UIN IMAM BONJOL PADANG

MAp Journal
Mathematics & Applications

Web : ejournal.uinib.ac.id/jurnal/index.php/MAp
Email : map_journal@uinib.ac.id

PENGGUNAAN *PARTICLE SWARM OPTIMIZATION* PADA JARINGAN SYARAF TIRUAN UNTUK KLASIFIKASI SINYAL RADAR

PARTICLE SWARM OPTIMIZATION IN ARTIFICIAL NEURAL NETWORKS FOR RADAR SIGNAL CLASSIFICATION

Mohammad Jamhuri[1§], Tri Utomo[2]

[1]Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim [m.jamhuri@live.com]
[2]Program Studi Matematika, Institut Teknologi Sumatra [three1st@gmail.com]
[§]*Corresponding Author*

Received 16^{th} Jun 2024; Accepted 02^{nd} Des 2024; Published 10^{th} Des 2024;

Abstrak

Klasifikasi sinyal radar merupakan salah satu tugas penting yang memiliki aplikasi luas, termasuk dalam domain militer, navigasi, dan pengawasan cuaca. Jaringan Syaraf Tiruan (JST) telah terbukti efektif dalam menyelesaikan tugas klasifikasi kompleks berkat kemampuannya dalam memodelkan pola dan hubungan non-linear dalam data. Salah satu tantangan mendasar dalam implementasi JST adalah penentuan jumlah node optimal pada *hidden layer*, yang secara signifikan memengaruhi performa model. Penelitian ini mengusulkan pendekatan berbasis *Particle Swarm Optimization* (PSO) untuk mengoptimalkan konfigurasi JST dalam klasifikasi sinyal radar. PSO, sebagai algoritma optimasi berbasis populasi yang terinspirasi dari perilaku sosial kawanan, memungkinkan eksplorasi ruang solusi secara lebih efisien dan efektif dibandingkan metode tradisional. Hasil penelitian menunjukkan bahwa penerapan PSO pada JST secara signifikan meningkatkan metrik performa model, termasuk *accuracy*, *precision*, *recall*, dan *F1-score*, dibandingkan dengan metode baseline. Namun demikian, penggunaan PSO tidak memberikan peningkatan efisiensi dalam hal waktu komputasi. Temuan ini memberikan kontribusi penting dalam pengembangan model pembelajaran mesin yang lebih akurat untuk aplikasi praktis seperti pengawasan cuaca dan sistem pertahanan, sekaligus memperkaya kajian teoretis di bidang optimasi dan jaringan syaraf tiruan.

Kata Kunci: Jaringan Syaraf Tiruan, Klasifikasi Sinyal, Optimasi *Swarm*, Pembelajaran Mesin.

Abstract

Radar signal classification is a critical task with broad applications, including military operations, navigation, and weather surveillance. Artificial Neural Networks (ANNs) have proven highly effective in addressing complex classification tasks due to their capability to model nonlinear patterns and relationships in data. A fundamental challenge in implementing ANNs lies in determining the optimal number of nodes in the hidden layer, which significantly impacts the model's performance. This study proposes a Particle Swarm Optimization (PSO)-based approach to optimize ANN configurations for radar signal classification. PSO, a population-based optimization algorithm inspired by the social behavior of swarms, enables a more effective and comprehensive exploration of the solution space compared to traditional methods. The findings demonstrate that applying PSO to ANNs significantly improves performance metrics such as accuracy, precision, recall, and F1-score when compared to baseline methods. However, the use of PSO does not enhance computational efficiency in terms of execution time. These results provide valuable contributions to the development of more accurate machine learning models

for practical applications, such as weather monitoring and defense systems, while also enriching theoretical studies in the fields of optimization and artificial neural networks.
***Keywords**: Artificial Neural Network, Machine Learning, Signal Classification, Swarm Optimization.*

1. Pendahuluan

Sinyal radar memainkan peran penting dalam berbagai aplikasi, termasuk militer, navigasi, dan pengawasan cuaca [1]. Kemampuan untuk mengklasifikasikan sinyal radar dengan akurasi tinggi sangat penting untuk meningkatkan efektivitas dan efisiensi sistem radar [2]. Dalam beberapa dekade terakhir, metode pembelajaran mesin, khususnya jaringan syaraf tiruan (JST), telah menunjukkan kinerja yang sangat baik dalam tugas-tugas klasifikasi yang kompleks [3]. JST mampu menangkap pola dan hubungan non-linear dalam data, membuatnya menjadi pilihan yang populer untuk analisis sinyal radar [4].

