PREDIKSI MASA STUDI MAHASISWA MATEMATIKA IPB BERDASARKAN INDEKS PRESTASI KUMULATIF MENGGUNAKAN JARINGAN SYARAF TIRUAN

*S. Nurdiati[1], F. Bukhari[2], M. K. Najib[3] dan K. Hilmi[4]

[1,2,3,4)]Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam,
Institut Pertanian Bogor, Jl. Meranti, Kampus IPB Dramaga Bogor.
nurdiati@apps.ipb.ac.id ***corresponding author**
fahrenbu@apps.ipb.ac.id, mkhoirun_najib@apps.ipb.ac.id, kautsarhilmi@yahoo.co.id

Abstrak

Akreditasi sebuah program studi sangat dipengaruhi oleh masa studi dan Indeks Prestasi Kumulatif (IPK) lulusannya. Beberapa penelitian menunjukkan adanya keterkaitan antara kelulusan dengan IPK mahasiswa. Namun, model prediksi lama masa studi berdasarkan IPK masih sedikit. Oleh karena itu, penelitian ini bertujuan untuk memprediksi masa studi mahasiswa berdasarkan IPK menggunakan model jaringan syaraf tiruan (JST) berbasis *backpropagation*. Beberapa fungsi pelatihan diterapkan, meliputi *gradient descent*, Nesterov *accelerated gradient descent*, *Adaptive moment estimation* (Adam), dan Nesterov Adam (Nadam). Data yang digunakan dalam penelitian ini adalah data masa studi dan IPK semester 1-6 mahasiswa S1 Matematika IPB. Hasil penelitian menunjukkan bahwa model JST terbaik dihasilkan oleh jaringan dengan jumlah *input node* 6 yang dinormalisasi dengan *batch normalization* (*BatchNorm*), *hidden node* 10 dan *output node* 1. Parameter jaringan terbaik diperoleh dari percobaan menggunakan fungsi pelatihan *gradient descent* dan laju pembelajaran 0.5 dengan MAE sebesar 1.887 pada data *testing*. Fungsi pelatihan *gradient descent* memperlihatkan adanya penurunan nilai MAE ketika nilai laju pembelajaran meningkat. Sementara itu, pada fungsi pelatihan lainnya, terdapat tren bahwa semakin kecil nilai laju pembelajaran maka semakin kecil pula nilai MAE yang dihasilkan. Berdasarkan model JST terpilih, nilai IPK yang paling berpengaruh pada masa studi mahasiswa matematika IPB adalah nilai IPK pada semester 3, yaitu masa mahasiswa matematika IPB pertama kali menerima mata kuliah mayor dari Departemen Matematika secara keseluruhan. Kepentingan dari fitur ini sangat tinggi, mencapai 75.62%. Model JST terpilih menghasilkan MAPE sebesar 3.8% dan RMSPE sebesar 4.9% pada data *testing*.

Kata kunci: *backpropagation*, masa studi, *neural network*, prediksi.

1 Pendahuluan

Perguruan tinggi dituntut untuk memiliki keunggulan bersaing dan kualitas yang baik. Keberhasilan sistem pendidikan dalam suatu perguruan tinggi diukur dari peningkatan kualitas pendidikan dari tahun ke tahun. Standar Nasional Pendidikan Tinggi (SN Dikti) merupakan tolok ukur kualitas pendidikan tersebut dan salah satu kriterianya adalah kelulusan mahasiswa [16]. Kelulusan merupakan hasil akhir dari proses pembelajaran setelah mengikuti kuliah pada suatu perguruan tinggi.

2020 Mathematics Subject Classification: 68T07.
Diajukan: 30/5/2022, diterima: 22/7/2022.

Dalam proses belajar, mahasiswa mendapatkan nilai setiap semester berupa Indeks Prestasi (IP) dan hasil belajar secara keseluruhan dapat dilihat dari Indeks Prestasi Kumulatif (IPK). Permenristek Dikti Nomor 56 tahun 2017 mendefinisikan IPK sebagai hasil penilaian capaian pembelajaran lulusan pada akhir program studi. Mahasiswa dapat mengevaluasi proses belajar untuk mengukur kemampuannya berdasarkan IPK yang diperoleh.

Selain proses belajar, mahasiswa juga wajib mengerjakan penelitian tugas akhir berupa skripsi sebagai syarat menyelesaikan pendidikan sarjananya. Tidak dapat dipungkiri, masih terdapat perbedaan waktu dalam penyelesaian skripsi pada setiap mahasiswa. Kecepatan penyelesaian atau kelulusan tersebut diasumsikan memiliki keterkaitan dengan IPK [7,18]. Artinya, mahasiswa yang memiliki IPK tinggi mampu menyelesaikan masa studinya lebih cepat. Asrib [1] dan Budiarti [3] telah membuktikan bahwa terdapat keterkaitan antara kelulusan dengan Indeks Prestasi Kumulatif (IPK) mahasiswa. Artinya, mahasiswa bisa diprediksi masa studinya berdasarkan IPK yang diperoleh.

Akreditasi sebuah program studi sangat dipengaruhi oleh masa waktu studi dan IPK lulusannya. Keduanya merupakan indikator aspek pembelajaran menurut Peraturan Badan Akreditasi Nasional Perguruan Tinggi (BAN-PT) Nomor 3 tahun 2019 tentang instrumen akreditasi perguruan tinggi. Dalam laporan kinerja perguruan tinggi IPB pada tahun 2017, baru sekitar 32,2% jumlah lulusan program sarjana yang lulus tepat waktu. Sementara itu, IPB menargetkan capaian kelulusan tepat waktu mahasiswa program sarjana adalah 60% dari total jumlah lulusan.

Terdapat banyak penelitian yang telah mempelajari tentang keterkaitan IPK dan masa studi. Nurdiati dan Najib [19] menggunakan metode regresi berbasis *machine learning* untuk memprediksi masa studi berdasarkan IPK. Mawarni [14] melakukan prediksi IPK dan masa studi calon mahasiswa baru secara terpisah menggunakan *backpropagation*. *Backpropagation* merupakan salah satu metode dalam Jaringan Saraf Tiruan (JST) dengan pelatihan jenis terkontrol dengan pola penyesuaian bobot untuk mencapai nilai kesalahan yang minimum antara keluaran hasil prediksi dengan keluaran nyata [12]. Model JST memiliki kemampuan dalam emulasi, analisis, prediksi dan asosiasi [23]. Ibrahim [10] melakukan prediksi kinerja akademik mahasiswa dengan membandingkan tiga metode, yakni jaringan saraf tiruan, pohon keputusan, dan regresi linear. Hasilnya menunjukkan bahwa jaringan saraf tiruan mengungguli kedua metode yang lain. Menurut Simbolon [25], *backpropagation* merupakan algoritma populer dengan nilai RMSE terbaik untuk teknik prediksi.

