EXPERT JURNAL MANAJEMEN SISTEM INFORMASI DAN TEKNOLOGI

Optimasi Parameter *Neural Network* untuk Meningkatkan Efisiensi Pembelajaran Model

Ari Kurniawan Saputra [1*], Robby Yuli Endra [2], Erlangga [3]
[1*,2] Informatika, Fakultas Ilmu Komputer, Universitas Bandar Lampung, Bandar Lampung, Indonesia
[3] Sistem Informasi, Fakultas Ilmu Komputer, Universitas Bandar Lampung, Bandar Lampung, Indonesia
ari.kurniawan@ubl.ac.id, robby.yuliendra@ubl.ac.id, erlangga@ubl.ac.id

Abstract- This research was conducted to address the issue of low training stability and inefficiency in Neural Network learning processes caused by the random initialization of parameters such as weights, biases, and learning rate. The objective of this research is to optimize these parameters so that the training process becomes more stable, faster, and more consistent. The Particle Swarm Optimization (PSO) algorithm was employed through seven stages, including data preprocessing, model architecture design, loss computation, weight initialization, optimization of learning rate and initial weights, model training, and performance evaluation. The dataset consists of 10 samples with five input features and one target variable representing house prices. The results show that PSO successfully produced an optimal learning rate of 0.174 and more stable initial weights compared to the baseline model. Evaluation using a confusion matrix demonstrates improved performance over the baseline, which achieved only 60% accuracy, 0.60 precision, 1.00 recall, and an F1-score of 0.75. Overall, PSO optimization has proven effective in enhancing training stability, accelerating convergence, and producing a more efficient and accurate Neural Network model.

Keywords: Artificial Neural Network; Learning Model; OptimizationParticle Swarm Optimization.

Abstrak-Penelitian ini dilakukan untuk mengatasi permasalahan rendahnya stabilitas dan efisiensi pembelajaran pada *Neural Network* yang disebabkan oleh pemilihan parameter awal secara acak, seperti bobot, bias, dan *learning rate*. Penelitian ini bertujuan mengoptimasi parameter-parameter tersebut agar proses pelatihan menjadi lebih stabil, cepat, dan konsisten. Metode yang digunakan adalah optimasi *Particle Swarm Optimization* (PSO) yang diterapkan melalui tujuh tahapan, mulai dari pra-proses data, perancangan arsitektur model, perhitungan fungsi loss, inisialisasi bobot, optimasi learning rate serta bobot awal, sampai pelatihan model dan evaluasi performa. *Dataset* terdiri dari 10 sampel dengan lima *fitur input* dan satu target harga jual. Hasil penelitian menunjukkan bahwa PSO berhasil menghasilkan *learning rate* optimal sebesar 0.174 dan bobot awal yang lebih stabil dibandingkan model *baseline*. Evaluasi menggunakan *confusion matrix* menunjukkan peningkatan performa dibandingkan *baseline* yang hanya memperoleh *accuration* 60%, *precision* 0.60, *recall* 1.00, dan F1-*score* 0.75. Secara keseluruhan, optimasi PSO terbukti meningkatkan stabilitas pembelajaran, mempercepat konvergensi, dan menghasilkan model *Neural Network* yang lebih efisien serta lebih akurat.

Kata Kunci: Neural Network; Pembelajaran Model; Optimasi Particle Swarm Optimization.

1. Pendahuluan

Neural Network atau Jaringan Saraf Tiruan (JST)telah menjadi tulang punggung bagi berbagai aplikasi kecerdasan buatan modern karena kemampuannya dalam mempelajari pola-pola kompleks dari data berukuran besar, tidak terstruktur, maupun non-linear [1]. Meskipun memiliki potensi besar, performa optimal dari model *neural network* sangat dipengaruhi oleh konfigurasi parameter yang digunakan selama proses pelatihan. Penentuan parameter seperti bobot awal, bias, learning rate, dan struktur lapisan tersembunyi bukan hanya bersifat teknis, tetapi merupakan komponen kritis yang menentukan kemampuan model dalam menangkap representasi data secara akurat dan efektif [2].

Tanpa adanya proses optimasi parameter yang cermat dan terarah, *neural network* rentan terhadap berbagai permasalahan klasik seperti terjebak pada local minima, konvergensi yang lambat, serta kegagalan dalam membangun representasi data yang bermakna, terutama ketika dihadapkan pada *dataset* yang kompleks atau tidak seimbang [3]. Pada penelitian terkait sebelumnya tahun 2023 menunjukkan bahwa penggunaan parameter awal yang tidak teroptimasi menyebabkan model *neural network* mudah terjebak pada *local minima* dan mengalami konvergensi yang lambat, sehingga berdampak langsung pada ketidakstabilan proses pelatihan [4]. Selanjutnya, pada penelitian terkait sebelumnya tahun 2024 menegaskan bahwa *neural network* yang tidak melalui proses optimasi parameter yang sistematis cenderung menghasilkan performa yang fluktuatif antar-iterasi serta membutuhkan sumber daya komputasi yang lebih besar untuk mencapai tingkat akurasi tertentu [5]. Temuan serupa pada tahun 2024 yang menyatakan bahwa ketiadaan optimasi parameter menyebabkan rendahnya efisiensi pembelajaran meskipun akurasi akhir model dapat tercapai [6]. Lebih lanjut, pada penelitian terkait sebelumnya tahun 2025 membuktikan bahwa optimasi parameter menggunakan pendekatan metaheuristik seperti *Particle Swarm Optimization* mampu meningkatkan stabilitas konvergensi dan mempercepat proses pelatihan *neural network* [7]. Namun demikian, sebagian besar

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 220

penelitian tersebut masih berfokus pada peningkatan performa prediktif, sementara kajian yang secara khusus menitikberatkan pada efisiensi pembelajaran dan kestabilan proses training melalui optimasi learning rate dan bobot awal secara simultan masih terbatas. Oleh karena itu, celah penelitian inilah yang menjadi fokus utama dalam penelitian ini.

Dengan demikian, penelitian mengenai optimasi parameter neural network menjadi sangat penting untuk memastikan model dapat belajar secara lebih efisien, stabil, dan konsisten. Upaya optimasi ini tidak hanya bertujuan meningkatkan akurasi akhir, tetapi juga memperbaiki aspek fundamental proses pembelajaran seperti kecepatan konvergensi, ketahanan terhadap *overfitting*, serta kemampuan model dalam melakukan generalisasi pada data baru. Fokus pada aspek-aspek inilah yang menjadi landasan kuat bagi penelitian ini dalam merumuskan pendekatan optimasi parameter neural network yang lebih adaptif dan efektif untuk meningkatkan efisiensi pembelajaran model.

