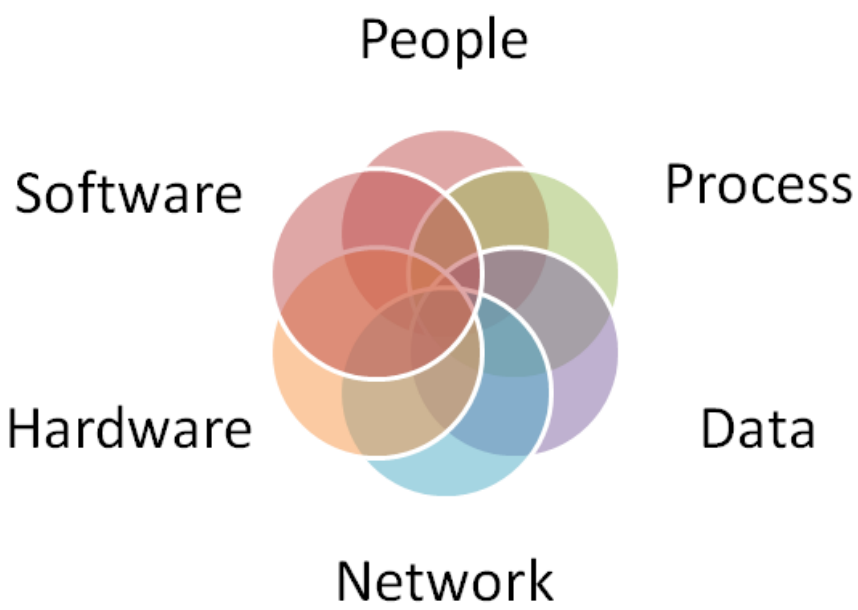
OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

ISSN 1979-3979
9 771979 397002

jurnal sisfo

Inspirasi Profesional Sistem Informasi

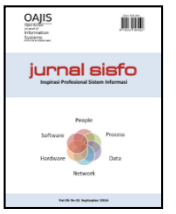


Vol. 11 No. 01 Mei 2024

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

jurnal sisfo

Jurnal Sisfo Vol. 11 No. 01 (2024)



Pimpinan Redaksi

Sholiq

Dewan Redaksi

Reny Nadlifatin

Mudjahidin

Tining Haryanti

Faizal Mahananto

Rizal Risnanda Hutama

Radityo Prasetianto Wibowo

Tata Pelaksana Usaha

Heppy Nuryanti

Sekretariat

Departemen Sistem Informasi – Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember (ITS) – Surabaya
Telp. 031-5999944 Fax. 031-5964965
Email: editor@jurnalsisfo.org
Website: http://jurnalsisfo.org

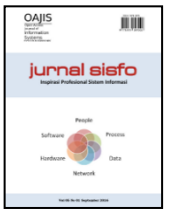
Jurnal SISFO juga dipublikasikan di *Open Access Journal of Information Systems* (OAJIS)

Website: http://is.its.ac.id/pubs/oajis/index.php

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

jurnal sisfo

Jurnal Sisfo Vol. 11 No. 01 (2024)



Mitra Bestari

Prof. Mahendrawathi ER., S.T., M.Sc, Ph.D. (Institut Teknologi Sepuluh Nopember)

Prof. Nur Aini Rakhmawati, S.Kom., M.Sc.Eng., Ph.D. (Institut Teknologi Sepuluh Nopember)

Dr. Muhammad Ainul Yaqin, S.Si., M.Kom. (Universitas Islam Negeri Maulana Malik Ibrahim)

Dr. Apol Pribadi Subriadi, S.T., M.T. (Institut Teknologi Sepuluh Nopember)

Dr. Bambang Setiawan, S.Kom., M.T. (Institut Teknologi Sepuluh Nopember)

Dr. Wiwik Anggraeni, S.Si., M.Kom. (Institut Teknologi Sepuluh Nopember)

Dr. Indra Waspada, S.T., M.T.I. (Universitas Diponegoro)

M. Amirul Haq, S.T., M.Sc., Ph.D. (Universitas Muhammadiyah Surabaya)

Ashr Hafiizh Tantri, S.Kom., M.Kom. (Universitas Muhammadiyah Surabaya)

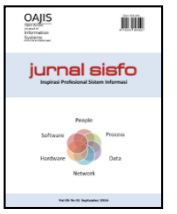
Doddy Ridwandono, S.Kom., M.Kom. (Universitas Pembangunan Nasional “Veteran Jawa Timur”)

Dhiani Tresna Absari, S.T., M.Kom. (Universitas Surabaya)

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

jurnal sisfo

Jurnal Sisfo Vol. 11 No. 01 (2024)



Daftar Isi

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

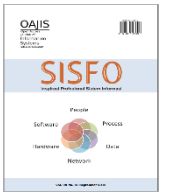
jurnal sisfo

Jurnal Sisfo Vol. 11 No. 1 (2024)

Halaman ini sengaja dikosongkan

OAJIS
Open Access Journal of Information Systems
is.its.ac.id/pubs/oajis/

SISFO
Inspirasi Profesional Sistem Informasi
Jurnal SISFO Vol.11 No.01 (2024) 25-38



Teori dan Penerapan Backpropagation Neural Networks untuk Internet of Things: Online dan Batch Mode

Anisa Dzulkarnain* , Mochamad Nizar Palefi Ma'ady

Program Studi Sistem Informasi, Fakultas Rekayasa Industri, Telkom University

Abstract

The field of artificial intelligence research is rapidly advancing and has been successfully implemented in various sectors such as healthcare, education, finance, security, business, and industry 4.0. Artificial intelligence essentially originates from the resurgence of research on artificial neural networks, namely deep learning. However, the rapid progress of this research topic, which has led to the emergence of numerous libraries and toolboxes, is not matched with a fundamental understanding of backpropagation neural networks learning. Yet, in applying deep learning, one is confronted with the challenge of setting parameters correctly so that deep learning can work effectively and efficiently, not through arbitrary parameter settings. Moreover, dependence on the use of libraries can result in dependency without understanding the basic concepts, which can hinder further research development, such as how to engineer the use of gradients in updating weights. Therefore, this article seeks to educate readers about the use of gradients in the backpropagation neural networks procedure through two learning approaches: yaitu online (stochastic gradient descent) dan batch mode (batch gradient descent). To give a better understanding, both approaches are applied to a case on the Internet of Things, and analysis of comparative patterns with error rate values using MatLab computation is demonstrated.