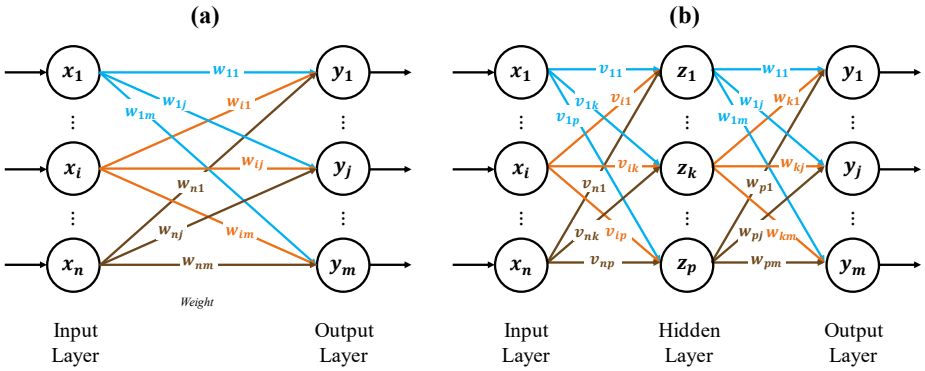
Berdasarkan uraian di atas, penelitian ini mengembangkan suatu model prediksi masa studi mahasiswa S1 matematika IPB berdasarkan IPK menggunakan model JST. Algoritma *backpropagation* diterapkan untuk memprediksi masa studi mahasiswa matematika IPB berdasarkan IPK dengan beberapa fungsi pelatihan, yaitu *gradient descent*, Nesterov *accelerated gradient descent*, *Adaptive moment estimation* (Adam), dan Nesterov Adam. Normalisasi batch (*batch normalization*) digunakan untuk mempercepat proses pelatihan model jaring syaraf tiruan. Penelitian ini membandingkan akurasi prediksi yang dihasilkan oleh masing-masing metode pelatihan tersebut dengan jumlah node pada *hidden layer* dan laju pembelajaran yang berbeda-beda. Terakhir, model JST terpilih digunakan untuk mengukur tingkat kepentingan dari masing-masing fitur (*feature importance*) dan diuji menggunakan data *testing* untuk mengukur performa model JST.

2 Metode Penelitian

2.1 Jaringan Syaraf Tiruan

Jaringan Saraf Tiruan (JST) atau disebut juga *Artificial Neural Networks* (ANN) adalah serangkaian sistem komputasi (*computing systems*) yang terinspirasi dari jaringan syaraf biologis pembentuk otak manusia maupun hewan [2]. JST terdiri dari beberapa elemen pemrosesan yang menerima *input* dan memberikan *output* berdasarkan fungsi aktivasi yang telah ditentukan sebelumnya [6]. Elemen pemrosesan tersebut sering disebut dengan unit, atau node yang merepresentasikan suatu neuron pada sistem jaringan syaraf otak. Setiap node terhubung ke node lain melalui tautan komunikasi terarah, masing-masing dengan bobot (*weight*) yang terkait [8].

Berdasarkan jumlah lapisan, struktur JST dapat dibedakan menjadi dua jenis, yaitu *single* dan *multi layer* (Gambar 1). Jaringan *single layer* hanya memiliki satu layer yang terboboti. Dengan demikian, setiap node pada jaringan *single layer* hanya dikelompokkan ke dalam *input layer* atau *output layer*. Gambar 1a menunjukkan JST *single layer* dengan *n* node *input* dan *m* node *output*. Sementara itu, pada jaringan *multi-layer*, terdapat *hidden layer* di antara *input* dan *output layer*. Suatu *hidden layer* terdiri atas beberapa *hidden node* dan suatu JST bisa memiliki satu atau lebih *hidden layer*. Gambar 1b menunjukkan JST *multi layer* dengan satu *hidden layer* yang terdiri atas *p hidden node*.

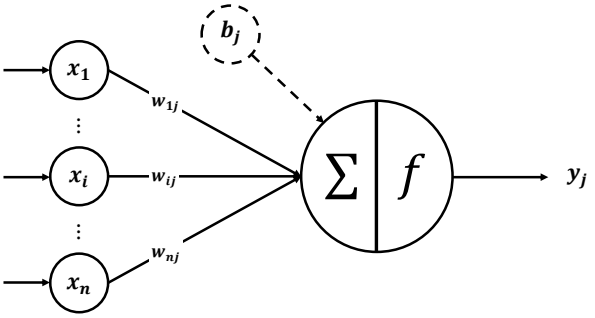


Gambar 1. Ilustrasi arsitektur JST: (a) single layer, dan (b) multi-layer.

Setiap elemen pemrosesan (node *hidden* dan *output*) pada suatu JST memiliki keadaan internal yang disebut aktivitas. Aktivitas dari setiap elemen pemrosesan tersebut terdiri atas penjumlahan dari setiap node pada layer sebelumnya yang telah diboboti dan bias (jika diperlukan) [8]. Selanjutnya, hasil penjumlahan tersebut ditransformasi dengan suatu fungsi aktivasi yang telah dipilih, seperti fungsi identitas, sigmoid, ReLU, dan sebagainya (Tabel 1). Sebagai ilustrasi, aktivitas yang dilakukan oleh node y_j (Gambar 1a) dapat dilihat pada Gambar 2 berikut. Dengan demikian, aktivitas pada node y_j adalah sebagai berikut.

$$y_j = f\left(\sum_{i=1}^{n} w_{ij}x_i + b_j\right) \tag{1}$$

dengan x_i adalah node input ke-i, w_{ij} adalah bobot dari input x_i ke y_j, b_j adalah bias untuk node y_j, dan $f(\cdot)$ merupakan fungsi aktivasi yang digunakan.



Gambar 2. Ilustrasi aktivitas yang terjadi pada node y_j.

Tabel 1. Beberapa fungsi aktivasi pada jaringan syaraf tiruan.