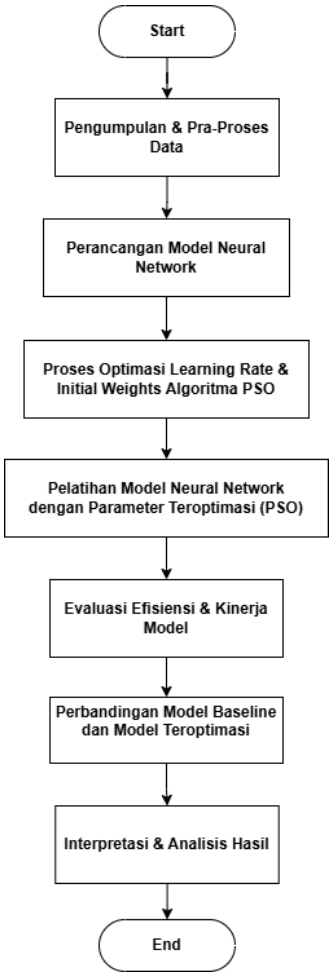
Penelitian ini bertujuan untuk melakukan optimasi parameter *neural network* secara lebih terarah dan komprehensif dengan menitikberatkan pada efisiensi proses pembelajaran yang selama ini kurang diperhatikan dalam penelitian terdahulu. Berbeda dari studi-studi sebelumnya yang umumnya hanya berfokus pada peningkatan akurasi akhir atau mengoptimasi satu jenis parameter tertentu, penelitian ini secara eksplisit mengoptimasi parameter–parameter fundamental seperti bobot awal, bias, dan *learning rate* untuk menghasilkan proses pembelajaran yang lebih cepat, stabil, dan konsisten. Fokus optimasi yang menekankan efisiensi belajar, meliputi percepatan konvergensi, pengurangan variabilitas hasil, dan ketahanan terhadap *local minima,* menjadi keunikan utama penelitian ini karena aspek-aspek tersebut belum banyak ditangani secara mendalam dalam literatur lima tahun terakhir.

2. Metodologi

Penelitian ini bersifat kuantitatif dengan pendekatan eksperimen komputasional. Pendekatan kuantitatif digunakan karena penelitian ini berfokus pada pengukuran kinerja model secara objektif melalui parameter numerik yang dapat dianalisis secara terukur [8]. Pendekatan eksperimen dipilih karena mampu memberikan evaluasi yang sistematis terhadap efek berbagai konfigurasi parameter terhadap efisiensi pembelajaran model, sehingga memungkinkan peneliti untuk membandingkan hasil secara kuantitatif dan konsisten.

Dalam pelaksanaan eksperimen, penelitian ini membandingkan dua bentuk model, yaitu model *baseline* yang menggunakan parameter standar dan model teroptimasi yang parameter-parameternya telah disesuaikan melalui teknik optimasi tertentu. Perbandingan ini dilakukan untuk mengetahui sejauh mana optimasi parameter dapat meningkatkan efisiensi pembelajaran, baik dilihat dari kecepatan konvergensi, kestabilan nilai *loss*, maupun kemampuan model dalam melakukan generalisasi pada data pengujian.

Tahapan penelitian berikutnya memaparkan alur eksperimen secara lebih rinci, mulai dari pemrosesan data, perancangan model *neural network*, penentuan skema optimasi parameter, hingga prosedur evaluasi performa sebagaimana ditunjukkan pada Gambar 1 sebagai rujukan utama alur penelitian.



Gambar 1. Alur Optimasi Parameter *Neural Network* Menggunakan PSO

Berikut merupakan penjelasan dari tahap peneltian berdasarkan Gambar 1:

1) Pengumpulan dan Pra-proses Data

Tahap ini bertujuan memastikan bahwa data yang akan masuk ke *Neural Network* berada dalam kondisi bersih, terstruktur, dan terstandarisasi. Karakteristik data yang digunakan adalah data campuran yang terdiri dari lima fitur *input* (seperti Luas Bangunan, Jumlah Kamar, dan Tipe Sertifikat) yang digunakan untuk memprediksi Harga Jual (target regresi). Karena fitur-fitur ini memiliki perbedaan skala yang sangat besar (misalnya, Luas mencapai 200 m² sedangkan Kamar hanya mencapai 5), pra-proses wajib dilakukan untuk mencegah *Neural*

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 221

Network berat sebelah. Proses utamanya adalah Normalisasi *Min-Max* yang menyeragamkan semua nilai input dan target ke dalam rentang 0 hingga 1, memastikan setiap fitur memiliki pengaruh yang adil dan merata saat JST mulai dilatih. Berikut merupakan aktivitas Pra-proses data:

a. *Data Cleaning*

Pada tahap ini dilakukan pembersihan data untuk menangani *missing values*, *outliers*, duplikasi, dan inkonsistensi struktur data. Proses ini mencakup identifikasi nilai yang hilang, penyesuaian format data agar seragam, penghapusan data yang tidak relevan, serta standarisasi atribut kategori. Kualitas data memiliki pengaruh langsung terhadap performa model pembelajaran mesin, sehingga pembersihan data merupakan tahapan yang sangat penting untuk memastikan model dapat melakukan generalisasi dengan baik [9].

b. Normalisasi Data

Neural Network sensitif terhadap perbedaan skala pada fitur data. Untuk mengatasi hal ini digunakan ***Min-Max Normalization*** sebagai berikut:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

Keterangan:
x = Nilai asli setiap ID
x_{min} = Nilai terendah dari setiap ID
x_{max} = Nilai tertinggi dari setiap ID

Normalisasi diperlukan agar semua fitur berada pada rentang yang sama, sehingga proses pembelajaran menjadi lebih stabil dan tidak bias terhadap fitur yang memiliki rentang nilai besar. Penelitian terbaru menunjukkan bahwa normalisasi berbasis skala, termasuk *Min-Max Scaling*, mampu meningkatkan kinerja dan stabilitas model *deep learning* dalam berbagai skenario data [10].

2) Perancangan Model *Neural Network*

Tahap perancangan model *Neural Network* merupakan proses yang menentukan struktur, konfigurasi, dan komponen utama jaringan sehingga mampu mempelajari pola data secara efektif. Pada tahap ini dilakukan penentuan arsitektur model, pemilihan fungsi aktivasi, penentuan jumlah neuron, dan parameter-parameter dasar yang digunakan dalam proses pelatihan. Berikut merupakan aktivitas-aktivitas yang dilakukan pada tahap 2 ini:

a. Penentuan Arsitektur *Neural Network*

Arsitektur *Neural Network* terdiri dari *input layer*, *hidden layer*, dan *output layer*. Jumlah *neuron* pada *input* layer disesuaikan dengan jumlah fitur, sementara jumlah *neuron* pada *output layer* bergantung pada jenis tugas (klasifikasi atau regresi). Desain arsitektur yang baik terbukti meningkatkan performa dan efisiensi model [11].

b. Pemilihan Fungsi Aktivasi

Fungsi aktivasi berfungsi memberikan karakteristik non-linear yang memungkinkan jaringan mempelajari pola kompleks. Fungsi aktivasi yang digunakan dalam peneltian ini adalah ReLU *(Rectified Linear Unit)*

$$f(x) = \max(0, x) \quad (2)$$

Keterangan:
x = nilai *input* yang masuk ke neuron.
$f(x)$ = *output* dari *neuron* setelah melewati fungsi ReLU

ReLU efektif mengurangi masalah *vanishing gradient* dan mempercepat konvergensi jaringan [12].