Keywords: Internet of Things, Backpropagation, Online learning, Batch learning, Deep Learning, Neural Networks

Abstrak

Riset kecerdasan artifisial kini kian berkembang pesat dan telah banyak berhasil diimplementasikan di berbagai sektor seperti kesehatan, pendidikan, keuangan, keamanan, bisnis hingga industri 4.0. Kecerdasan artifisial sejatinya memiliki esensi yang berasal dari lahirnya kembali penelitian *artificial neural networks*, yaitu *deep learning*. Akan tetapi, kemajuan yang pesat dari penelitian topik ini, yang mana hingga banyaknya bermunculan *library* dan *toolbox*, tidak diimbangi dengan pemahaman mendasar pembelajaran *backpropagation neural networks*. Padahal, dalam menerapkan *deep learning* akan dihadapkan pada bagaimana pengaturan parameter yang tepat agar *deep learning* dapat bekerja efektif dan efisien, yakni tidak dengan cara pengaturan parameter secara asal-asalan. Terlebih lagi, ketergantungan pada penggunaan *library* dapat mengakibatkan ketergantungan tanpa memahami konsep dasar yang akan menyulitkan pada pengembangan penelitian selanjutnya seperti bagaimana melakukan rekayasa penggunaan *gradient* dalam memperbarui bobot. Oleh karena itu, artikel ini berupaya untuk mengedukasi pembaca mengenai penggunaan *gradient* dalam prosedur *backpropagation neural networks* melalui dua pendekatan pembelajaran, yaitu *online* (*stochastic gradient descent*) dan *batch* (*gradient descent*) *mode*. Untuk pemahaman yang lebih baik, kedua pendekatan ini diaplikasikan pada kasus *Internet of Things* dan didemonstrasikan analisis perbandingan pola tingkat nilai *error* menggunakan komputasi MatLab.

Kata kunci: Internet of Things, Backpropagation, Online learning, Batch learning, Deep Learning, Neural Networks.

*Corresponding Author
Email address: anisadzulkarnain@telkomuniversity.ac.id (Anisa Dzulkarnain)
https://doi.org/10.24089/j.sisfo.2024.05.003

.

Histori Artikel: Disubmit 15-03-2024; Direvisi 25-04-2024; Diterima 29-04-2024; Tersedia online 31-05-2024

1. Pendahuluan

Penelitian di bidang kecerdasan artifisial (*Artificial Intelligence* / AI) semakin naik daun dan berkembang sangat pesat. Itu diawali dengan munculnya metode *deep learning* yang cikal bakalnya dari penemuan jaringan syaraf tiruan oleh Warren McCulloch pada tahun 1943, sedangkan *deep learning* sendiri ditemukan sekitar 40 tahun kemudian oleh Geoffrey Hinton [1]. Sejak saat itu, seiring berkembangnya juga penelitian di bidang komputasi komputer canggih (*super computing*), pemrosesan melalui komputer semakin cepat, maka pengembangan *deep learning* juga semakin berkembang. *Deep learning* berprinsip pada memperbaiki bobot dengan menggunakan *gradient* saat proses *backpropagation* sedemikian hingga dapat menghasilkan model yang tepat untuk suatu konteks input dan output tertentu. Hingga saat ini ada banyak varian dari *deep learning* seperti *long short term memory* (LSTM) [2], *graph convolutional network* (GCN) [3], *gated recurrent unit* (GRU) [4], *convolutional neural network* (CNN) [5], *graph neural network* (GNN) [6], atau *deep reinforcement learning* (RL)[7].

Perkembangan yang pesat ini menghadirkan ekosistem penggunaan *library* algoritma di bidang kecerdasan artificial yang 'siap pakai', misalkan Scikit-Learn [8], PyTorch [9], atau Keras [10]. Satu sisi, ini memudahkan pengguna mengimplementasikan algoritma. Akan tetapi, bagi pengembang AI justru menghadapi tantangan ketika menggunakan *library* [11]. Beberapa kesulitan bagi pengembang AI adalah seperti tidak mengetahui apakah bagus atau buruk praktiknya menggunakan suatu *library*, tidak memahami alur prosedur dari suatu algoritma *library*, sulit mengenali research gaps antar algoritma, dll [11][software2-0]. Terlebih lagi bila memadukan seluruh library dalam suatu *pipeline* penerapan algoritma yang mana juga membutuhkan library lain pendukung seperti Numpy [12], SciPy [13], Pandas [14], Matplotlib [15], dll [16]. Suatu *pipeline* pengembangan algoritma meliputi beberapa aspek yaitu pengumpulan data, pembersihan data, pemilihan fitur, pelatihan model, dan evaluasi model, yang mana itu masing-masing akan membutuhkan *library* berbeda-beda kegunaannya [16].

Oleh karena itu, artikel ini menyoroti pentingnya memahami fundamental dari pengembangan AI berbasis *neural networks*, yang mana semua yang disebutkan di atas secara umum sama-sama menggunakan *neural networks* yang menggunakan konsep *backpropagation* dan pengaturan parameter *learning rate* dalam praktiknya [17]. Artikel ini bertujuan untuk mengangkat kembali pentingnya memahami konsep *backpropagation* sebelum menerapkan *library* yang tersedia secara langsung, agar pengembang AI dapat mempertimbangkan bagaimana kustomisasi dari *library* tersebut terhadap aplikasi berbasis AI yang dikembangkannya. Pada dasarnya, berdasarkan literatur, teknik memperbaiki model atau memperbarui bobot jaringan memiliki dua cara, yaitu pembelajaran *on-line mode* dan *batch mode* [18]. *Batch mode backpropagation* menghimpun seluruh nilai *gradient*, lalu melakukan rata-rata terlebih dahulu sebelum memperbarui bobot. Tidak seperti *batch mode*, *on-line backpropagation* memperbarui bobot setelah melalui setiap data [19]. Sedangkan parameter yang mempengaruhi juga ada beberapa macam seperti *learning rate* [20], momentum [20], *batch size* [21], *weight decay (regularization)* [22], *proportional factor* [23], dan pemilihan *activation function*, yang mana itu semua perlu dipahami sebelum menggunakan *library*.

Untuk menunjang hal di atas, artikel ini menerapkan dan mengedukasi kepada pembaca tentang bagaimana peran *backpropagation neural networks* dapat memperbarui bobot secara iteratif terhadap konteks permasalahan di lingkungan *internet of things* (IoT). Ini juga telah diterapkan dalam bentuk *deep learning* oleh banyak penelitian yang lain namun menggunakan *library* [24], [25], [26]. Agar terstruktur, artikel ini di awali dengan pembahasan literatur terkait di Bab 2, lalu Bab 3 menjelaskan metodologi dari tahapan alur penelitian. Bab 4 membahas mengenai hasil dan disimpulkan di Bab 5.

2. Tinjauan Pustaka

2.1 BPNN Sebagai Pondasi Metode Deep Learning

Machine learning pada algoritma jaringan syaraf tiruan (*artificial neural networks*) mengadopsi cara kerja otak manusia dalam mempelajari suatu hal, yaitu bagaimana sinapsis dapat termodifikasi sesuai stimulan dari lingkungan [27], [28]. Pada konteks pembelajaran, proses pembelajaran terdapat pada kemampuan memperbarui bobot perubahan *gradient* di dalam arsitektur jaringan syaraf tiruan. Jaringan syaraf tiruan dapat dikategorikan sebagai operasi matematika dasar atau algoritma untuk berbagai macam kepentingan [29]. Jaringan saraf tiruan memiliki simpul neuron yang saling berhubungan, mirip dengan web, di otak manusia [30]. *Neural Networks* terdiri dari banyak *node* (neuron) yang tersebar di berbagai lapisan (*layer*), dengan setiap layer dapat memiliki banyak *node*. Setiap *node* pada arsitektur ANN memiliki bobot tertentu [31]. Jaringan syaraf tiruan juga disebut model *black-box* karena terdapat *hidden layer* [32]. ANN memiliki banyak aplikasi, termasuk *image and speech recognition*, *natural language processing*, diagnosis medis, hingga *forecasting* di bidang keuangan [33].