Pemilihan arsitektur JST yang optimal, khususnya jumlah *node* dalam *hidden layer*, merupakan tantangan yang signifikan [5]. Jumlah *node* yang tidak optimal dapat menyebabkan JST mengalami *overfitting* atau *underfitting*, yang pada gilirannya dapat mengurangi akurasi klasifikasi [6]. Oleh karena itu, diperlukan metode yang efektif untuk menentukan jumlah *node* yang optimal dalam *hidden layer*.

Meskipun banyak penelitian telah mengeksplorasi penggunaan JST untuk klasifikasi sinyal radar [7], masih terdapat kesenjangan dalam pemilihan jumlah *node* yang optimal dalam *hidden layer* [8]. Sebagian besar studi menggunakan pendekatan *trial-and-error* atau *heuristik* untuk menentukan jumlah *node* [9], yang bisa memakan waktu dan tidak selalu menghasilkan solusi yang optimal. Oleh karena itu, diperlukan metode optimasi yang lebih sistematis dan efisien.

Penelitian ini mengusulkan penggunaan *Particle Swarm Optimization* (PSO) untuk menentukan jumlah *node* optimal dalam *hidden layer* dari JST dalam tugas klasifikasi sinyal radar [10]. PSO adalah metode optimasi berbasis populasi yang terinspirasi oleh perilaku sosial kawanan burung atau ikan [11]. PSO bekerja dengan mengeksplorasi ruang solusi secara lebih luas dan menghindari jebakan solusi lokal, sehingga dapat menemukan jumlah *node* yang optimal dengan lebih efisien. Dalam pendekatan ini, setiap partikel dalam PSO merepresentasikan jumlah *node* dalam *hidden layer*. Proses optimasi melibatkan pembaruan posisi dan kecepatan partikel berdasarkan nilai personal *best* dan *global best*, yang ditentukan oleh performa JST pada dataset validasi.

Penggunaan PSO untuk menentukan jumlah *node* dalam *hidden layer* diharapkan dapat meningkatkan akurasi dan efisiensi klasifikasi sinyal radar. Dengan menemukan arsitektur JST yang optimal, kinerja sistem radar dapat ditingkatkan, sehingga mampu memberikan kontribusi signifikan dalam aplikasi praktis seperti pengawasan cuaca dan sistem pertahanan. Hasil penelitian ini juga memberikan wawasan teoretis yang berharga dalam bidang pembelajaran mesin dan optimasi.

2. Metodologi

Pada bagian ini dijelaskan secara rinci langkah-langkah yang digunakan dalam penelitian ini untuk mengoptimalkan arsitektur JST menggunakan PSO dalam tugas klasifikasi sinyal radar. Pertama, kami menguraikan dataset sinyal radar yang digunakan, termasuk langkah-langkah prapemrosesan data untuk memastikan kualitas dan konsistensi. Selanjutnya, dideskripsikan desain JST yang diusulkan, termasuk struktur lapisan dan fungsi aktivasi yang digunakan. Proses optimasi PSO dijelaskan dengan detail, mulai dari inisialisasi partikel hingga evaluasi *fitness* dan pembaruan posisi serta kecepatan partikel. Selain itu, kami menyajikan implementasi akhir dari JST yang dioptimasi, diikuti dengan evaluasi kinerja model menggunakan berbagai metrik performa. Pada bagian ini juga didiskusikan tentang kompleksitas dan waktu komputasi yang diperlukan oleh metode yang diusulkan serta perbandingan dengan metode *baseline* lainnya. Dengan metodologi yang komprehensif ini, kami bertujuan untuk menunjukkan efektivitas PSO dalam mengoptimalkan arsitektur JST untuk klasifikasi sinyal radar, serta implikasi praktisnya dalam aplikasi nyata.

2.1. Data yang Digunakan

Dataset sinyal radar yang digunakan dalam penelitian ini berasal dari *Repository* UCI *Machine Learning*, yang dapat di akses di *https://archive.ics.uci.edu/dataset/52/ionosphere*. Dataset ini terdiri dari sejumlah fitur yang diekstraksi dari sinyal radar yang dipantulkan dari *ionosfer*. Dataset tersebut mengandung 34 fitur yang merupakan pengukuran kontinu dari sinyal radar. Setiap sampel dalam dataset diklasifikasikan sebagai "*good*" jika sinyal menunjukkan adanya struktur di *ionosfer*, atau "*bad*" jika sinyal tidak menunjukkan adanya struktur. Secara keseluruhan, dataset terdiri dari 351 sampel.