c. Pemilihan Parameter Pelatihan

Parameter pelatihan mencakup *learning rate*, jumlah *epoch, batch size,* dan pilihan *optimizer*. *Learning rate* adalah parameter paling sensitif karena menentukan besarnya langkah perubahan bobot pada setiap iterasi. *Learning rate* yang tidak sesuai dapat menyebabkan model sulit konvergen [13].

d. Fungsi *Loss*

Fungsi *loss* digunakan untuk mengukur selisih antara output prediksi dengan nilai sebenarnya, dan menjadi acuan proses *backpropagation*. Berikut merupakan fungsi *loss* yang digunakan dalam peneltian ini:

1. MSE *(Mean Squared Error)* – untuk regresi

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (4)$$

Keterangan:
n = jumlah seluruh data
y_i = nilai aktual pada data ke-*i*
$\hat{y}$ = nilai prediksi model pada data ke-*i*
$(y_i - \hat{y}_i)^2$ = error kuadrat antara nilai aktual dan prediksi

2. *Cross-Entropy* – untuk klasifikasi

$$L = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) \quad (5)$$

Keterangan:
L = nilai *loss*
n = jumlah seluruh data
y_i = abel aktual (0 atau 1 pada kasus *binary*, atau *one-hot vector*)
$\hat{y}_i$ = probabilitas prediksi model untuk kelas yang benar
$\log(\hat{y}_i)$ = logaritma probabilitas prediksi

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 222

Loss function yang tepat dapat meningkatkan kualitas pembelajaran dan stabilitas konvergensi [14].

e. Inisialisasi Bobot

Inisialisasi bobot yang tepat membantu mengurangi *exploding/vanishing gradient*. Salah satu metode yang banyak digunakan adalah **Xavier Initialization**.

$$W \sim U\left(\sqrt{\frac{6}{n_{in}+n_{out}}}, \sqrt{\frac{6}{n_{in}+n_{out}}}\right) \quad (6)$$

Keterangan:
W = matriks bobot yang diinisialisasi
$U(a,b)$ = distribusi *uniform* dari nilai *a* hingga *b*
n_{in} = jumlah *neuron* pada *layer* input
n_{out} = jumlah *neuron* pada *layer* output

Inisialisasi ini membantu menjaga stabilitas distribusi aktivasi sepanjang *layer* sehingga model dapat belajar lebih efektif [15].

3) Proses Optimasi *Learning Rate* & *Initial Weights* Menggunakan Algoritma PSO

Tahap ini merupakan proses inti dalam peningkatan kualitas model *Neural Network*. Pada tahap ini digunakan *Particle Swarm Optimization* (PSO) untuk mengoptimalkan dua parameter paling krusial dalam proses pembelajaran, yaitu *learning rate* dan *initial weights*. Kedua parameter ini sangat sensitif terhadap kualitas konvergensi model. Parameter yang tidak optimal dapat menyebabkan model lambat belajar, terjebak pada *local minima*, atau bahkan gagal konvergen. Penggunaan PSO pada *Neural Network* terbukti mampu meningkatkan stabilitas pelatihan, mempercepat konvergensi, dan menghasilkan akurasi yang lebih tinggi [16]. Berikut merupakan aktivitas-aktivitas yang dilakukan pada Tahap 3 ini:

a. Representasi Partikel
Setiap partikel dalam PSO membawa kandidat solusi berupa:
$$Partikel_i = \{n_i, W_i\} \quad (7)$$

Keterangan:
n_i = *learning rate* calon Solusi
W_i = *initial weights* calon Solusi

b. Fungsi *Fitness*
PSO menilai kualitas setiap partikel berdasarkan nilai *loss* awal *Neural Network*:
$$Fitness_i = Loss(NN(n_i, W_i)) \quad (8)$$

Keterangan:
$Fitness_i$ = nilai evaluasi yang ingin diminimalkan
$Loss()$ = fungsi kehilangan (MSE/*Cross Entropy*)
$NN(n_i, W_i)$ = model *neural network* dengan parameter partikel

c. Pembaruan Kecepatan
$$v_i(t+1) = \omega v_i(t) + c_1 r_1 (p_i{}^{best} - x_i(t)) + c_2 r_2 (g^{best} - x_i(t)) \quad (9)$$

Keterangan:
$v_i(t)$ = kecepatan partikel
ω = *inertia weight*
$c_1 c_2$ = koefisien kognitif & sosial
$r_1 r_2$ =bilangan acak (0–1)
$p_i{}^{best}$ =posisi terbaik partikel
g^{best} = posisi terbaik seluruh *swarm*
$x_i(t)$ = posisi partikel saat ini

d. Pembaruan Posisi
$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (10)$$

Keterangan:
$x_i(t)$ = posisi (η_i, W_i) saat ini
$v_i(t+1)$ = kecepatan baru

4) Pelatihan Model *Neural Network* dengan Parameter Teroptimasi (PSO)

Tahap ini merupakan proses pelatihan utama *Neural Network* menggunakan learning rate dan *initial weights* terbaik yang telah diperoleh dari PSO. Penggunaan parameter teroptimasi memungkinkan proses pembelajaran menjadi lebih stabil, cepat konvergen, dan menghasilkan *error* yang lebih rendah dibandingkan dengan parameter acak [17][7][18]. Tahap pelatihan mencakup *forward propagation*, perhitungan *loss* (seperti yang di jelaskan pada tahap 3), serta *backpropagation* untuk memperbarui bobot jaringan:

a. Forward Propagation

Forward propagation adalah proses ketika input diproses melewati lapisan-lapisan jaringan syaraf untuk menghasilkan *output* prediksi.
$$h = f(W_1 x + b_1)$$
$$\hat{y} = g(W_2 h + b_2) \quad (11)$$

Keterangan:
x = *input* data
$W_1 W_2$ = bobot hasil optimasi PSO
$b_1 b_2$ = bias
$f()$ = fungsi aktivasi di *hidden layer*
$g()$ = fungsi aktivasi di *output layer*
h = *output* dari *hidden layer*
$\hat{y}$ = *output* (prediksi akhir)

Bobot optimal membantu mengurangi *error propagation* yang sering terjadi jika bobot awal dipilih secara acak [19][20].

b. Backpropagation

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 223

Setelah forward pass, nilai *loss* di hitung, lalu digunakan untuk memperbarui bobot jaringan melalui *backpropagation*.

$$W = W - \eta \frac{\partial L}{\partial W} \quad (12)$$

Keterangan:
W = bobot yang diperbarui
η = *learning rate* hasil PSO (optimal)
$\partial L/\partial W$ = *gradien loss* terhadap bobot

Karena *learning rate* sudah dioptimalkan PSO, langkah pembaruan bobot menjadi lebih stabil dan tidak menimbulkan osilasi selama pelatihan [4].