Algoritma yang paling sering digunakan untuk melatih jaringan saraf tiruan pada *deep learning* adalah *backpropagation* [28]. Karena sifat pembaruan parameter yang nonlokal—perbaruan parameter di satu lapisan bergantung pada aktivitas di lapisan yang lebih dalam—implementasi langsung *backpropagation* di otak sering dianggap tidak masuk akal secara biologis [34]. Salah satu keunggulan algoritma *backpropagation* adalah kemampuan untuk menghubungkan proses input dan output melalui *feed forward* dan *backward* [35]. Keberhasilan *deep learning* bergantung pada algoritma *backpropagation* yang digunakan untuk melatih *deep neural network* [36].

2.2 Penerapan BPNN pada Studi Kasus Internet of Things

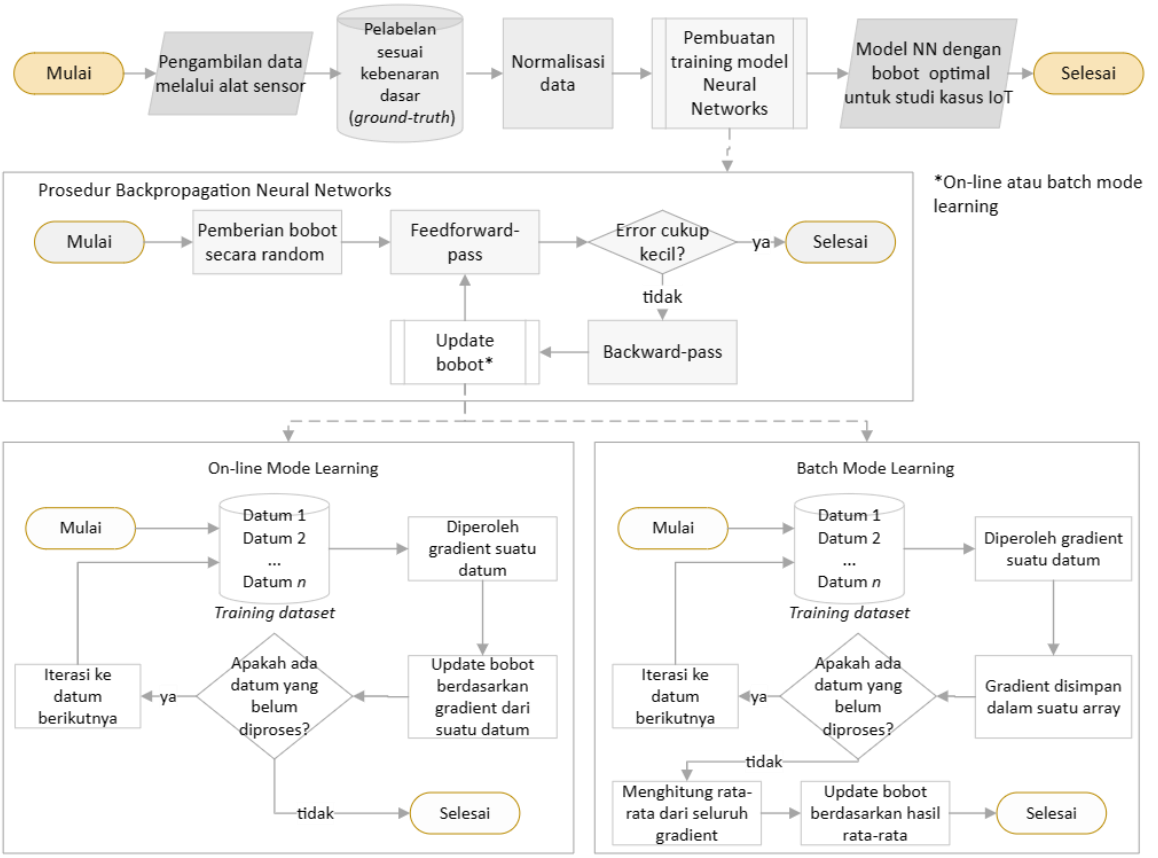
Pengaplikasian *internet of things* (IoT) telah banyak dilakukan di berbagai kasus [37], [38], [39]. Beberapa diantaranya juga dilakukan penerapan *machine learning* pada *internet of things* [40], [41]. *Internet of Things* (IoT) dan *Artificial Neural Networks* (ANN) adalah dua teknologi besar yang semakin terintegrasi dalam sistem pemantauan dan diagnostik [42]. Fitur utama *Internet of Things* adalah memungkinkan orang terbebas dari batasan karena mereka harus berada di lokasi perangkat [43]. Di masa depan, *Internet of Things* (IoT) akan meningkatkan efisiensi dan mengendalikan operasi sehari-hari, industri, pertanian, dan proses lainnya. Banyak komponen penting yang berkurang secara signifikan, termasuk berkurangnya waktu yang hilang karena kerusakan sistem atau mesin yang tidak terduga, berkurangnya konsumsi energi, dan berkurangnya waktu perjalanan [43]. Penelitian terkait penerapan BPNN dimanfaatkan untuk memprediksi terjadinya kemacetan dan pola evolusinya yang dilakukan oleh Xiaolei et al [44]. Penelitian tersebut menggabungkan *Intelligent Transportation System* dan *internet of things*. Hasi penelitian menunjukkan akurasi model yang dikembangkan mencapai 88%. BPNN juga diterapkan untuk mengembangkan model pengujian pemantauan daya [45]. Akurasi dari model tersebut sebesar 97.5% menunjukkan BPNN merupakan metode yang cocok untuk diterapkan pada peralatan *internet of things.*

2.3 On-line dan Batch mode BPNN dalam Memperbarui Bobot

Terdapat dua cara memperbarui bobot pada *backpropagation neural networks* (BPNN), yaitu *on-line mode* dan *batch mode* [46]. *On-line mode* memungkinkan perbaruan bobot setelah melewati seluruh data. Kelebihan dari *on-line mode* yaitu pada proses training meminimalisir *error* ouput jaringan [47]. Selain itu, Kemampuan algoritma untuk on-line mode membutuhkan perubahan parameter [48]. Pada pembelajaran batch mode, perbaruan bobot dilakukan setelah menyelesaikan seluruh *epoch* [49]. Pada *batch-mode* dilakukan akumulasi perubahan bobot dan kemudian diterapkan perubahan tersebut untuk memperbarui bobot setelah melewati seluruh *epoch* [49].

3. Metodologi

Berikut tahapan alur dari penelitian ini seperti ditunjukkan pada Gambar 1.



Gambar 1. Tahapan *backpropagation neural networks mode: online* dan *batch*.

Pada penelitian ini, obyek penerapan metode adalah pada studi kasus *Internet of Things*, yang mana data yang diperoleh bersumber dari alat sensor menangkap data pada sistem pengawasan kualitas air tambak. Karena itu, tahapan ini melalui proses normalisasi agar data siap diproses di model *neural networks*, kemudian data melalui proses *backpropagation* dengan beberapa iterasi hingga selisih antara keluaran (y_1^3) dan target (t) dianggap cukup sedikit atau *error* kecil. Lebih detail dijelaskan pada sub-bab berikut.