Fungsi aktivasi	**Persamaan $f(x)$**	***Range***
Identitas/Linear	x	$[-\infty, \infty]$
Rectified Linear Unit (ReLU)	$x^+ = \max(0, x)$	$[0, \infty]$
Sigmoid	$(1 + e^{-x})^{-1}$	$[0,1]$
Sigmoid bipolar	$(1 - e^{-x})(1 + e^{-x})^{-1}$	$[-1,1]$
Tangen hiperbolik	$(e^x - e^{-x})(e^x + e^{-x})^{-1}$	$[-1,1]$

Meskipun aturan dari JST cukup jelas, namun banyaknya *hidden node* maupun *hidden layer* dari suatu arsitektur JST untuk menyelesaikan masalah regresi maupun klasifikasi tidaklah pasti [20]. Jika *hidden layer* dan *hidden node* terlalu sedikit, maka JST tidak mampu menguraikan masalah dengan baik. Sebaliknya, semakin banyak *hidden layer* dan *hidden node*, maka arsitektur JST akan memiliki parameter bobot yang semakin banyak, sehingga arsitektur JST membutuhkan penyimpanan yang banyak dan waktu komputasi yang lama dalam proses penyesuaian bobot [21]. Selain itu, arsitektur JST dengan *hidden layer* dan *hidden node* yang terlalu banyak akan memiliki risiko adanya *overfitting* pada proses pelatihan.

Tugas utama dalam menggunakan JST adalah menyesuaikan bobot dari arsitektur JST yang telah ditetapkan. Proses penyesuaian bobot ini disebut sebagai proses pelatihan (*learning process*). Proses pelatihan merupakan satu-satunya cara untuk JST dalam menyimpan informasi, sehingga JST yang tidak dilatih tidak dapat digunakan. Adapun algoritma pelatihan yang umum dan popular digunakan untuk pelatihan jaringan syaraf tiruan adalah algoritma *backpropagation*.

2.2 Algoritma *Backpropagation*

Algoritma *backpropagation* (*backward propagation of errors*) merupakan algoritma klasik untuk melatih suatu JST *multi layer* [4]. Kelebihan algoritma *backpropagation* adalah menyediakan metode sistematis untuk menentukan kesalahan (*error*) pada *hidden node*. Setelah kesalahan pada *hidden node* ditentukan, bobot dari masing-masing node diperbaiki menggunakan metode optimasi atau fungsi pelatihan (*learning function*), seperti *gradient descent* [24]. Dengan demikian, algoritma *backpropagation* secara umum terdiri atas tiga tahap [9], yaitu: prosedur umpan maju (*feedforward*), perambatan balik kesalahan (*backpropagation error*) dan penyesuaian bobot. Algoritma 1 menunjukkan *pseudocode* dari *backpropagation* dalam melatih suatu model jaringan syaraf tiruan.

Algoritma 1. *Backpropagation* untuk perbaikan bobot jaringan syaraf tiruan

Dibutuhkan: arsitektur JST, data *training*, data *testing*
Inisiasi: nilai bobot dari setiap node secara acak
Definisikan: maksimum epoch,
Definisikan: fungsi biaya (*cost function*), fungsi pelatihan dan laju pembelajaran.
***While*:** maksimum epoch
  For *i* pada data *training*
    Lakukan prediksi menggunakan arsitektur JST
  End For
  Hitung nilai fungsi biaya (kesalahan) antara data *training* dan prediksi
  Lakukan perambatan balik (*backpropagation*) kesalahan
  Perbarui nilai bobot setiap node dengan suatu fungsi pelatihan dan laju pembelajaran
End While
Return model dengan parameter yang memiliki kesalahan paling minimum pada data *testing*.

Terdapat banyak jenis fungsi pelatihan yang dapat digunakan pada algoritma *backpropagation*. Salah satu yang sederhana dan sering digunakan adalah *gradient descent*. *Gradient descent* memberikan pembaruan nilai bobot dan bias (jika diperlukan) pada node ke-j melalui persamaan

$$w_{ij}^{*} = w_{ij} - \eta \cdot \frac{\partial C}{\partial w_{ij}}, \qquad b_j^{*} = b_j - \eta \cdot \frac{\partial C}{\partial b_j} \tag{2}$$

dengan η adalah laju pembelajaran yang ditetapkan, C adalah fungsi biaya, w_{ij} dan w_{ij}^{*} adalah bobot dari node ke-i ke node ke-j sebelum dan setelah pembaruan, serta b_j dan b_j^{*} adalah bias pada node ke-j sebelum dan setelah pembaruan [22]. Selain itu, fungsi pelatihan lain yang digunakan yaitu Nesterov *Accelarated Gradient descent* (NAG) [17], *Adaptive moment estimation* (Adam) [13], dan Nesterov-Adam (Nadam) [5].

Laju pembelajaran (*learning rate*) η biasanya ditentukan sebelum proses pelatihan dimulai. Semakin kecil nilai η, maka proses pelatihan akan mempelajari informasi yang diberikan oleh data *training* dengan semakin detail. Namun, jika nilai η terlalu kecil, proses pembelajaran akan memakan waktu yang sangat lama.

Sementara itu, fungsi biaya (*cost function*), yang bisa disebut juga *loss function*, merupakan fungsi yang digunakan untuk mengukur kesalahan prediksi dari JST terhadap data aktual. Pada kasus regresi, fungsi biaya yang sering digunakan adalah *root mean squared error* (RMSE) dan *mean absolute error* (MAE). Sementara pada kasus klasifikasi, JST dapat menggunakan fungsi *zero-one loss*, *cross-entropy*, dan sebagainya. Pada penelitian ini, fungsi biaya yang digunakan adalah MAE dengan persamaan, yaitu

$$C(\hat{y}, y) = \frac{1}{n}\sum_{i=1}^{m} |\hat{y}_i - y_i| \tag{3}$$

dengan *n* adalah banyaknya data, serta $\hat{y}_i$ dan y_i adalah data prediksi dan aktual ke-i.

2.3 *Batch Normalization*

Normalisasi *batch* (*batch normalization*) digunakan untuk mengurangi pergeseran kovariat internal sehingga dapat mempercepat pelatihan suatu model jaring syaraf tiruan. Normalisasi *batch* memperbaiki *mean* dan varian pada lapisan *input* serta mereduksi ketergantungan *gradient* terhadap nilai awal parameter, sehingga dimungkinkan untuk menggunakan laju pembelajaran yang tinggi tanpa risiko adanya divergensi. Terdapat dua langkah dalam normalisasi *batch*. Langkah pertama normalisasi *batch* adalah untuk suatu lapisan berdimensi d (biasanya pada lapisan *input*) yaitu $x = \left(x^{(1)}, x^{(2)}, \dots, x^{(d)}\right)$, setiap dimensi dinormalisasi menggunakan persamaan

$$\hat{x}^{(k)} = \frac{x^{(k)} - E\left[x^{(k)}\right]}{\sqrt{Var[x^{(k)}]}} \tag{4}$$

dengan nilai ekspektasi $E\left[x^{(k)}\right]$ dan varian $Var\left[x^{(k)}\right]$ dari data *training*. Normalisasi ini mempercepat konvergensi, bahkan ketika fitur-fiturnya tidak memiliki keterkaitan.