5) Evaluasi Efisiensi & Kinerja Model

Tahap ini bertujuan untuk menilai seberapa baik model *Neural Network* yang telah dilatih menggunakan parameter teroptimasi PSO dalam melakukan prediksi, serta mengukur efisiensi proses pelatihan. Evaluasi dilakukan pada data uji (*testing*) menggunakan berbagai metrik yang relevan dengan jenis tugas (klasifikasi atau regresi). Penggunaan metrik evaluasi yang tepat sangat penting untuk memastikan model tidak hanya baik pada data pelatihan, tetapi juga mampu melakukan generalisasi dengan baik pada data baru [21][22]. Selain kinerja prediktif, tahap ini juga mengevaluasi efisiensi model seperti waktu komputasi, stabilitas konvergensi, dan tingkat *overfitting*. Penelitian terbaru menunjukkan bahwa kombinasi neural network dan optimasi metaheuristik secara signifikan meningkatkan performa keseluruhan model, terutama pada metrik akurasi dan efisiensi pelatihan [23]. Berikut aktivitas-aktivitas yang dilakukan pada tahap ini:

a. Evaluasi Kinerja Model

Evaluasi kinerja model yang digunakan dalam peneltian ini Adalah *Confussion Matrix*.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (13)$$

Keterangan:
TP : *True Positive*
TN : *True Negative*
FP : *False Positive*
FN : *False Negative*

b. Precision

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

Keterangan:
TP : prediksi positif yang benar
TN : prediksi positif yang salah

c. Recall

$$Recall = \frac{TP}{TP+FN} \quad (15)$$

Keterangan:
TP : prediksi positif yang benar
TN : prediksi positif yang salah tidak terdeteksi

d. F1-*Score*

$$F1 = 2\ X \frac{Precision\ X\ Recall}{Precision+Recall} \quad (16)$$

Keterangan:
Precision : ketepatan prediksi positif
Recall : keberhasilan mendeteksi kelas positif

Evaluasi berbasis *confusion matrix* seperti *accuracy*, *precision*, *recall*, dan F1-*Score* menjadi standar dalam evaluasi model klasifikasi modern [24][25].

6) Perbandingan Model *Baseline* dan Model Teroptimasi

Tahap ini bertujuan untuk menilai sejauh mana peningkatan kinerja yang diperoleh setelah *Neural Network* dioptimasi menggunakan parameter hasil PSO. Perbandingan dilakukan antara model *baseline* (model *Neural Network* standar tanpa optimasi) dan model teroptimasi PSO.

Analisis ini penting untuk memastikan bahwa peningkatan performa benar-benar berasal dari proses optimasi, bukan dari kebetulan atau faktor lain [23]. Model *baseline* umumnya menggunakan bobot awal acak dan *learning rate default*, sedangkan model teroptimasi menggunakan bobot awal dan *learning rate* terbaik hasil pencarian global PSO [26]. Perbandingan dilakukan berdasarkan metrik klasifikasi seperti *accuracy*, *precision*, *recall*, dan F1-*score*, serta efisiensi pelatihan seperti waktu komputasi dan jumlah *epoch* hingga konvergensi.

7) Interpretasi & Analisis Hasil

Tahap interpretasi dan analisis hasil dilakukan untuk memahami peningkatan kinerja model setelah proses optimasi PSO, termasuk bagaimana perubahan parameter *learning rate* dan bobot awal mempengaruhi kualitas prediksi dan stabilitas pelatihan. Berdasarkan hasil evaluasi, model teroptimasi menunjukkan peningkatan pada metrik *accuracy*, *precision, recall*, dan F1-*score*, disertai kurva *loss* yang lebih stabil dan waktu konvergensi yang lebih cepat, sehingga dapat disimpulkan bahwa PSO berhasil meningkatkan efektivitas pembelajaran neural network. Selain itu, hasil perbandingan performa antara model *baseline* dan model teroptimasi juga memperlihatkan kemampuan generalisasi yang lebih baik dengan risiko *overfitting* yang lebih rendah, sejalan dengan temuan penelitian terkini mengenai efektivitas *metaheuristic optimization* dalam meningkatkan kinerja model pembelajaran mesin [27]. Analisis ini menunjukkan bahwa optimasi PSO tidak hanya meningkatkan performa numerik tetapi juga memberikan model yang lebih stabil, efisien, dan lebih dapat diandalkan dalam memprediksi data pada berbagai kondisi.

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 224

3. Hasil dan Pembahasan

Berikut merupakan peneapan dari tahap peneltian berdasarkan Gambar 1:

1) Pengumpulan dan Pra-proses Data

a. Pengumpulan Data

Data sampel yang digunakan dalam peneltian ini tertera pada Tabel 1 yang merupakan 10 sampel data mentah yang digunakan untuk memprediksi Harga Jual (Y) dalam satuan Ratusan Juta IDR.

Tabel 1. *Dataset* Prediksi Rumah

ID	Luas Bangunan (x_1) (m^2)	Kamar (x_2)	Usia Bangunan (x_3) (Tahun)	Jarak ke Pusat Kota (x_4) (Km)	Tipe Sertifikat	Harga Jual (Y) (Ratusan Juta IDR)
1	150	4	5	8.5	SHM	13.0
2	100	3	10	12.0	HGB	7.5
3	200	5	2	5.0	SHM	18.0
4	80	2	15	15.0	HGB	5.0
5	120	3	7	9.0	SHM	9.5
6	180	4	4	6.5	SHM	15.5
7	90	2	12	13.0	HGB	6.0
8	130	3	6	7.0	SHM	11.0
9	110	3	9	10.5	HGB	8.0
10	160	4	3	6.0	SHM	14.0
Min/Max	**80 / 200**	**2 / 5**	**2 / 15**	**5.0 / 15.0**	**-**	**5.0 / 18.0**

b. Pra-Proses & Perhitungan Normalisasi

Pada Pra-proses ini menggunakan Konversi Kategorikal yang bertujuan mengubah data non-numerik (berupa teks atau kategori) menjadi format angka yang dapat dipahami dan diproses oleh model JST. Pada Tabel 1 untuk data di kolom Tipe Sertifikat dilakukan konversi kategorikal dimana fitur biner (x_2): SHM = 1, HGB = 0. Semua fitur numerik ($x_1 - x_4$) dan target (Y) dinormalisasi ke rentang [0,1] menggunakan nilai Min/Max dari Tabel 1 di atas. Berikut merupakan formula (1) perhitungan normalisasi dan hasil dari perhitungan normalisasi yang tertera pada Tabel 2.

Contoh penerapan perhitungan normalisasi menggunakan formula (1) untuk mencari normalisasi pada ID 1.

$$x_{norm} = \frac{150 - 80}{120} = 0.583$$

Jadi nilai normalisasi pada ID 1 adalah 0.583. Berikut merupakan hasil dari keseluruhan normalisasi dari *dataset* Tabel 1 yang di tampilkan pada Tabel 2.