3.1 Dataset dan Normalisasi Data

Pengumpulan data dilakukan pada sistem monitoring kualitas air tambak untuk hewan ternak lobster air tawar. Lobster air tawar terkenal sensitif terhadap perubahan kadar air, artinya bila salah satu indikator bernilai terlalu tinggi atau rendah, maka lobster air tawar dapat stress atau bahkan mati. Indikator kualitas air tambak yang dapat dideteksi oleh alat sensor umumnya terdiri dari TDS, suhu, dan pH. Biaya pembelian ketiga alat sensor ini juga dinilai masih terjangkau (*reasonable*). Label atau kelas mengindikasikan 1 untuk kualitas air tambak yang baik, dan angka 0 menunjukkan tambak memiliki kualitas yang buruk, yang mana diketahuinya kelas ini berdasarkan kenyataan di lapangan atau menurut pakar (lihat Tabel 1).

Tabel 1. *Dataset* hasil pengambilan data dari alat sensor IoT.

Data	TDS	Suhu	pH	Kelas
1	276	25.8	7.7	1
2	327	22	8	0
3	282	30	7.5	1

$x\ ternormalisasi = \frac{(x - x\ minimum)}{rentang\ x}$, dimana $rentang\ x = x\ maximum - x\ minimum.$ (1)

Pada Tabel 1, dapat dilihat data sebelum dilakukan normalisasi memiliki. apabila data belum dilakukan normalisasi, data antar variabel variansinya tinggi. Dalam ilmu statistika, data dikenal menjadi 2 macam yaitu data homogen dan data heterogen. Apabila data antar variabel memiliki variansi yang tinggi, dikategorikan sebagai data heterogen. Sedangkan jika data antar variabel memiliki variansi yang rendah, dikategorikan sebagai data homogen. Dalam analisis statistika diperlukan data yang homogen. Oleh karena itu diperlukan proses normalisasi untuk mengurangi variansi data antar variabel. Misalkan pada indikator atau fitur TDS, maka rentang x = 327 – 276 = 51. Kemudian, nilai normalisasi dapat diketahui untuk setiap datum, yaitu misalkan datum ke-2: $x_2\ ternormalisasi = \frac{(327-276)}{51} = 1$. Itu dilakukan untuk semua data, maka diperolehlah seperti pada Tabel 2. Hasil normalisasi ini sebenarnya akan jauh lebih memudahkan algoritma dalam pemrosesan dibandingkan dengan data asli, yaitu berdampak pada komputasi komputer. Selain itu, data asli bila langsung diproses juga tetap perlu perlakuan agar nilai kelasnya dapat sesuai

Tabel 2. *Dataset* setelah proses normalisasi.

Data	TDS	Suhu	pH	Kelas
1	0	0.5	0.4	1
2	1	0	1	0
3	0.1	1	0	1

3.2 Prosedur Backpropagation Neural Networks

Prosedur *Backpropagation* merupakan suatu proses yang terdiri dari *feedforward- dan backward-pass*. Proses ini di dalamnya melibatkan penggunaan fungsi *non-linearity* atau disebut juga dengan fungsi aktivasi. oleh karena itu, *neural networks* disebut juga metode nonlinier. Fungsi aktivasi dapat berupa Sigmoid, ReLU, Tanh, dll. Berikut penjelasan mengenai *feedforward-pass* dan *backward-pass* beserta komputasi matematis dari sampel data Tabel 2 di atas. Untuk konteks permasalahan IoT di atas, fungsi aktivasi yang digunakan adalah fungsi aktivasi Sigmoid. Tabel 3 menunjukkan bobot sebagai contoh perhitungan matematis pada artikel ini.

Tabel 3. Nilai bobot yang digunakan sebagai sampel perhitungan.

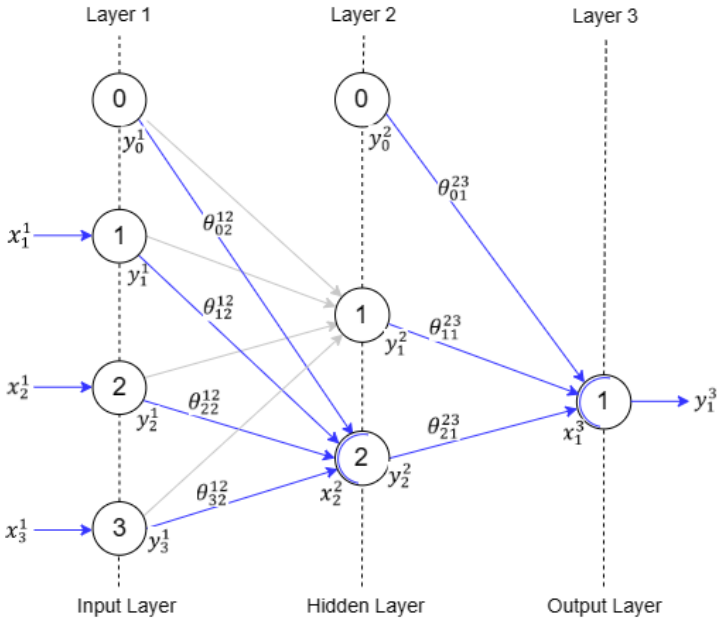
Input Layer ke *Hidden Layer*	Bobot	*Hidden Layer* ke *Output Layer*	Bobot
θ_{02}^{12}	1	θ_{01}^{23}	0.2
θ_{12}^{12}	0.3	θ_{11}^{23}	0.1
θ_{22}^{12}	0.4	θ_{21}^{23}	0.1
θ_{32}^{12}	0.5		

Tabel 4. Data ke-1 sebagai input sampel perhitungan dengan kelas bernilai 1.

Indikator	Input	Nilai
TDS	x_1^1	0
Suhu	x_2^1	0.5
pH	x_3^1	0.4

Untuk mendukung pemahaman yang komprehensif, Tabel 4 menunjukkan input dari data ke-1 yang akan didemonstrasikan perhitungannya pada sub-bab berikut ini.

3.2.1 *Feedforward-pass*



Gambar 2. Arsitektur *neural networks* 3-2-1 untuk proses *feedforward-pass*.
Perhitungan matematis ditunjukkan untuk jaringan yang bergaris biru.

Untuk memudahkan pemahaman teori, berikut ini demonstrasi perhitungan sekali iterasi (*epoch*) dari *feedforward-pass* dengan tiga input dan satu output pada arsitektur NN 3-2-1. Agar iterasi lebih sederhana, proses berfokus pada garis yang diindikasikan dengan warna biru (lihat Gambar 2).

