru adalah seperti pada Tabel 7.

Tabel 7. Confusion Matrix

	true Terlambat	*true* Waktu	Tepat
pred. Terlambat	95	1	
pred. Tepat Waktu	0	1	

Dari total keseluruhan data *testing* yang digunakan sebanyak 97 data, 95 diantaranya mendapat hasil *true-positive* (prediksi terlambat dan kenyataan terlambat), 1 diantaranya *true-negative* (prediksi terlambat namun kenyataan tepat waktu), 0 diantaranya *false-negative* (prediksi tepat wakt namun kenyataan terlambat), dan 1 diantaranya *false-positive* (prediksi tepat waktu dan kenyataan tepat waktu).

Hasil akurasi dapat didapatkan dengan cara membagikan jumlah prediksi yang sesuai kenyataan (*true-positive*+*false-positive*) dengan jumlah data keseluruhan (97) kemudian dikalikan 100.

$$Akurasi = \frac{96}{97} \times 100 = 98.97$$

4. SIMPULAN

Dari hasil penelitian dapat diambil kesimpulan bahwa :

a. Parameter seperti jumlah *input node*s, *training cycles* (*epoch*) *learning rate*, momentum, jumlah *hidden layer*, dan jumlah *hidden node* dapat mempengaruhi akurasi dari algoritma *backpropagation*.
b. Karena data training yang digunakan dominan memiliki target terlambat daripada tepat waktu, menyebabkan algoritma yang diterapkan di aplikasi cenderung menghasilkan *output* terlambat.

c. Jumlah data *training* yang digunakan dapat mempengaruhi akurasi dan kerasionalan *output* algoritma *backpropagation*.
d. Berdasarkan penelitian yang dilakukan dan hasil dari implementasi algoritma *backpropagation* pada aplikasi yang dibuat, bahwa faktor yang paling mempengaruhi hasil prediksi adalah index prestasi semester (IPS) mahasiswa yang bersangkutan.
e. Pada proses prediksi dengan menggunakan aplikasi yang telah dibuat, prediksi kelulusan mahasiswa STMIK Banjarbaru dengan menggunakan 97 sampel data angkatan 2011, 2012 dan 2013 didapatkan akurasi sebesar 98.97 %.

DAFTAR PUSTAKA

[1] BAN.PT., "**Buku 6-Matriks Penilaian Akreditasi Sarjana"**, Jakarta: BAN.PT, 2008.

[2] STMIK Banjarbaru, "**Laporan Kelulusan Tahun 2015**", STMIK Banjarbaru, Banjarbaru, 2015.

[3] R. Ansari, "**Prediksi Kelulusan Mahasiswa Dengan Jaringan Syaraf Tiruan**", Jurnal Teknologi Informasi Universitas Lambung Mangkurat, vol. 1, pp. 18-23, 2016.

[4] M. A. Lestari Handayani, "**Penerapan JST (Backpropagation) untuk Prediksi Curah Hujan**", Seminar Nasional Teknologi Informasi, Komunikasi dan Industri, vol. 7, 2015.

[5] S. Redjeki, "**Perbandingan Algoritma Backpropagation dan K-Nearest Neighbor untuk Identifikasi Penyakit**", Seminar Nasional Aplikasi Teknologi Informasi, 2013.

[6] STMIK Banjarbaru, "**Peraturan Akademik**", Banjarbaru: STMIK Banjarbaru, 2017.

[7] M. K. Cowan, "**Machine Learning**", Standford: Standford University, 2013.

[8] E. Riyanto, "**Sistem Pengenalan Pengucap Manusia Dengan Ekstraksi Ciri MFCC dan Algoritma Jaringan Saraf Tiruan Perambatan Balik Sebagai Pengenalnya**", Pasca Sarjana Universitas Diponegoro, Surabaya, 2013.

[9] L. Fausett, "**Fundamentals of Neural Networks: Architectures, Algorithms, and Applications**", Upper Saddle River: Prentice-Hall, 1994.

[10] O. N. AL-Allaf, "**Improving the Performance of Backpropagation Neural Network**", Journal of Computer Science, vol. 6, pp. 1347-1354, 2010.

[11] R. S. W. Indah Suryani, "**Penerapan Exponential Smoothing untuk Transformasi Data dalam Meningkatkan Akurasi Neural Network**", Journal of Intelligent Systems, vol. 1, pp. 67-75, 2015.