Perhatikan bahwa dengan menormalkan setiap *input* dari sebuah lapisan dapat mengubah apa yang dapat direpresentasikan oleh layer tersebut. Untuk mengatasi hal ini, langkah kedua dari normalisasi *batch* diperkenalkan yaitu dengan menskalakan dan menggeser nilai yang ternormalisasi $\hat{x}^{(k)}$,

$$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)} \tag{5}$$

dengan $y^{(k)}$ adalah *output* dari normalisasi *batch*, $\gamma^{(k)}$ dan $\beta^{(k)}$ keduanya merupakan parameter skala dan pergeseran untuk setiap *input* $\hat{x}^{(k)}$ yang telah ternormalisasi [11].

2.4 Data dan Tahapan Penelitian

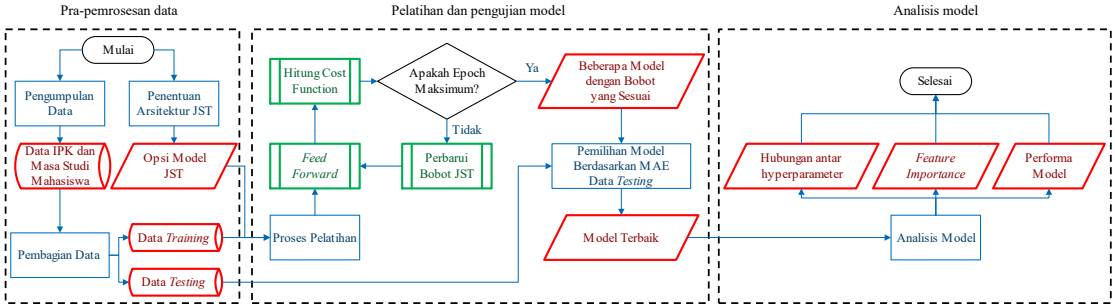
Terdapat dua jenis data yang digunakan pada penelitian ini yaitu data *training* dan data *testing*. Data *training* digunakan untuk membentuk model JST menggunakan algoritma *backpropagation*. Sementara itu, data *testing* digunakan untuk menguji model yang terbentuk. Data *training* berasal dari mahasiswa Matematika S1 IPB Angkatan 50 dan 51, sedangkan data *testing* berasal dari mahasiswa Matematika S1 IPB Angkatan 52 dan 53 yang telah lulus pada tahun 2019-2020. Data yang dikumpulkan adalah data indeks prestasi (IP) mahasiswa dari semester 1 sampai 6 (dinotasikan oleh $X_1, X_2, \dots, X_6$) dan data masa studi mahasiswa (dinotasikan oleh Y).

Penelitian ini diawali dengan mengumpulkan data yang telah dideskripsikan sebelumnya. Setelah itu, arsitektur JST dikonstruksi menggunakan spesifikasi seperti pada Tabel 2. Keseluruhan proses ini disebut dengan tahap *pre-processing* data.

Setelah tahap *pre-processing* data dilakukan, tahap selanjutnya adalah melatih model yang diusulkan menggunakan metode *backpropagation* (Algoritma 1) dengan beberapa *hyper-parameter*, seperti fungsi pelatihan, node *hidden*, dan laju pembelajaran. Dengan model JST tersebut, masa studi pada data *testing* diprediksi, kemudian dipilih satu model JST yang memiliki nilai akurasi yang paling baik. Lebih lanjut, hasil pembelajaran model JST dianalisis untuk melihat hubungan antar *hyper-parameter*. Selain itu, model terpilih juga akan dianalisis untuk mengetahui karakteristik nilai prediksi yang dihasilkan. Untuk lebih jelasnya, Gambar 3 menunjukkan diagram alur dari penelitian ini.

Tabel 2. Spesifikasi jaringan syaraf tiruan yang digunakan

Karakteristik	**Spesifikasi**
Arsitektur jaringan	*Multilayer* dengan 1 *hidden layer*
Jumlah *node input*	6 ($X_1, X_2, \ldots, X_6$)
Normalisasi *input*	*Batch normalization*
Jumlah *node output*	1 (Y)
Jumlah *node hidden layer*	5 / 10 / 15
Fungsi aktivasi *hidden layer*	Sigmoid bipolar
Fungsi aktivasi *output layer*	Identitas
Algoritma pembelajaran	*Backpropagation*
Fungsi pelatihan	*Gradient Descent* (GD) / Nesterov *Accelarated Gradient Descent* (NAG) / *Adaptive Moment Estimation* (Adam) / Nesterov-Adam (Nadam)
Fungsi biaya	*Mean Absolute Error* (MAE)
Laju pembelajaran	0.5 / 0.1 / 0.05 / 0.01
Maksimum *epoch*	1000



Gambar 3. Diagram alur penelitian.

3 Hasil dan Pembahasan

3.1 Pra-pemrosesan Data

Model JST dikonstruksi dan dilatih pada bahasa pemrograman Julia menggunakan paket *Flux.jl*. Flux adalah *library* untuk *machine learning* pada bahasa pemrograman Julia yang diarahkan untuk *pipeline* produksi berperforma tinggi. Adapun konstruksi model JST dan jumlah parameter yang dihasilkan dapat dilihat pada Tabel 3 berikut. Setiap fungsi tersebut dihubungkan menggunakan fungsi *Chain* pada Julia.

Tabel 3. Konstruksi model JST menggunakan paket *Flux.jl* pada bahasa pemrograman Julia beserta jumlah parameter yang dihasilkan dari masing-masing lapisan.