$y_0^1 = 1;\ \ y_1^1 = x_1^1;\ \ y_2^1 = x_2^1;\ \ y_3^1 = x_3^1$

$x_2^2 = y_0^1\theta_{02}^{12} + y_1^1\theta_{12}^{12} + y_2^1\theta_{22}^{12} + y_3^1\theta_{32}^{12} = 1$

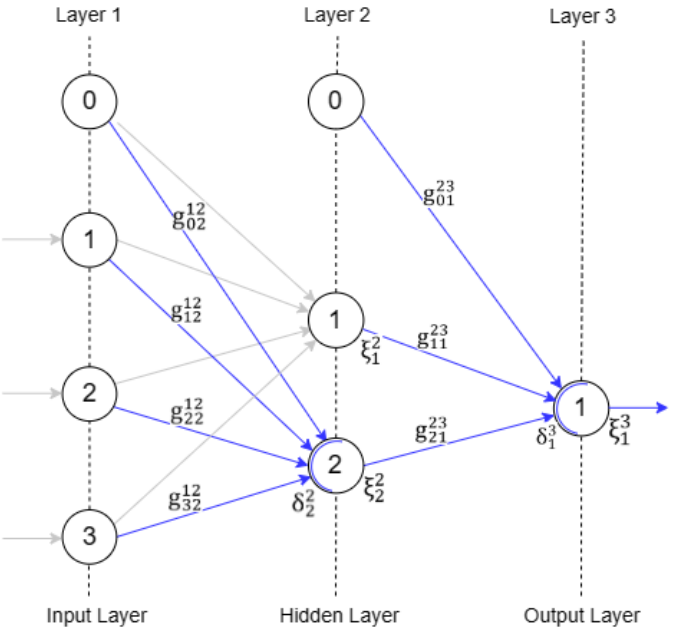
$y_0^2 = 1;\ \ y_1^2 = 1;\ \ y_2^2 = 1$ (*menggunakan fungsi aktivasi Sigmoid*)

$x_1^3 = y_0^2\theta_{01}^{23} + y_1^2\theta_{11}^{23} + y_2^2\theta_{21}^{23} = 0.5$

$y_1^3 = 0.5$

Setelah diperoleh y_1^3, berikutnya dicek tingkat *error rate* yakni dibandingkan dengan target (*t*). Sehingga, *error rate* = $|y_1^3 - t| = 0.5$. Karena tingkat *error* masih dianggap besar, maka perlu perbaikan bobot yang terus dilakukan (iterasi) atau disebut juga dengan mencari model algoritma yang tepat.

3.2.2 Backward-pass



Gambar 3. Arsitektur *neural networks* 3-2-1 untuk proses *backward-pass*.
Perhitungan matematis ditunjukkan untuk jaringan yang bergaris biru.

Karena masih dianggap memiliki *error* cukup besar, maka *backward-pass* dilakukan. Berikut ini demonstrasi numerik dari perhitungan *backward-pass* sekali iterasi. Dalam hal ini juga berfokus pada garis berwarna biru pada Gambar 3. Hasil dari keseluruhan perhitungan *backward-pass* terdapat pada Tabel 5.

$$\xi_1^3(error) = |y_1^3 - t| = 0.5$$

$$\delta_1^3 = [y_1^3(1 - y_1^3)]\xi_1^3$$

$$= [0.5(1 - 0.5^{\square})]\, 0.5$$

$$= 0.125$$

$$g_{01}^{23} = y_0^2\ \delta_1^3 = 0.125$$

$$\xi_2^2 = \theta_{21old}^{23}\ \delta_1^3 = 0.025$$

$$\delta_2^2 = [y_2^2\ (1 - y_2^2\ )]\ \xi_2^2 = 0$$

Tabel 5.Nilai *gradient* hasil dari *backpropagation* NN.

Input Layer ke *Hidden Layer*	*Gradient*	*Hidden Layer* ke *Output Layer*	*Gradient*
g_{02}^{12}	0	g_{01}^{23}	0.125
g_{12}^{12}	0	g_{11}^{23}	0.125
g_{22}^{12}	0	g_{21}^{23}	0.125
g_{32}^{12}	0		

3.3 Update Bobot via Gradient

Ada dua metode pembaruan bobot, yaitu melalui *on-line* atau *batch mode*. Berikut ini perbedaan matematis antara *on-line* dan *batch mode learning* dengan memanfaatkan *gradient* yang diperoleh dari proses *backward-pass*.

3.3.1 On-line learning (*stochastic gradient descent*)

Di *on-line learning*, *gradient* diproses pada setiap datum dari *dataset*. Karena itu diperoleh nilai *loss function* (*J*) untuk setiap data. Sebagai contoh memperbarui bobot θ_{12}^{12} pada *gradient* g_{12}^{12}.

$\nabla_\theta J(\theta; x_i, y_i)$, dimana *i* adalah tiap datum dari *dataset* untuk nilai *x* dan *y*.

$\theta_{new} \coloneqq \theta_{old} - \alpha \nabla_\theta J(\theta; x_i, y_i)$, dimana α merupakan *learning rate* yang dapat berupa *proportional term*, *momentum*, dll.

Misalkan, iterasi pertama dari hasil *backward-pass* di atas, lalu melakukan update bobot seperti berikut ini. Pada konteks ini asumsinya tanpa menggunakan nilai *learning rate* α.

$\nabla_\theta J(\theta; x_i, y_i) = \mathrm{g}_{12}^{12} = 0.$ (untuk data pertama dalam sekali epoch).

$\theta_{new} \coloneqq \theta_{old} - \alpha \nabla_\theta J(\theta; x_i, y_i)$

$\theta_{21}^{23} = \theta_{21\ old}^{23} - g_{21}^{23} = 0.2 - 0.125 = 0.075$

3.3.2 Batch-mode learning (*batch gradient descent*)

Berbeda dengan *on-line learning*, *batch-mode learning* menghimpun seluruh *gradient* terlebih dahulu, lalu melakukan rata-rata. Maka nilai itulah yang akan digunakan untuk memperbarui bobot.

$\nabla_\theta J(\theta; X, Y) = \frac{1}{|X|} \sum_{i=1}^{|X|} \nabla_\theta J(\theta; x_i, y_i)$, dimana *i* adalah tiap datum dari *dataset* untuk nilai *x* dan *y*.