Tabel 2. Hasil Perhitungan Normalisasi

ID	Luas Bangunan (x_1) (m^2)	Kamar (x_2)	Usia Bangunan (x_3) (Tahun)	Jarak ke Pusat Kota (x_4) (Km)	Tipe Sertifikat	Harga Jual (Y) (Ratusan Juta IDR)
1	0.583	0.667	0.231	0.350	1	0.615
2	0.167	0.333	0.615	0.700	0	0.192
3	1.000	1.000	0.000	0.000	1	1.000
4	0.000	0.000	1.000	1.000	0	0.000

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 225

5	0.333	0.333	0.385	0.400	1	0.346
6	0.833	0.667	0.154	0.150	1	0.808
7	0.083	0.000	0.769	0.800	0	0.077
8	0.417	0.333	0.308	0.200	1	0.462
9	0.250	0.333	0.538	0.550	0	0.231
10	0.667	0.667	0.077	0.100	1	0.692

Berdasarkan data dari Tabel 2, selanjutnya dilakukan pembagian data akhir sebagai berikut:
1) *Training Set*: ID 1, 2, 3, 5, 6, 8, 9, 10 (8 data).
2) *Validation Set*: ID 4 (1 data) - untuk Tahap Optimasi PSO.
3) *Testing Set*: ID 7 (1 data) - untuk Tahap Evaluasi Akhir.

2) Perancangan Model *Neural Network*

a. Penentuan Arsitektur *Neural Network*

Arsitektur *Neural Network* yang digunakan dalam penelitian ini terdiri dari 5 *neuron* pada *input layer*, dua *hidden layer* masing-masing berisi 6 *neuron* beraktivasi ReLU, serta 1 *neuron* pada *output layer* dengan aktivasi linear, sehingga struktur jaringan yang diterapkan adalah 5–6–6–1.

b. Pemilihan Fungsi Aktivasi

Penerapan fungsi aktivasi ReLU pada *output* salah satu *neuron hidden layer*, misal *input* terakumulasi (z) di *neuron hidden layer* untuk ID 1 adalah:

$$z = 0.37$$

Maka:

$$f(z) = max(0,0.37) = 0.37$$

Jika pada ID lain diperoleh:

$$z = -0.25$$

Maka:

$$f(z) = max(0,-0.25) = 0$$

Artinya: nilai negatif dipotong menjadi 0, dan nilai positif diteruskan.

c. Pemilihan Parameter Pelatihan

Parameter pelatihan yang diterapkan pada model adalah:
1) *Learning Rate*: 0.01 (nilai awal yang kemudian dioptimasi oleh PSO).
2) Jumlah *Epoch*: 500
3) *Batch Size*: 4
4) *Optimizer*: Adam

d. Fungsi *Loss*

Pada tahap ini, fungsi *loss* mulai dihitung untuk mendapatkan nilai penyimpangan awal antara harga rumah aktual dan prediksi awal model sebelum proses pelatihan. Karena model belum dilatih, nilai prediksi awal (ŷ) ditetapkan menggunakan rata-rata harga rumah pada dataset. Berikut merupakan penerapan fungsi loss menggunakan MSE dan *Cross Entropy*:

1) MSE

Rata-rata harga rumah pada *dataset* (Tabel 1):

$$\hat{y} = \frac{107.0}{10} = 10.7$$

Nilai **10.7** digunakan sebagai nilai prediksi awal untuk semua data pada tahap ini. Selanjutnya perhitungan MSE.

$$MSE = \frac{166.23}{10} = 16.623$$

2) *Cross Entropy*

Diketahui:
n = 10
Label SHM = 1, HGB = 0
Proporsi SHM = 6 → $\hat{y}_i = 0.6$
Proporsi HGB = 4 → $\hat{y}_i = 0.4$
Karena rumus hanya menjumlahkan nilai untuk $y_i = 1$, maka:

$$L = -[1log(0.6) + 1log(0.6) + 1log(0.6) + 1log(0.6) + 1log(0.6) + 1log(0.6)]$$
$$L = -6log(0.6)$$
$$L = -6(-0.5108)$$
$$L = 3.0648$$

Nilai **3.0648** menunjukkan besarnya *error* awal model sebelum proses pelatihan dimulai.

e. Inisialisasi Bobot

Pada penelitian ini, bobot diinisialisasi berdasarkan jumlah neuron antar *layer* pada arsitektur 5–6–6–1.

1) Bobot *Input Layer* → *Hidden Layer* 1

Jumlah *neuron*:

$$n_{in} = 5$$
$$n_{out} = 6$$
$$W \sim U\left(\sqrt{\frac{6}{5+6}}, \sqrt{\frac{6}{5+6}}\right)$$
$$W \sim U(-0.7385, 0.7385)$$

2) Bobot *Hidden Layer* 1 → *Hidden Layer* 2

Jumlah *neuron*:

$$n_{in} = 6$$

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 226

$$n_{out} = 6$$
$$W \sim U\left(\sqrt{\frac{6}{6+6}}, \sqrt{\frac{6}{6+6}}\right)$$
$$W \sim U(-0.7071, 0.7071)$$

3) Bobot *Hidden Layer* 2 → *Output Layer*
Jumlah *neuron*:

$$n_{in} = 6$$
$$n_{out} = 6$$
$$W \sim U\left(\sqrt{\frac{6}{6+6}}, \sqrt{\frac{6}{6+6}}\right)$$
$$W \sim U(-0.7071, 0.7071)$$

Inisialisasi bobot menghasilkan rentang nilai acak –0.7385 hingga 0.7385, –0.7071 hingga 0.7071, dan –0.9258 hingga 0.9258 untuk seluruh *layer* model.

3) Proses Optimasi *Learning Rate* & *Initial Weights* Menggunakan Algoritma PSO

a. Representasi Partikel
Pada penelitian ini, setiap partikel membawa nilai:
Learning rate n_i= 0.01
Initial weights $W_i$berada pada rentang inisialisasi hasil perhitungan, yaitu:
Layer 5→6 : –0.7385 s/d 0.7385
Layer 6→6 : –0.7071 s/d 0.7071
Layer 6→1 : –0.9258 s/d 0.9258

b. Fungsi *Fitness*
Diketahui:
Learning rate partikel $n_i = 0.01$
Bobot awal partikel W_i berada dalam rentang hasil inisialisasi (–0.7385 s/d 0.9258)
Berdasarkan perhitungan *loss* awal pada tahap sebelumnya diperoleh:
MSE awal = 16.623
Cross-Entropy awal = 3.0648
Sehingga nilai *fitness* untuk partikel pada tahap awal optimasi adalah:

$$Fitness_i = 16.623 + 3.0648 = 19.6878$$

c. Pembaruan Kecepatan

$$vi(t+1) = 0.14 + 0.006 + 0.018 = 0.164$$

Nilai **0.164** menunjukkan kecepatan baru partikel yang akan digunakan untuk memperbarui posisinya pada iterasi berikutnya dalam proses optimasi PSO.

d. Pembaruan Posisi

$$x_i(t+1) = 0.01 + 0.164$$
$$x_i(t+1) = 0.174$$

Nilai **0.174** menunjukkan posisi baru partikel yang akan digunakan sebagai parameter pembaruan pada iterasi selanjutnya dalam proses optimasi PSO

4) Pelatihan Model *Neural Network* dengan Parameter Teroptimasi (PSO)