Jenis lapisan	**Jumlah node**	**Fungsi pada Julia**	**Jumlah parameter**
Input	6	-	-
Batch normalization	6	BatchNorm(6)	6 (γ) + 6 (β)
Hidden[*)]	n	Dense(6, n, sig_bipolar)	$6 \times n$ (bobot) + n (bias)
Output	1	Dense(n,1)	$n \times 1$ (bobot) + 1 (bias)

[*)] *lapisan menggunakan fungsi sig_bipolar yang didefinisikan berdasarkan Tabel 1*

3.2 Pelatihan dan Pengujian Model JST

Tahap selanjutnya adalah melatih model JST yang didefinisikan menggunakan data *training*. Beberapa *hyper-parameter* seperti fungsi pelatihan (GD, NAG, Adam, dan Nadam), laju pembelajaran ($\eta = 0.01, 0.05, 0.1, 0.5$), dan jumlah node *hidden layer* $(5, 10, 15)$, digunakan untuk memperoleh model yang baik. Setiap model JST yang sudah dilatih kemudian diuji menggunakan data *testing*. Karena proses pelatihan model bergantung pada nilai inisiasi parameter yang ditentukan secara acak, proses pelatihan setiap model JST direpetisi sebanyak 10 kali. Model dipilih berdasarkan nilai MAE yang paling kecil pada data *testing*. Hasil pelatihan dan pengujian model untuk setiap *hyper-parameter* dapat dilihat pada Tabel 4 berikut. Dari 10 kali repetisi, Tabel 4 menampilkan nilai MAE minimum dari prediksi model JST terpilih menggunakan data *training* dan *testing* untuk setiap kombinasi *hyper-parameter* yang ditetapkan. Nilai yang dicetak tebal merupakan nilai optimum dari masing-masing fungsi pelatihan.

Tabel 4. Nilai *mean absolute error* (MAE) dari model JST terpilih untuk setiap *hyperparameter* pada data *training* dan *testing*.

η	Node	GD		NAG		Adam		Nadam	
		Train	***Test***	***Train***	***Test***	***Train***	***Test***	***Train***	***Test***
0.5	5	5.030	1.903	5.876	2.391	5.876	2.382	5.959	2.384
0.5	10	4.996	**1.887**	5.881	2.380	5.941	2.396	5.876	2.394
0.5	15	<u>4.571</u>	1.961	5.924	2.382	6.827	2.526	5.876	2.382
0.1	5	5.181	2.195	5.876	2.383	5.876	2.382	5.842	2.388
0.1	10	5.263	2.146	5.876	2.380	5.149	2.257	5.789	2.312
0.1	15	4.935	2.193	5.877	2.381	4.812	**2.027**	5.120	**2.090**
0.05	5	5.391	2.274	5.546	**2.169**	5.009	2.244	5.285	2.197
0.05	10	5.255	2.372	5.876	2.381	<u>4.736</u>	2.214	4.684	2.266
0.05	15	5.199	2.293	5.088	2.306	4.750	2.249	<u>4.456</u>	2.164
0.01	5	6.114	2.381	5.219	2.258	5.873	2.380	5.865	2.380
0.01	10	5.863	2.378	<u>5.040</u>	2.321	5.364	2.336	5.290	2.301
0.01	15	5.696	2.370	5.203	2.348	5.269	2.311	5.005	2.322

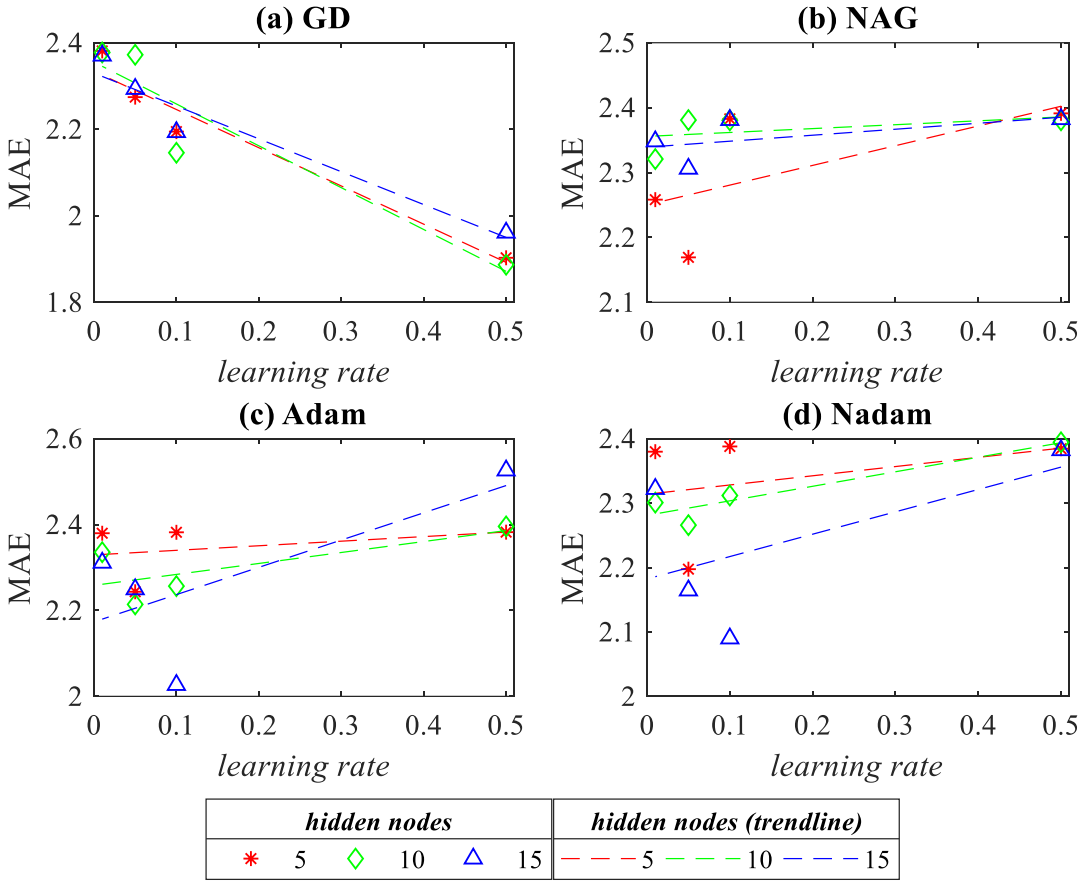
Secara umum, model yang dihasilkan dari setiap kombinasi *hyper-parameter* memiliki nilai MAE pada data *testing* antara 1.887 hingga 2.394. Fungsi pelatihan GD menghasilkan kesalahan terkecil ketika menggunakan laju pembelajaran 0.5 pada model JST yang memiliki lapisan *hidden* sebanyak 10 node. Nilai MAE yang dihasilkan pada data *training* dan *testing* masing-masing adalah 4.996 dan 1.887. Model JST yang dilatih menggunakan GD ini adalah yang paling baik dibandingkan ketiga fungsi pelatihan lainnya. Fungsi pelatihan Adam dan Nadam menghasilkan kesalahan terkecil ketika menggunakan laju pembelajaran 0.1 pada model JST yang memiliki lapisan *hidden* sebanyak 15 node. Nilai MAE yang dihasilkan pada data *testing* masing-masing adalah 2.027 dan 2.090. Sementara itu, fungsi pelatihan NAG menghasilkan kesalahan terkecil ketika menggunakan laju pembelajaran 0.05 pada model JST yang memiliki lapisan *hidden* sebanyak 5 node dengan nilai MAE yang dihasilkan pada data *testing* adalah 2.169.