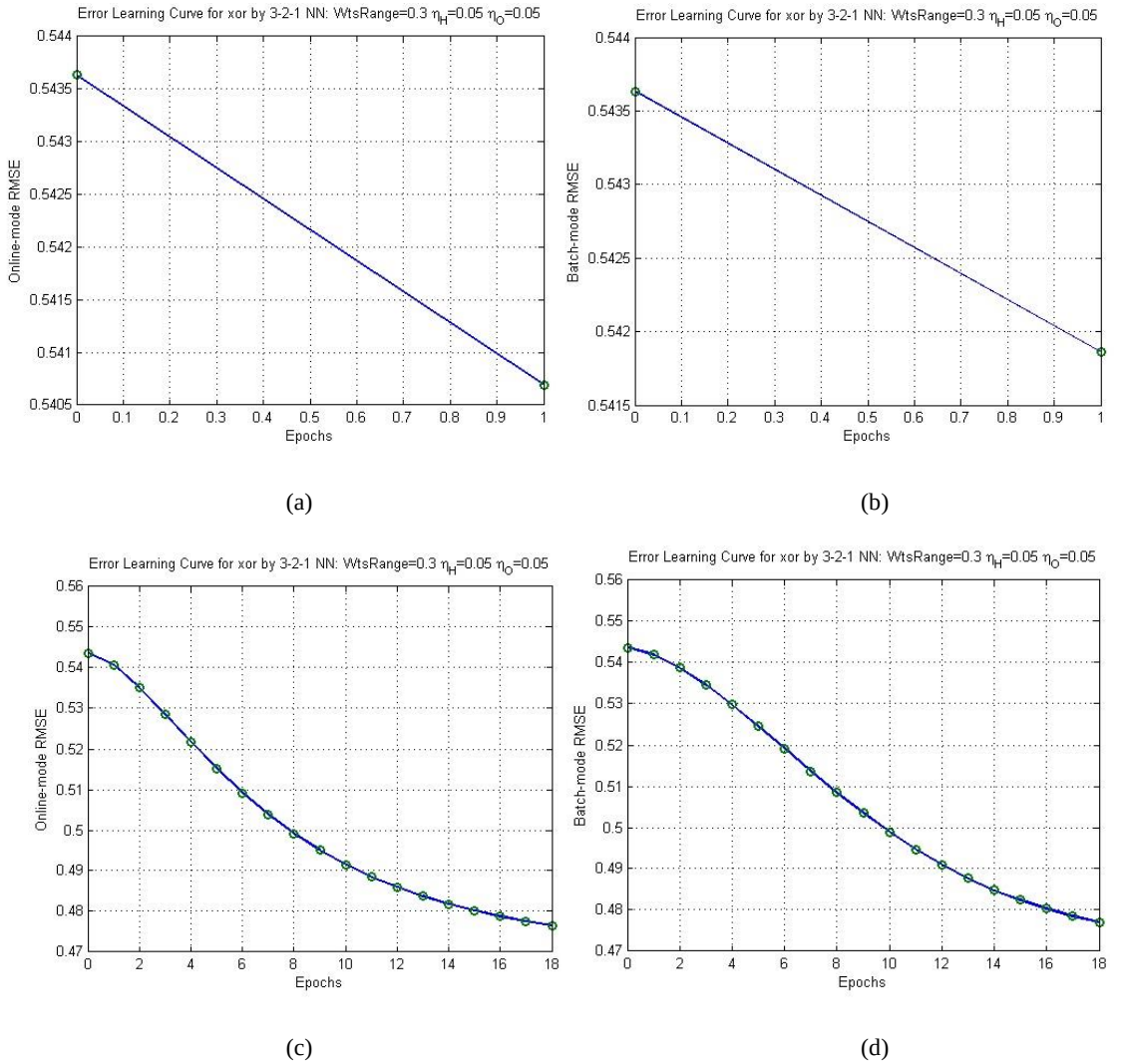
$\theta_{new} \coloneqq \theta_{old} - \alpha \nabla_\theta J(\theta; X, Y)$, dimana α merupakan *learning rate* dapat berupa *proportional term*, *momentum*, dll.

$\nabla_\theta J(\theta; X, Y) = \mathrm{g}_{12}^{12} = 0.$ (rata-rata *gradient* dari seluruh data dalam sekali *epoch*).

$\theta_{new} \coloneqq \theta_{old} - \alpha \nabla_\theta J(\theta; X, Y)$

4. Hasil dan Pembahasan

Dalam penelitian ini, kasus yang diangkat adalah melakukan klasifikasi perhitungan algoritma *neural networks* beserta perbaikan bobotnya melalui *on-line* dan *batch backpropagation* menggunakan *gradient* untuk studi kasus *Internet of Things*. Pada bab sebelumnya telah dijelaskan bagaimana teori prosedur dari tahapan metode penelitian namun masih sekedar data sampel. Bab ini menunjukkan hasil penerapan dengan *dataset* yang sama menggunakan MatLab. Perbedaan antara *online* dan *batch mode* lebih terlihat jelas ditunjukkan dalam bentuk gambar grafik hasil dari MatLab. Dengan komposisi parameter yang sama, yaitu learning rate, dan momentum, juga weight random yang sama. Data dari Tabel 2 diujikan dalam dua cara, yaitu sekali iterasi dan iterasi hingga dianggap cukup terlihat perbandingan efisiensinya, yaitu sebanyak 18 kali iterasi.



Gambar 4. Grafik RMSE *online* dan *batch backpropagation* menggunakan MatLab.

Gambar 4 (a) dan (c) menunjukkan hasil dari *on-line mode*. Terlihat (a) yakni pada sekali *epoch* memiliki nilai RMSE yang lebih rendah dari *batch-mode* seperti Gambar 4 (b). Sehingga, penurunan *online-mode* relatif lebih signifikan hanya dengan 18x *epoch* (lihat Gambar 4 (c) dibanding Gambar 4 (d)). *Setting* keduanya sama termasuk bobot dan outputnya. Namun, pada *epoch* ke-18 memiliki bobot dan *output* yang berbeda. Untuk lebih detailnya dapat melihat pada Tabel 6 di bawah ini.

Tabel 6. Perbandingan hasil *online* dan *batch mode learning*

Neural Networks Setup		
BPNN Mode	*On-line mode*	*Batch mode*
Learning rate parameter:		
- *Eta at output layer*	0.05	0.05
- *Eta at hidden layer*	0.05	0.05
- Momentum	0.8	0.8
Architecture of NN:	3-2-1	3-2-1
Initial RMSE:	0.543633	0.543633
Random weight SEED:	11	11
Initial weight values:		
- θ^{12}	0.2126, 0.1622, -0.1346, -0.2727, 0.1810, 0.02154	0.2126, 0.1622, -0.1346, -0.2727, 0.1810, 0.02154
- θ^{12}_{bias}	0.0852, -0.2608	0.0852, -0.2608
- θ^{23}	-0.0781, -0.2318	-0.0781, -0.2318
- θ^{23}_{bias}	-0.2629	-0.2629
Output values per data:		
- data 1	0.397671	0.397671
- data 2	0.400651	0.400651
- data 3	0.397266	0.397266
Neural Networks Learning Termination		
BPNN Mode	*On-line mode*	*Batch mode*
Number of epoch:	18	18
RMSE:	0.476535	0.477027
Final weight values:		
- θ^{12}	0.2031, 0.1733, -0.1421 -0.2634, 0.1610, 0.2203	0.2091, 0.1654, -0.1374 -0.2617, 0.1565, 0.2214
- θ^{12}_{bias}	0.0895, -0.2773	0.0858, -0.2809
- θ^{23}	0.2061, 0.0364	0.2027, 0.0296
- θ^{23}_{bias}	0.2503	0.2458
Output values per data:		
- data 1	0.593065	0.590699
- data 2	0.592992	0.590869
- data 3	0.595004	0.592556

Berdasarkan Tabel 6, kedua percobaan, *online*-dan-*batch mode*, memiliki pengaturan *neural networks setup* yang sama meliputi learning rate menggunakan eta dan momentum, arsitektur yang digunakan, nilai bobot seluruhnya dan percobaan menggunakan *random seed* yang sama. Hasilnya, cukup dengan jumlah iterasi sebanyak 18 kali, terlihat perbedaan penurunan tingkat *error* (RMSE) yang cukup signifikan untuk *online* dan *batch mode*, yaitu 0.476535 dan 0.477027, secara berurutan. Tentu penurunan *error* ini semakin terlihat apabila jumlah *epoch* diperbanyak seperti 100 atau 1.000 kali iterasi. Akan tetapi, sebagai studi kasus sampel, maka penelitian ini menggunakan data sampel dengan jumlah yang relatif sedikit yang mana di bawah 50% *error rate* dianggap cukup kecil dan iterasi dapat dihentikan.