Sebelum data digunakan untuk pelatihan model, beberapa langkah prapemrosesan dilakukan untuk memastikan kualitas dan konsistensi data. Pertama, fitur kontinu dinormalisasi ke dalam rentang [0,1] menggunakan persamaan berikut:

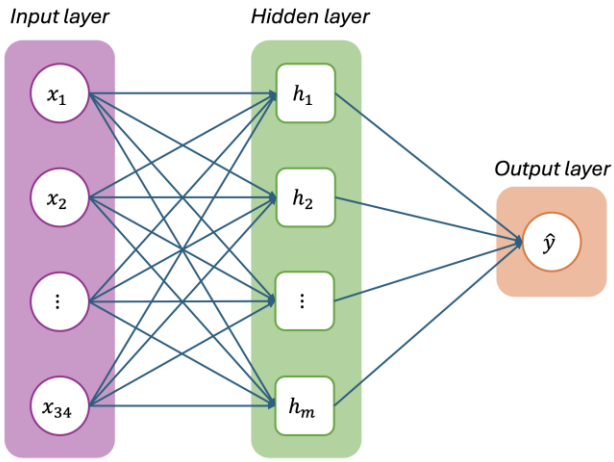
$$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

dengan x_i adalah data ke-i, x adalah vektor feature, dan x_i' adalah hasil normalisasi.

Untuk memastikan setiap fitur memiliki kontribusi yang sama dalam pelatihan model. Setelah normalisasi, dataset dibagi menjadi tiga set: 70% untuk pelatihan, 15% untuk validasi, dan 15% untuk pengujian. Pembagian tersebut dilakukan secara acak namun tetap menjaga proporsi kelas untuk memastikan representasi yang seimbang. Terakhir, label dikodekan sebagai 0 untuk "*bad*" dan 1 untuk "*good*" agar bisa digunakan dalam model klasifikasi biner. Langkah prapemrosesan ini memastikan bahwa data yang digunakan dalam pelatihan model memiliki kualitas yang baik dan siap untuk dianalisis lebih lanjut.

2.2. Desain Jaringan Syaraf Tiruan (JST)

Desain jaringan syaraf tiruan (JST) yang diusulkan terdiri dari *input layer*, satu *hidden layer*, dan *output layer*. Pada *input layer*, terdapat 34 *neuron* yang sesuai dengan jumlah fitur dalam dataset sinyal radar. Sebuah *hidden layer*, yang jumlah neuronnya akan dioptimasi menggunakan *Particle Swarm Optimization* (PSO), berfungsi sebagai lapisan tersembunyi yang mengolah data dari *input layer*. *Output layer* terdiri dari dua neuron yang digunakan untuk klasifikasi biner, yaitu mengklasifikasikan sinyal sebagai "*good*" atau "*bad*". Arsitektur jaringan syaraf tiruan yang diusulkan dapat dilihat pada Gambar 1.



Gambar 1. Arsitektur Jaringan Syaraf Tiruan yang Diusulkan

Untuk fungsi aktivasi, *hidden layer* menggunakan fungsi aktivasi *Rectified Linear Unit* (ReLU), yang didefinisikan sebagai $f(x) = \max(0, x)$ [12]. Fungsi ini memungkinkan jaringan untuk menangani non-linearitas dalam data dan telah tebukti efektif dalam melatih jaringan saraf tiruan yang dalam [13]. *Output layer* menggunakan fungsi aktivasi sigmoid, yang didefinisikan sebagai $\sigma(x) = \frac{1}{1+e^{-x}}$ [14]. Fungsi sigmoid cocok untuk klasifikasi biner karena menghasilkan output dalam rentang $[0, 1]$, yang dapat diinterpretasikan sebagai probabilitas kelas [15].