Selain itu, nilai yang bergaris bawah menunjukkan nilai MAE terkecil pada proses pelatihan. Pada fungsi pelatihan GD, nilai yang paling kecil diperoleh ketika

menggunakan laju pembelajaran 0.5 pada model JST yang memiliki lapisan *hidden* sebanyak 15 node. Meskipun nilai MAE pada data *training* lebih kecil dibandingkan model serupa dengan 10 node *hidden*, tetapi nilai MAE pada data testing *dari* model dengan 15 node *hidden* lebih besar dibandingkan model dengan 10 node hidden. Ini menunjukkan adanya *overfitting* pada model dengan 15 node *hidden*, sehingga nilai MAE pada data *testing* bukan yang terkecil, meskipun nilai MAE pada data *training*-nya minimum.

3.3 Analisis Kesalahan

Analisis kesalahan dimaksudkan untuk mengetahui *trendline* perubahan MAE pada data *testing* apabila terdapat perubahan pada laju pembelajaran (Gambar 4). Gambar 4a menunjukkan *trendline* fungsi pelatihan GD yang memperlihatkan adanya penurunan nilai MAE ketika adanya peningkatan nilai laju pembelajaran. Artinya, model JST pada penelitian ini dapat dilatih dengan baik menggunakan laju pembelajaran yang besar. Sebaliknya, pada fungsi pelatihan lainnya, terdapat tren bahwa semakin kecil nilai laju pembelajaran yang diberikan maka semakin kecil nilai MAE yang dihasilkan.



Gambar 4. Trendline MAE berdasarkan laju pembelajaran (learning rate) untuk setiap jumlah hidden node yang digunakan.

3.4 Performa Model JST Terpilih

Model JST dengan MAE terbaik diperoleh pada model JST dengan jumlah node pada *hidden layer* sebanyak 10 node, yang dilatih menggunakan fungsi pelatihan GD dan laju pembelajaran 0.5. Adapun spesifikasi dan performa model yang dipilih adalah sebagai berikut.

1. *Batch Normalization*

Batch normalization adalah lapisan pertama dari model JST yang diusulkan. Terdapat dua jenis parameter yaitu γ dan β yang masing-masingnya terdiri atas enam elemen, yaitu

$$\begin{aligned}\gamma &= [1.44189 \quad 1.35901 \quad 1.99382 \quad 0.697085 \quad 2.73135 \quad -0.652675]^T \\ \beta &= [-1.45178 \quad 0.442606 \quad 1.33695 \quad 0.233569 \quad -1.32246 \quad 0.332607]^T\end{aligned} \tag{6}$$

dengan nilai ekspektasi dan varian dari enam *input* data *training* yaitu

$$\begin{aligned}E\left[x^{(k)}\right] &= [3.555 \quad 3.403 \quad 3.352 \quad 3.333 \quad 3.293 \quad 3.214]^T \\ Var\left[x^{(k)}\right] &= [0.099 \quad 0.130 \quad 0.164 \quad 0.167 \quad 0.172 \quad 0.198]^T\end{aligned} \tag{7}$$

Berdasarkan nilai ekspektasi yang dihasilkan, IPK mahasiswa matematika IPB pada data *training* selalu mengalami penurunan dari semester ke semester. Sementara itu, nilai variansnya selalu meningkat dari semester ke semester. Setiap data *input* yang masuk ke dalam lapisan *BatchNorm* dinormalisasi berdasarkan Persamaan 5.

2. Fitur Penting (*Feature Importance*)

Karena model JST terpilih memiliki 10 node pada lapisan *hidden*, jumlah parameter yang dihasilkan pada lapisan *hidden* adalah 70 parameter (bobot dan bias) dan lapisan *output* adalah 11 parameter (bobot dan bias). Akan sangat sulit untuk menganalisis parameter dari setiap lapisan, tetapi sangat menarik untuk menilai kepentingan dari setiap fitur berdasarkan model JST terpilih.

Konsepnya adalah mengukur pentingnya fitur dengan menghitung peningkatan kesalahan prediksi model setelah mengubah fitur tersebut. Sebuah fitur dikatakan "penting" jika pengacakan nilai fitur akan meningkatkan kesalahan model, karena dalam hal ini model mengandalkan fitur tersebut untuk prediksi. Sebuah fitur dikatakan "tidak penting" jika pengacakan nilai fitur membuat kesalahan model tidak berubah, karena dalam hal ini model mengabaikan fitur tersebut untuk prediksi [15]. Adapun langkah-langkah dalam menghitung tingkat kepentingan fitur pada model JST terpilih adalah sebagai berikut.

a. Hitung nilai prediksi masa studi mahasiswa menggunakan model JST terpilih dengan data *testing* yang telah diperbarui nilainya secara acak pada fitur yang ingin diukur kepentingannya. Nilai diacak pada selang 0 sampai 4, karena nilai IPK berada pada selang tersebut.
b. Hitung nilai kesalahan prediksi masa studi mahasiswa tersebut terhadap nilai aktual masa studi.
c. Ulangi tahapan a. dan b. sebanyak 100 repetisi kemudian hitung nilai rata-rata dari kesalahan yang dihasilkan.
d. Hitung peningkatan kesalahan yang dihasilkan terhadap kesalahan dari prediksi model JST terpilih menggunakan data *testing*. Model JST terpilih memiliki nilai kesalahan pada *testing* sebesar 1.887 (Tabel 4).

e. Hitung persentase kepentingan fitur.

Tabel 5 menunjukkan hasil pengukuran tingkat kepentingan dari setiap fitur pada model JST terpilih.

Tabel 5. Tingkat kepentingan fitur (*feature importance*) berdasarkan model JST terpilih.