Beda pendekatan pembaruan nilai bobot ini menunjukkan pentingnya teknik penanganan *gradient* dengan tepat terlebih lagi bagi fokusnya pengembangan *deep learning*, secara juga umumnya juga terdapat *backpropagation*. Selain itu, pada Tabel 6 tersebut juga terlihat perbedaan nilai bobot dan data keluaran setelah melalui pembaruan *gradient* antara *online* dan *batch*. Itu dikarenakan *gradient* dari kedua pendekatan berbeda di suatu iterasi yang sama. Sehingga, nilai pada bobot yang dihasilkan oleh *online mode* merupakan nilai yang data keluarannya lebih mendekati kebenaran dasar (*ground truth*) daripada nilai bobot dari *batch mode*. Temuan lain dari hasil pemrosesan *backpropagation* juga adalah nilai bobot dapat juga berupa nilai negatif, yang mana tidak masalah bagi algoritma dan nilai negatif untuk menuju akurasi yang lebih tinggi.

5. Kesimpulan

Penelitian ini menghasilkan kesimpulan dan saran untuk pengembangan penelitian selanjutnya seperti pada sub-bab berikut.

5.1 Simpulan

Di dunia riset *artificial intelligence*, topik *deep learning* telah dibahas secara sangat meluas dengan berbagai varian pengembangannya. Namun, dasar dari *deep learning* kurang mendapat perhatian oleh pengembang AI. Itu disebabkan banyaknya *library* algoritma yang tersedia yang memudahkan pengaplikasian tanpa harus memahami prosedur dari algoritma tersebut. Sedangkan, konsep *backpropagation* merupakan pondasi dari *deep learning* karena di dalamnya teredapat cara pembaruan bobot model training untuk mencapai akurasi yang lebih baik. Oleh karena pentinya hal tersebut, penelitian ini membahas bagaimana cara memperbarui bobot melalui *gradient* melalui dua cara, yaitu *on-line* dan *batch mode*.

Penerapan *backpropagation neural networks* terhadap studi kasus *Internet of Things* melalui proses normalisasi data terlebih dahulu karena data yang diperoleh dari alat sensor yang berbeda-beda yang mana itu memiliki rentang nilai berbeda, yaitu ada tiga indikator sebagai masukan: TDS, Suhu, dan pH. Sedangkan, keluaran dari kasus monitoring kualitas air tambak ini ada satu, yaitu kelas kualitas tambak air dengan nilai (1) untuk layak dan (0) untuk tidak layak. Hasilnya diperoleh bahwa *on-line learning* lebih baik daripada *batch-mode* dengan pembuktian melalui komputasi MatLab yakni nilai RMSE 0.476535 dan 0.477027, secara berurutan untuk *epoch* ke-18. Sehingga, apabila dilakukan *epoch* yang lebih banyak akan menghasilkan perbedaan yang lebih signifikan.

Penelitian ini diharapkan dapat mengedukasi pembaca dalam menerapkan *deep learning* maupun varian pengembangan lainnya dengan pengaturan kombinasi antara parameter *learning rate*, arsitektur *neural networks*, dan perlakuan terhadap *gradient* dengan lebih tepat demi mencapai akurasi yang lebih tinggi. Penelitian ini dianggap penting dengan contoh sampel penerapan sederhana terlebih bila menghadapi persoalan yang lebih kompleks.

5.2 Saran

Pada penelitian ini diajukan dua mode pembaruan bobot, yaitu *on-line dan batch*. Saran untuk penelitian selanjutnya adalah dengan membandingkan metode pembaruan bobot lainnya seperti *adaptive learning rate*, *hessian-free*, *conjugate gradient*, dan *levenberg-marquardt algorithm*.

6. Daftar Rujukan

[1] LeCun, Yann, Y. Bengio, and G. Hinton, "Deep Learning," *nature*, pp. 436–444, 2015.

[2] D. B, Z. Q, G. J, and W. L, "Deep learning with long short-term memory neural networks combining wavelet transform and principal component analysis for daily urban water demand forecasting.," *Expert Syst Appl*.