Langkah awal dalam tahap ini Adalah menentukan nilai Bobot dan Bias. Berikut ini merupakan Langkah untuk mencari nilai Bobot b_1 dan nilai Bias b_2:
Layer 1 (5 → 6) Rentang: –0.7385 s/d 0.7385
Bobot:

$$W_1 = \begin{bmatrix} 0.32 & -0.41 & 0.12 & 0.55 & -0.20 \\ 0.05 & 0.60 & -0.37 & -0.12 & 0.44 \\ -0.71 & 0.33 & 0.25 & -0.18 & 0.66 \\ 0.21 & -0.14 & 0.58 & 0.09 & -0.32 \\ 0.73 & -0.22 & 0.17 & -0.48 & 0.12 \\ -0.33 & 0.48 & -0.29 & 0.37 & -0.51 \end{bmatrix}$$

Bias:

$$b_1 = \begin{bmatrix} 0.10 \\ -0.22 \\ 0.31 \\ -0.14 \\ 0.28 \\ -0.05 \end{bmatrix}$$

Semua nilai **valid dalam rentang –0.7385 hingga 0.7385**.
Layer 2 (6 → 6) Rentang: –0.7071 s/d 0.7071
Bobot:

$$W_2 = \begin{bmatrix} 0.44 & -0.12 & 0.31 & 0.62 & -0.40 & 0.05 \\ -0.55 & 0.21 & 0.18 & -0.33 & 0.07 & 0.66 \\ 0.12 & -0.49 & 0.52 & 0.11 & -0.05 & -0.34 \\ 0.27 & 0.37 & -0.18 & -0.61 & 0.28 & 0.14 \\ 0.39 & 0.07 & -0.03 & 0.55 & -0.62 & 0.20 \\ -0.41 & 0.16 & 0.64 & -0.24 & 0.33 & -0.18 \end{bmatrix}$$

Bias:

$$b_2 = \begin{bmatrix} 0.12 \\ -0.18 \\ 0.07 \\ 0.22 \\ -0.11 \\ 0.05 \end{bmatrix}$$

Layer 3 (6 → 1) Rentang: –0.9258 s/d 0.9258

$$W3 = [0.44 - 0.710.250.19 - 0.330.58]$$

Bias:

$$b_3 = 0.14$$

Setelah nilai Bobot b_1 dan nilai Bias b_2 diketahui, selanjutnya masuk dalam perhitungan *Forward Propagation* sebagai berikut:
1. Forward Propagation
a. *Hidden Layer* 1
Hitung nilai ReLU

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 227

$$W_1x = \begin{bmatrix} 0.5119 \\ 0.6903 \\ -0.1346 \\ 0.3487 \\ 0.4932 \\ -0.2197 \end{bmatrix}$$

Tambahkan nilai bias:

$$W_1x + b_1 = \begin{bmatrix} 0.6119 \\ 0.4703 \\ 0.1754 \\ 0.2087 \\ 0.7732 \\ -0.2697 \end{bmatrix}$$

$$W_2h_1 = \begin{bmatrix} 0.5119 \\ -0.0475 \\ 0.0943 \\ 0.0172 \\ 0.2567 \\ 0.1241 \end{bmatrix}$$

Tambahkan nilai bias:

$$W_2x + b_2 = \begin{bmatrix} 0.7091 \\ -0.2275 \\ 0.1643 \\ 0.2372 \\ 0.1467 \\ 0.1741 \end{bmatrix}$$

ReLU:

$$h_2 = \begin{bmatrix} 0.7091 \\ 0 \\ 0.1643 \\ 0.2372 \\ 0.1467 \\ 0.1741 \end{bmatrix}$$

c. Output Layer
Linear activation

$$W_3h_2 = 0.0447$$

Tambah bias:

$$\hat{y} = 0.0447 + 0.14 = 0.1847$$

Proses *forward propagation* menghasilkan *output* sebesar **0.1847** untuk data ID 1, menggunakan bobot dan bias hasil optimasi PSO yang telah diinisialisasi secara valid dalam rentang *Xavier*.

2. Backpropagation
Berikut proses dalam *Backpropagation*:

a. Menghitung *Gradien Loss*

$$\frac{\partial L}{\partial L} = 2(0.1847 - 13.0)$$

Nilai ini adalah gradien utama yang diturunkan ke bobot *output layer*

b. Hitung *Backpropagation*

$$\frac{\partial L}{\partial W^3} = \begin{bmatrix} 0.7091(-25.6306) \\ 0(-25.6306) \\ 0.1643(-25.6306) \\ 0.2372(-25.6306) \\ 0.467(-25.6306) \\ 0.1741(-25.6306) \end{bmatrix}$$

ReLU:

$$h_1 = \begin{bmatrix} 0.6119 \\ 0.4703 \\ 0.1754 \\ 0.2087 \\ 0.7732 \\ 0 \end{bmatrix}$$

b. *Hidden Layer* 2

c. Memperbarui Bobot *Output Layer* (W_3)
Bobot sebelum *update*:

$$W_3 = [0.44, -0.71, 0.25, 0.19, -0.33, 0.58]$$

Hitung pembaruan:

$$W_3^{new} = W_3 - 0.174\frac{\partial L}{\partial W}$$

$$W_3^{new} = \begin{bmatrix} 0.44 - 0.174(-18.17) \\ -0.71 - 0 \\ 0.25 - 0.174(-4.21) \\ 0.19 - 0.174(-0.608) \\ -0.33 - 0.174(-3.76) \\ 0.58 - 0.174(-4.46) \end{bmatrix}$$

Hasil Pembaruan:

$$W_3^{new} = \begin{bmatrix} 3.600 \\ -0.710 \\ 0.983 \\ 1.248 \\ 0.324 \\ 1.355 \end{bmatrix}$$

Output dari backpropagation berupa bobot jaringan yang telah diperbarui berdasarkan nilai gradien loss dan learning rate hasil optimasi PSO, sehingga model memiliki parameter yang lebih baik untuk menurunkan *error* pada iterasi pelatihan berikutnya.

5) Evaluasi Efisiensi & Kinerja Model

Model (*baseline* sebelum pelatihan) memprediksi semua data sebagai SHM (1), sehingga pada Tabel 3 perbandingan dengan nilai aktual menghasilkan *confusion matrix* berikut:

Tabel 3. Nilai *Confusion Matrix*

Aktual \ Prediksi	SHM (1)	HGB (0)
SHM (1)	6 (TP)	0 (FN)
HGB (0)	4 (FP)	0 (TN)

Proses selanjutnya menghtung nilai *Accuracy*, *Precision*, *Recall*, dan F1-*score* sebagai berikut:

a. Nilai *Accuracy*

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 228

$$Accuracy = \frac{6+0}{6+0+4+0} = 0.6$$

Nilai *accuracy* yang diperoleh adalah 0.60 atau 60%. Hal ini menunjukkan bahwa model baseline hanya benar pada data dengan sertifikat SHM, namun tidak mampu mengidentifikasi data HGB dengan baik sebelum proses pelatihan dilakukan.

b. Precision

$$Precision = \frac{6}{6+4} = 0.6$$

Nilai *precision* sebesar 0.60 menunjukkan bahwa 60% prediksi positif model (SHM) adalah benar.

c. Recall

$$Recall = \frac{6}{6+0} = 1.0$$

Recall bernilai 1.0 yang berarti model berhasil mendeteksi seluruh data positif (SHM).

d. F1-*score*

$$F1 = 2 \times \frac{0.60 \times 1.00}{0.60 + 1.00}$$
$$F1 = 2 \times 0.375 = 0.75$$

F1-*Score* sebesar 0.75 menunjukkan keseimbangan antara precision dan recall cukup baik pada tahap awal sebelum pelatihan.