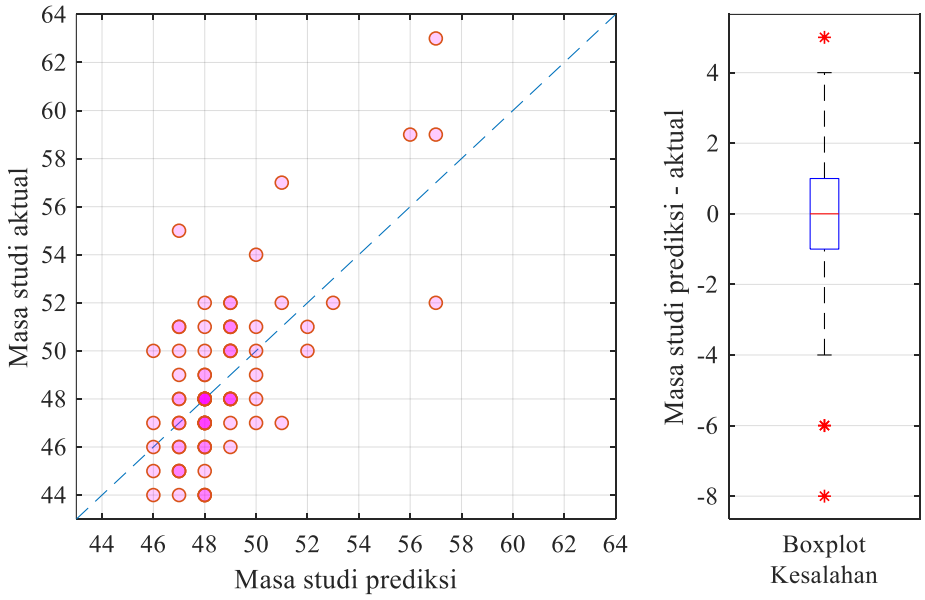
Semester	Rata-rata kesalahan	Rata-rata peningkatan kesalahan	Persentase kepentingan fitur*)
1	2.44	0.55	0.90 %
2	7.88	5.99	9.81 %
3	**48.04**	**46.16**	**75.62 %**
4	2.83	0.95	1.56 %
5	9.18	7.29	11.94 %
6	1.99	0.10	0.16 %

*) *Semakin tinggi nilai persentase, maka semakin penting fitur tersebut bagi model JST.*

Berdasarkan Tabel 5, nilai IPK yang paling berpengaruh pada masa studi mahasiswa matematika IPB adalah pada semester 3. Pada semester ini, mahasiswa matematika IPB beralih dari Program Pendidikan Kompetensi Umum (PPKU) yang mempelajari mata kuliah secara umum ke program pendidikan pada Departemen Matematika yang mempelajari mata kuliah pada program studi S-1 Matematika. Kepentingan dari fitur ini sangat tinggi, mencapai 75.62%.

3. Performa Model JST Terpilih

Gambar 5 menunjukkan perbandingan masa studi prediksi dan aktual berdasarkan model JST terpilih beserta boxplot dari selisih keduanya.



Gambar 5. Perbandingan hasil prediksi masa studi pada data testing menggunakan model JST terhadap data aktual, beserta box plot dari selisih keduanya.

Berdasarkan *boxplot* kesalahan, model JST terpilih menghasilkan kesalahan yang menyebar normal pada jangkauan interkuantil dari -4 hingga 4. Nilai median kesalahan bernilai nol, serta nilai Q1 dan Q3 masing-masing bernilai -1 dan 1. Terdapat tiga pencilan kesalahan., *underestimated* yaitu -6 dan -8 bulan serta *overestimated* 5 bulan. Seperti yang telah ditunjukkan sebelumnya, nilai MAE dari model prediksi pada data *testing* adalah 1.887. Sementara itu, ukuran performa model lainnya disajikan pada Tabel 6. Berdasarkan Tabel 6, model JST terpilih memiliki akurasi yang baik dengan MAPE dan RMSPE kurang dari 5%.

Tabel 6. Ukuran performa model JST terpilih.

Nama ukuran performa	**Ukuran performa model**
Mean Absolute Error (MAE)	1.887 bulan
Root Mean Squared Error (RMSE)	2.469 bulan
Mean Absolute Proportional Error (MAPE)	3.799 %
Root Mean Squared Proportional Error (RMSPE)	4.857 %

4 Simpulan dan Saran

Berdasarkan hasil penelitian yang telah dilakukan dapat disimpulkan bahwa arsitektur model JST terbaik dihasilkan oleh jaringan dengan jumlah *input node* 6 yang dinormalisasi dengan *batch normalization* (*BatchNorm*), *hidden node* 10 dan *output node* 1. Parameter jaringan terbaik diperoleh dari percobaan menggunakan fungsi pelatihan *gradient descent* dan laju pembelajaran 0.5 dengan MAE sebesar 1.887. Fungsi pelatihan *gradient descent* memperlihatkan adanya penurunan nilai MAE ketika nilai laju pembelajaran meningkat. Sebaliknya, pada fungsi pelatihan lainnya, terdapat tren bahwa semakin kecil nilai laju pembelajaran maka semakin kecil pula nilai MAE yang dihasilkan. Berdasarkan model JST terpilih, nilai IPK yang paling berpengaruh pada masa studi mahasiswa matematika IPB adalah nilai IPK pada semester 3. Kepentingan dari fitur ini sangat tinggi, mencapai 75.62%. Model JST terpilih menghasilkan MAPE sebesar 3.8% dan RMSPE sebesar 4.9%.

Beberapa saran untuk penelitian lebih lanjut adalah perlu dilakukannya penelitian lanjutan untuk melatih JST dengan berbagai tipe data simulasi dan membandingkannya dengan teknik regresi atau klasifikasi data yang lain. Selain itu, pengujian terhadap kombinasi arsitektur dan parameter untuk model JST lain perlu dilakukan, seperti menambahkan jumlah *hidden layer* lebih dari satu *layer*, sehingga model JST mampu menangkap karakteristik data lebih dalam. Dengan pengujian yang lebih banyak, diharapkan parameter jaringan saraf tiruan yang lebih optimal dapat ditemukan.

Daftar Pustaka

[1] Asrib AR, Haedir. 2017. Analisis Hubungan Indeks Prestasi Kumulatif (IPK) Lulusan dan Lama Studi Jurusan Pendidikan Teknik sipil dan Perencanaan Fakultas Teknik Universitas Negeri Makassar. Di dalam: *Seminar Nasional Fakultas Teknik Universitas Negeri Makassar*. hlm: 32-36. [diakses 2022 Mei 20]. http://eprints.unm.ac.id/5800/10/6%20Ahmad%20Rifqi%20Asrib.pdf

[2] Bland C, Tonello L, Biganzoli E, Snowdon D, Antuono P, Lanza M. 2020. *Advances in Artificial Neural Networks*. Scientific Research.