[3] Kipf, T. N, and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[4] Dey, Rahul, and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in *IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, IEEE, 2017.
[5] Li, Zewen, and et al., "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, pp. 6999–7019, 2022.
[6] Scarselli, Franco, and et al., "The graph neural network model," in *IEEE transactions on neural networks*, 2008, pp. 61–80.
[7] Francois-Lavet, Vincent, and et. al, "An introduction to deep reinforcement learning.," *Foundations and Trends® in Machine Learning*, vol. 11, no. 3–4, pp. 219–354, 2018.
[8] "Scikit-Learn." Accessed: Aug. 01, 2021. [Online]. Available: https://scikit-learn.org/stable/
[9] "Pytorch." Accessed: Aug. 01, 2021. [Online]. Available: https://pytorch.org/
[10] N. Ketkar, "Introduction to keras," *Deep learning with python: a hands-on introduction*, pp. 97–111, 2017.
[11] M. Dilhara, A. Ketkar, and D. Dig, "Understanding Software-2.0," *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 4, pp. 1–42, Oct. 2021, doi: 10.1145/3453478.
[12] "Numpy." Accessed: Mar. 13, 2024. [Online]. Available: https://numpy.org/
[13] Virtanen, Pauli, and et al., "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature method*, pp. 261–272, 2020.
[14] "Pandas." Accessed: Mar. 13, 2024. [Online]. Available: https://pandas.pydata.org/
[15] "Matplotlib." Accessed: Aug. 01, 2021. [Online]. Available: https://matplotlib.org/
[16] Y. Wang *et al.*, "Can Machine Learning Pipelines Be Better Configured?," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, New York, NY, USA: ACM, Nov. 2023, pp. 463–475. doi: 10.1145/3611643.3616352.
[17] Kosson, Atli, and et al., "Pipelined backpropagation at scale: training large models without batches," in *Proceedings of Machine Learning and Systems 3*, 2021, pp. 479–501.
[18] Heskes, Tom, and W. Wiegerinck, "A theoretical comparison of batch-mode, on-line, cyclic, and almost-cyclic learning," in *IEEE transactions on neural networks 7*, 1996, pp. 919–925.
[19] Gori, Marco, and marco Maggini, "Optimal convergence of on-line backpropagation," in *IEEE transactions on neural networks 7*, 1996, pp. 251–254.
[20] Kamiyama, Naoki, and et al., "Tuning of learning rate and momentum on backpropagation," in *ICCS/ISITA92*, Singapore: IEEE, 1992.
[21] Kalayeh, M. M, and M. Shah, "Training faster by separating modes of variation in batch-normalized models," in *IEEE transactions on pattern analysis and machine intelligence 42*, 2019, pp. 1483–1500.
[22] Xie, Zeke, I. Sato, and M. Sugiyama, "Stable weight decay regularization," 2020.
[23] Zweiri, Y. H, J. F. Whidborne, and L. D. Seneviratne, "A three-term backpropagation algorithm," in *Neurocomputing 50*, 2003, pp. 305–318.
[24] K. Zhang *et al.*, "Compacting Deep Neural Networks for Internet of Things: Methods and Applications," *IEEE Internet Things J*, vol. 8, no. 15, pp. 11935–11959, Aug. 2021, doi: 10.1109/JIOT.2021.3063497.
[25] M. Almiani, A. AbuGhazleh, Y. Jararweh, and A. Razaque, "DDoS detection in 5G-enabled IoT networks using deep Kalman backpropagation neural network," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 11, pp. 3337–3349, Nov. 2021, doi: 10.1007/s13042-021-01323-7.
[26] T. J. Saleem and M. A. Chishti, "Deep Learning for Internet of Things Data Analytics," *Procedia Comput Sci*, vol. 163, pp. 381–390, 2019, doi: 10.1016/j.procs.2019.12.120.
[27] G. I. Marthasari and A. Djunaidy, "Optimasi Data Latih Menggunakan Algoritma Genetika Untuk Peramalan Harga Emas Berbasis Generalized Regression Neural Network," *Sisfo*, vol. 05, no. 01, pp. 62–69, Mar. 2014, doi: 10.24089/j.sisfo.2014.03.014.
[28] T. P. Lillicrap, A. Santoro, L. Marris, C. Akerman, and G. Hinton, "Backpropagation and the brain."
[29] A. Maboudi Reveshti, E. Khosravirad, A. K. Rouzbahani, S. K. Fariman, H. Najafi, and A. Peivandizadeh, "Energy consumption prediction in an office building by examining occupancy rates and weather parameters using the moving average method and artificial neural network," *Heliyon*, p. e25307, Feb. 2024, doi: 10.1016/j.heliyon.2024.e25307.
[30] R. Dastres and M. Soori, "Artificial Neural Network Systems," *International Journal of Imaging and Robotics (IJIR)*, vol. 21, no. 2, pp. 13–25, Sep. 2021.
[31] T. S. Agatha and R. Tyasnurita, "Perbandingan Klasifikasi Kredibilitas Pengguna Kartu Kredit Menggunakan Decision Tree dan Neural Network".
[32] I. M. Prakoso, W. Anggraeni, and A. Mukhlason, "Penerapan Case-Based Reasoning pada Sistem Cerdas untuk Pendeteksian dan Penanganan Dini Penyakit Sapi," *Sisfo*, vol. 4, no. 5, pp. 360–368, Sep. 2013, doi: 10.24089/j.sisfo.2013.09.007.
[33] M. M. Mijwil, "Artificial Neural Networks Advantages and Disadvantages," *Mesopotamian Journal of Big Data*, vol. 2021, pp. 29–31, Aug. 2021, doi: 10.58496/MJBD/2021/006.
[34] R. Rosenbaum, "On the relationship between predictive coding and backpropagation," *PLoS One*, vol. 17, no. 3 March, Mar. 2022, doi: 10.1371/journal.pone.0266102.
[35] T. A. Gunawan, M. Syahril, B. Kusuma, M. Cahyono, and J. Nugroho, "The application of backpropagation neural network method to estimate the sediment loads," 2017.
[36] B. Millidge, T. Salvatori, Y. Song, R. Bogacz, and T. Lukasiewicz, "Predictive Coding: Towards a Future of Deep Learning beyond Backpropagation?," pp. 1–10, Feb. 2022, [Online]. Available: http://arxiv.org/abs/2202.09467

[37] N. Lekbangpong, T. Srisawat, A. Wanichsombat, and J. Muangprathub, "The Control Model for Environmental Factor Effecting on Growth of St. John's Wort," in *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, IEEE, Jul. 2019, pp. 158–163. doi: 10.1109/JCSSE.2019.8864201.

[38] A. S. Abdulraheem *et al.*, "Home Automation System based on IoT," vol. 62, pp. 2453–2464, 2020, [Online]. Available: https://www.researchgate.net/publication/342561938

[39] M. Syarovy, A. P. Nugroho, and L. Sutiarso, "Pemanfaatan Model Neural Network Dalam Generasi Baru Pertanian Presisi Di Perkebunan Kelapa Sawit," *WARTA Pusat Penelitian Kelapa Sawit*, vol. 28, no. 1, pp. 39–54, Feb. 2023, doi: 10.22302/iopri.war.warta.v28i1.97.

[40] F. Aminifar, M. Abedini, T. Amraee, P. Jafarian, M. H. Samimi, and M. Shahidehpour, "A review of power system protection and asset management with machine learning techniques," *Energy Systems*, vol. 13, no. 4, pp. 855–892, Nov. 2022, doi: 10.1007/s12667-021-00448-6.

[41] Y. He, M. Kong, C. Du, D. Yao, and M. Yu, "Communication Security Analysis of Intelligent Transportation System Using 5G Internet of Things From the Perspective of Big Data," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2022, doi: 10.1109/TITS.2022.3141788.

[42] S. Siddiqui *et al.*, "Artificial Neural Network (ANN) Enabled Internet of Things (IoT) Architecture for Music Therapy," *Electronics (Basel)*, vol. 9, no. 12, p. 2019, Nov. 2020, doi: 10.3390/electronics9122019.

[43] M. A. Tawfeek, S. Alanazi, and A. A. A. El-Aziz, "Smart Greenhouse Based on ANN and IOT," *Processes*, vol. 10, no. 11, p. 2402, Nov. 2022, doi: 10.3390/pr10112402.

[44] X. Ma, H. Yu, Y. Wang, and Y. Wang, "Large-Scale Transportation Network Congestion Evolution Prediction Using Deep Learning Theory," *PLoS One*, vol. 10, no. 3, pp. 1–17, Mar. 2015, doi: 10.1371/journal.pone.0119044.

[45] Q. Ren, "Design and Implementation of Fault Diagnosis System for Power Internet of Things Equipment Based on Neural Network," *Mobile Information Systems*, vol. 2022, pp. 1–8, Aug. 2022, doi: 10.1155/2022/1887424.

[46] E. Bisong, "Batch vs. Online Learning," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Berkeley, CA: Apress, 2019, pp. 199–201. doi: 10.1007/978-1-4842-4470-8_15.

[47] W. M. Utomo, A. Bakar, M. Ahmad, T. Taufik, and R. Heriansyah, "Online Learning Neural Network Control of Buck-Boost Converter," in *2011 Eighth International Conference on Information Technology: New Generations*, IEEE, Apr. 2011, pp. 485–489. doi: 10.1109/ITNG.2011.216.

[48] A. V. Topalov and O. Kaynak, "Neural network modeling and control of cement mills using a variable structure systems theory based on-line learning mechanism," *J Process Control*, vol. 14, no. 5, pp. 581–589, Aug. 2004, doi: 10.1016/j.jprocont.2003.10.005.

[49] T. Nakama, "Theoretical analysis of batch and on-line training for gradient descent learning in neural networks," *Neurocomputing*, vol. 73, no. 1–3, pp. 151–159, Dec. 2009, doi: 10.1016/j.neucom.2009.05.017.

Halaman ini sengaja dikosongkan