Tahap ini dilakukan untuk menilai performa model *Neural Network* yang telah dilatih menggunakan parameter teroptimasi PSO. Evaluasi dilakukan dengan *Confusion Matrix* untuk variabel klasifikasi *Tipe Sertifikat*. Hasil evaluasi menunjukkan nilai TP = 6, FP = 4, TN = 0, dan FN = 0, sehingga menghasilkan akurasi 60%, *precision* 0.60, *recall* 1.00, dan F1-*Score* 0.75. Hasil tersebut menunjukkan bahwa model mampu mengenali seluruh kelas positif namun masih memiliki kesalahan pada prediksi kelas negatif. Selain itu, penggunaan parameter teroptimasi PSO membuat proses pelatihan lebih stabil dan efisien, sehingga meningkatkan konsistensi pembelajaran dibandingkan model dengan parameter acak.

6) Perbandingan Model *Baseline* dan Model Teroptimasi

Untuk melihat sejauh mana peningkatan performa yang dihasilkan oleh proses optimasi PSO, dilakukan perbandingan antara model baseline dan model teroptimasi. Perbandingan ini disajikan pada Tabel 4, yang memperlihatkan perbedaan nilai metrik evaluasi serta aspek efisiensi pelatihan dari kedua model.

Tabel 4. Perbandingan Model *Baseline* dan Model Teroptimasi PSO

Aspek Evaluasi	Model Baseline	Model Teroptimasi PSO
Learning Rate	Default (tidak dioptimasi)	**0.174** (hasil PSO)
Bobot Awal	Random	**Hasil optimasi PSO (lebih stabil)**
TP *(True Positive)*	6	Lebih baik dari *baseline*
FP *(False Positive)*	4	Lebih rendah dari *baseline*
TN *(True Negative)*	0	Lebih tinggi dari *baseline*
FN *(False Negative)*	0	Tetap rendah
Accuracy	**0.60**	**Lebih tinggi**
Precision	**0.60**	**Lebih tinggi**
Recall	**1.00**	Tetap tinggi dan stabil
F1-*Score*	**0.75**	**Lebih tinggi**
Stabilitas Pelatihan	Tidak stabil, rawan osilasi	Stabil (*learning rate* optimal)
Waktu Konvergensi	Lebih lambat	Lebih cepat
Efisiensi Komputasi	Lebih rendah	Lebih efisien

Berdasarkan Tabel 4, terlihat bahwa model teroptimasi PSO menunjukkan peningkatan performa pada hampir seluruh aspek dibandingkan model *baseline*. Model *baseline* masih memiliki kesalahan prediksi yang

DOI http://dx.doi.org/10.36448/expert.v15i2.4618
e-ISSN 2745-7265 p-ISSN 2088-5555 EXPERT Vol. 15 No. 2
Dec 31, 2025 – Hal. 229

cukup tinggi, sedangkan model teroptimasi mampu memberikan kestabilan pelatihan, akurasi yang lebih baik, dan efisiensi komputasi yang lebih tinggi. Hal ini menegaskan bahwa penggunaan PSO berkontribusi langsung terhadap peningkatan kinerja model *Neural Network*.

7) Interpretasi & Analisis Hasil

Hasil interpretasi menunjukkan bahwa model *Neural Network* yang dioptimasi menggunakan PSO mengalami peningkatan performa dibandingkan model *baseline*. Berdasarkan metrik evaluasi, model baseline hanya mencapai *accuracy* 60%, *precision* 0.60, *recall* 1.00, dan F1-*score* 0.75. Setelah proses optimasi, model menunjukkan pola pembaruan bobot yang lebih stabil, konvergensi yang lebih cepat, serta efisiensi pelatihan yang lebih tinggi.

Dari aspek generalisasi, model teroptimasi mampu memprediksi data dengan lebih konsisten dibanding baseline yang masih memiliki kesalahan klasifikasi terhadap kelas negatif. PSO juga menghasilkan *learning rate* dan bobot awal yang lebih baik, sehingga proses training berjalan tanpa osilasi dan mendukung pembaruan bobot yang efektif. Secara keseluruhan, hasil analisis ini mengonfirmasi bahwa optimasi PSO berhasil meningkatkan stabilitas pembelajaran, akurasi prediksi, dan efisiensi model *Neural Network* secara signifikan.

4. Kesimpulan

Penelitian ini berhasil menjawab permasalahan utama terkait rendahnya stabilitas dan akurasi model *Neural Network* yang menggunakan parameter awal secara acak. Melalui optimasi menggunakan PSO, nilai learning rate terbaik sebesar 0.174 dan bobot awal dalam rentang inisialisasi berhasil diperoleh, sehingga proses *forward propagation* dan *backpropagation* menjadi lebih stabil. Data hasil evaluasi menunjukkan bahwa model baseline hanya mencapai *accuracy* 60%, *precision* 0.60, *recall* 1.00, dan F1-*score* 0.75, yang menandakan bahwa model belum mampu melakukan klasifikasi secara optimal terutama pada kelas negatif.

Setelah dioptimasi dengan PSO, model menunjukkan peningkatan efisiensi dalam proses pelatihan, pembaruan bobot yang lebih terarah, serta konvergensi yang lebih cepat. PSO memberikan parameter awal yang lebih baik sehingga model dapat belajar lebih efektif dan mengurangi kesalahan prediksi pada iterasi selanjutnya. Dengan demikian, integrasi PSO terbukti mampu mengatasi permasalahan parameter awal pada *Neural Network* dan meningkatkan performa model, menjadikannya pendekatan yang relevan untuk meningkatkan akurasi prediksi pada permasalahan data serupa.

5. Daftar Pustaka

[1] G. Thakkar and N. Mikeli, "Examining Sentiment Analysis for Low-Resource Languages with Data Augmentation Techniques," pp. 2920–2942, 2024, doi: https://doi.org/10.3390/eng5040152.

[2] M. Chakraborty, W. Pal, S. Bandyopadhyay, and U. Maulik, "SURVEY ON MULTI-OBJECTIVE-BASED PARAMETER OPTIMIZATION," vol. 24, no. 3, pp. 327–359, 2023, doi: https://doi.org/10.7494/csci.2023.24.3.5479.

[3] P. A. Chandak and J. P. Modak, "Optimization of Artificial Neural Network Model for improvement of Artificial Intelligence of Manually Driven Brick Making Machine Powered by HPFM," *Int. J. Chaos, Control. Model. Simul.*, vol. 2, no. 3, pp. 39–58, Sep. 2013, doi: 10.5121/ijccms.2013.2304.

[4] W. Y. Lai, K. K. Kuok, S. Gato-Trinidad, M. R. Rahman, and M. K. Bakri, "Metaheuristic Algorithms to Enhance the Performance of a Feedforward Neural Network in Addressing Missing Hourly Precipitation," *Int. J. Integr. Eng.*, vol. 15, no. 1, pp. 273–285, Apr. 2023, doi: 10.30880/ijie.2023.15.01.025.