[3] Budiarti D, Wilandari Y, Suparti. 2014. Analisis hubungan antara lama studi, jalur masuk dan indeks prestasi kumulatif (IPK) menggunakan model log linier. *J Gaussian*. [diakses 2022 Mei 20]. 3(1):41–50. https://ejournal3.undip.ac.id/index.php/gaussian/article/viewFile/4774/4606

[4] Chow T. 2007. *Neural Network And Computing Learnong Algorithms And Applications*. London: Imperial college press.
[5] Dozat T. 2016. Incorporating Nesterov Momentum into Adam. Di dalam: *ICLR Workshop*. hlm 2013–2016. [diakses 2022 Juli 7]. https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ
[6] Eshragh F, Pooyandeh M, Marceau DJ. 2015. Automated negotiation in environmental resource management: Review and assessment. *J Environ Manage*. 162:148–157.
[7] Farida I, Hendric SWHL. 2019. Prediksi Pola Kelulusan Mahasiswa Menggunakan Teknik Data Mining Classification Emerging Pattern. *Petir*. [diakses 2022 Mei 20]. 12(1):1-17. https://doi.org/10.33322/petir.v12i1.414
[8] Fausett L V. 2018. *Foundations of neural network, Architectures, Algorithms, and Applications*. New York: John Wiley & Sons.
[9] Guo H, Nguyen H, Vu DA, Bui XN. 2021. Forecasting mining capital cost for open-pit mining projects based on artificial neural network approach. *Resour Policy*. 74: 101474.
[10] Ibrahim Z, Rusli D. 2007. Predicting Students' Academic Performance: Comparing Artificial Neural Network, Decision Tree and Linear Regression. Di dalam: *21st Annual SAS Malaysia Forum*. Kuala Lumpur. hlm:1-6. [diakses 2022 Mei 20]. https://www.researchgate.net/publication/228894873
[11] Ioffe S, Szegedy C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv Prepr*. 1502.03167v3. hlm. 1–11. [diakses 2022 Juli 7]. https://arxiv.org/pdf/1502.03167.pdf
[12] Lestari YD. 2017. Jaringan Syaraf Tiruan Untuk Prediksi Penjualan Jamur Menggunakan Algoritma Backropagation. *J ISD*. 2(1):2477–863. http://repository.widyamataram.ac.id/uploads/pdfs/88-176-1-PB.pdf
[13] Kingma DP, Ba JL. 2017. Adam: A method for stochastic optimization. *arXiv Prepr*. 1412.6980. hlm. 1–15. [diakses 2022 Juli 7]. https://arxiv.org/abs/1412.6980
[14] Mawarni N. 2020. Prediksi IPK dan Masa Studi Calon Mahasiswa Baru Menggunakan Jaringan Syaraf Tiruan Propagasi Balik [Skripsi]. Yogyakarta: Universitas Sanata Dharma.
[15] Molnar C. 2022. Global Model-Agnostic Methods. Di dalam: *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. Online. [diakses 2022 Juli 7]. https://christophm.github.io/interpretable-ml-book/global-methods.html
[16] Muspardi M, Yusmanila Y, Widya W. 2021. Pengembangan Instrumen Penilaian Keterampilan Umum Mahasiswa Berbasis Standar Nasional Pendidikan Tinggi (SN-DIKTI). *Edukatif J Ilmu Pendidik*. [diakses 2022 Mei 20]. 4(1):590–601. https://doi.org/10.31004/edukatif.v4i1.1897
[17] Nesterov Y. 1983. A method for unconstrained convex minimization problem with the rate of convergence O($1/k^2$). *Dokl AN USSR*. 269:543–547.
[18] Nugroho MF, Wibowo S. 2017. Fitur Seleksi Forward Selection Untuk Menetukan Atribut Yang Berpengaruh Pada Klasifikasi Kelulusan Mahasiswa Fakultas Ilmu Komputer UNAKI Semarang Menggunakan Algoritma Naive Bayes. *J Inform Upgris*. [diakses 2022 Mei 20]. 3(1):63–70. https://www.researchgate.net/publication/320301582
[19] Nurdiati S, Najib MK. 2022. Comparing Five Machine Learning-Based Regression Models for Predicting the Study Period of Mathematics Students at IPB University. *JTAM (Jurnal Teor dan Apl Mat)*. 6(2). Siap Terbit.
[20] Ramadhiat A, Tritoasmoro II, Wijayanto I. 2016. Analisis Penggunaan Algoritma Genetika untuk Meningkatkan Performansi dari Klasifikasi Genre Musik Berbasis Jaringan Syaraf Tiruan Back-propagation. *e-Proceeding Eng*. [diakses 2022 Mei 20]. 3(2):1527–1535. https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/302/282
[21] Ramadhona G, Setiawan BD, Bachtiar FA. 2018. Prediksi Produktivitas Padi Menggunakan Jaringan Syaraf Tiruan Backpropagation. *J Pengemb Teknol Inf dan Ilmu Komput*. [diakses 2022 Mei 20]. 2(12):6048–6057. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/3493/1378
[22] Rojas R. 1996. The Backpropagation Algorithm. Di dalam: *Neural Networks*. Berlin, Heidelberg: Springer. hlm:149–182. [diakses 2022 Mei 20]. https://doi.org/10.1007/978-3-642-61068-4_7
[23] Romadhon AS, Widyaningrum VT. 2015. Klasifikasi Mutu Jeruk Nipis Dengan Metode Learning Vector Quantization. *J Ilmiah Rekayasa*. [diakses 2022 Mei 20]. 8(2):121–128. https://journal.trunojoyo.ac.id/rekayasa/article/view/2065/1696
[24] Ruder S. 2016. An overview of gradient descent optimization algorithms. *arXiv*. [diakses 2022 Mei 20]. https://arxiv.org/pdf/1609.04747.pdf
[25] Simbolon IAR, Yatussa'ada F, Wanto A. 2019. Penerapan Algoritma Backpropagation dalam Memprediksi Persentase Penduduk Buta Huruf di Indonesia. *J Inform Upgris*. [diakses 2022 Mei 20]. 4(2):121:128. https://doi.org/10.26877/jiu.v4i2.2423