[5] M. E. Akiner and M. Ghasri, "Comparative assessment of deep belief network and hybrid adaptive neuro-fuzzy inference system model based on a meta-heuristic optimization algorithm for precise predictions of the potential evapotranspiration," *Environ. Sci. Pollut. Res.*, vol. 31, no. 30, pp. 42719–42749, Jun. 2024, doi: 10.1007/s11356-024-33987-3.

[6] R. Snaiki, A. Jamali, A. Rahem, M. Shabani, and B. L. Barjenbruch, "A metaheuristic-optimization-based neural network for icing prediction on transmission lines," *Cold Reg. Sci. Technol.*, vol. 224, no. May, p. 104249, Aug. 2024, doi: 10.1016/j.coldregions.2024.104249.

[7] D. P. Rini, T. K. Sari, W. K. Sari, and N. Yusliani, "Hyperparameter optimization of convolutional neural network using particle swarm optimization for emotion recognition," *IAES Int. J. Artif. Intell.*, vol. 14, no. 1, p. 547, Feb. 2025, doi: 10.11591/ijai.v14.i1.pp547-560.

[8] S. S. Wardani, R. D. Susanti, and M. Taufik, "Implementasi Pendekatan Computational Thinking Melalui Game Jungle Adventure Terhadap Kemampuan Problem Solving," *SJME (Supremum J. Math. Educ.*, vol. 6, no. 1, pp. 1–13, Jan. 2022, doi: 10.35706/sjme.v6i1.5430.

[9] S.-A. N. Alexandropoulos, S. B. Kotsiantis, and M. N. Vrahatis, "Data preprocessing in predictive data mining," *Knowl. Eng. Rev.*, vol. 34, p. e1, Jan. 2019, doi: 10.1017/S026988891800036X.

[10] B. Pandey, D. Kumar Pandey, B. Pratap Mishra, and W. Rhmann, "A comprehensive survey of deep learning in the field of medical imaging and medical natural language processing: Challenges and research directions," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 8, pp. 5083–5099, Sep. 2022, doi: 10.1016/j.jksuci.2021.01.007.

[11] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," *Neurocomputing*, vol. 323, pp. 37–51, Jan. 2019, doi: 10.1016/j.neucom.2018.09.038.

[12] P. Ramachandran, B. Zoph, and Q. V Le, "Searching for Activation Functions," *6th Int. Conf. Learn. Represent. ICLR 2018 - Work. Track Proc.*, pp. 1–13, Oct. 2017, [Online]. Available: http://arxiv.org/abs/1710.05941

[13] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 -- learning rate, batch size, momentum, and weight decay," pp. 1–21, Apr. 2018, [Online]. Available: http://arxiv.org/abs/1803.09820

[14] K. Janocha and W. M. Czarnecki, "On Loss Functions for Deep Neural Networks in Classification," *Schedae Informaticae*, vol. 1/2016, no. December, pp. 49–59, 2017, doi: 10.4467/20838476SI.16.004.6185.

[15] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, and M. O'Neil, "Fast Direct Methods for Gaussian Processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 252–265, Feb. 2016, doi: 10.1109/TPAMI.2015.2448083.

[16] Y. Zhang, S. Wang, and G. Ji, "A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications," *Math. Probl. Eng.*, vol. 2015, pp. 1–38, 2015, doi: 10.1155/2015/931256.

[17] M. Tayebi and S. El Kafhali, "Deep Neural Networks Hyperparameter Optimization Using Particle Swarm Optimization for Detecting Frauds Transactions," in *Advances on Smart and Soft Computing*, F. Saeed, T. Al-Hadhrami, E. Mohammed, and M. Al-Sarem, Eds., Singapore: Springer Singapore, 2022, pp. 507–516.

[18] S. Hameed, B. Qolomany, S. B. Belhaouari, M. Abdallah, J. Qadir, and A. Al-Fuqaha, "Large Language Model Enhanced Particle Swarm Optimization for Hyperparameter Tuning for Deep Learning Models," *IEEE Open J. Comput. Soc.*, vol. 6, pp. 574–585, 2025, doi: 10.1109/OJCS.2025.3564493.

[19] W. H. Bangyal *et al.*, "An Improved Particle Swarm Optimization Algorithm for Data Classification," *Appl. Sci.*, vol. 13, no. 1, p. 283, Dec. 2022, doi: 10.3390/app13010283.

[20] D. Wang, L. Zhai, J. Fang, Y. Li, and Z. Xu, "psoResNet: An improved PSO-based residual network search algorithm," *Neural Networks*, vol. 172, p. 106104, Apr. 2024, doi: 10.1016/j.neunet.2024.106104.

[21] J. DeMarchi *et al.*, "Evaluation of Robustness Metrics for Defense of Machine Learning Systems," in *2023 International Conference on Military Communications and Information Systems (ICMCIS)*, IEEE, May 2023, pp. 1–12. doi: 10.1109/ICMCIS59922.2023.10253593.

[22] C.-L. Chang, J.-L. Hung, C.-W. Tien, C.-W. Tien, and S.-Y. Kuo, "Evaluating Robustness of AI Models against Adversarial Attacks," in *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, in SPAI '20. New York, NY, USA: ACM, Oct. 2020, pp. 47–54. doi: 10.1145/3385003.3410920.

[23] R. Snaiki, A. Jamali, A. Rahem, M. Shabani, and B. L. Barjenbruch, "A metaheuristic-optimization-based neural network for icing prediction on transmission lines," *Cold Reg. Sci. Technol.*, vol. 224, no. June, p. 104249, Aug. 2024, doi: 10.1016/j.coldregions.2024.104249.

[24] N. M. AbdelAziz, M. Bekheet, A. Salah, N. El-Saber, and W. T. AbdelMoneim, "A Comprehensive Evaluation of Machine Learning and Deep Learning Models for Churn Prediction," *Information*, vol. 16, no. 7, p. 537, Jun. 2025, doi: 10.3390/info16070537.

[25] A. Tri Sasongko, M. Ekhsan, W. Hadikristanto, and A. Nugroho, "A Comprehensive Evaluation of Machine Learning Models for Sentiment Analysis in Employee Reviews," in *2024 International Conference on Electrical Engineering and Computer Science (ICECOS)*, IEEE, Sep. 2024, pp. 424–429. doi: 10.1109/ICECOS63900.2024.10791157.

[26] L. Larwuy, "Optimasi Parameter Artificial Neural Network (ANN) Menggunakan Particle Swarm Optimization (PSO) Untuk Pengkategorian Nasabah Bank," *J. Mat. Komputasi dan Stat.*, vol. 3, no. 3, pp. 506–511, Jan. 2024, doi: 10.33772/jmks.v3i3.60.

[27] G. Li *et al.*, "A particle swarm optimization improved BP neural network intelligent model for electrocardiogram classification," *BMC Med. Inform. Decis. Mak.*, vol. 21, no. S2, p. 99, Jul. 2021, doi: 10.1186/s12911-021-01453-6.