Secara matematis, JST ini dapat diformulasikan sebagai berikut. Misalkan $\mathbf{X} \in \mathbb{R}^{m\times 34}$ adalah matriks input dengan m sebagai jumlah sampel dan 34 sebagai jumlah fitur. Matriks bobot antara *input layer* dan *hidden layer* dilambangkan sebagai $\mathbf{W}_1 \in \mathbb{R}^{34\times h}$, di mana h adalah jumlah neuron dalam *hidden layer*, dan vektor bias untuk *hidden layer* adalah $\mathbf{b}_1 \in \mathbb{R}^{1\times h}$. Matriks bobot antara *hidden layer* dan *output layer* adalah $\mathbf{W}_2 \in \mathbb{R}^{h\times 2}$, dan vektor bias untuk *output layer* adalah $\mathbf{b}_2 \in \mathbb{R}^{1\times 2}$. Aktivasi dari *hidden layer* dilambangkan sebagai $\mathbf{H} \in \mathbb{R}^{m\times h}$, dan output dari jaringan dilambangkan sebagai $\hat{y} \in \mathbb{R}^{m\times 2}$.

Proses propagasi maju dilakukan dalam dua tahap. Pertama, input dipropagasikan ke *hidden layer* dengan persamaan

$$\mathbf{Z}_1 = \mathbf{X}\mathbf{W}_1 + \mathbf{b}_1$$

dan aktivasi dari *hidden layer* dihitung sebagai

$$\mathbf{H} = f(\mathbf{Z}_1) = \max(0, \mathbf{Z}_1)$$

Kedua, output dari *hidden layer* dipropagasikan ke *output layer* dengan persamaan

$$\mathbf{Z}_2 = \mathbf{H}\mathbf{W}_2 + \mathbf{b}_2$$

dan output jaringan dihitung sebagai

$$\hat{y} = \frac{1}{1 + e^{-\mathbf{Z}_2}}$$

Fungsi *loss* yang digunakan adalah *binary cross-entropy*, yang merupakan pilihan umum untuk tugas klasifikasi biner karena kemampuannya untuk mengukur perbedaan antara distribusi probabilitas yang diprediksi dan distribusi sebenarnya [16]. Fungsi *loss* tersebut didefinisikan sebagai

$$L = -\frac{1}{m}\sum_{i=1}^{m}(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i))$$

di mana y_i adalah label aktual dan $\hat{y}_i$ adalah prediksi model. Parameter $\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2,$ dan $\mathbf{b}_2$ dioptimasi menggunakan metode optimisasi *Adaptive Moment Estimation* (Adam) untuk meminimalkan fungsi *loss* tersebut.

2.3. *Particle Swarm Optimization* (PSO)

1. **Inisialisasi Partikel:** Setiap partikel merepresentasikan jumlah neuron dalam *hidden layer*, diinisialisasi dengan nilai acak dalam rentang yang telah ditentukan yaitu antara 1 hingga 128 [17].
2. **Evaluasi *Fitness*:** Untuk setiap partikel, bangun JST dengan jumlah neuron *N* yang diusulkan oleh partikel tersebut. Kemudian latih JST pada dataset pelatihan menggunakan metode Adam. Selanjutnya evaluasi kinerja JST pada dataset divalidasi menggunakan metrik performa *accuracy* atau *loss*. Nilai *fitness F* dari partikel adalah kinerja JST tersebut. Misalkan, untuk metrik akurasi:

   $$F = Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

3. **Pembaruan Kecepatan dan Posisi:** Kecepatan $v_i(t)$ dan posisi $x_i(t)$ partikel diperbarui menggunakan persamaan PSO yang diusulkan oleh Kennedy dan Eberhart [18] sebagai berikut:

   $$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1\big(p_i - x_i(t)\big) + \mathrm{c}_2 \cdot \mathrm{r}_2 \cdot \big(g - x_i(\mathrm{t})\big)$$

   dan

   $$x_i(t+1) = x_i(t) + v_i(t+1)$$

   di mana $v_i(t)$ adalah kecepatan partikel i pada iterasi t. $x_i(t)$ adalah posisi partikel i pada iterasi t. p_i adalah posisi personal *best* partikel i. g adalah posisi *global best*. w adalah faktor inersia. c_1 dan c_2 adalah koefisien akselerasi. r_1 dan r_2 adalah angka acak antara 0 dan 1.
4. **Penentuan Personal *Best* dan *Global best*:** Setiap partikel menyimpan nilai terbaik yang pernah dicapai (personal *best*) p_i. Seluruh partikel bersama-sama menentukan nilai terbaik global (*global best*) g.
5. **Iterasi:** Langkah-langkah evaluasi *fitness*, pembaruan kecepatan dan posisi, serta penentuan personal *best* dan *global best* diulang sampai kriteria konvergensi tercapai.

2.4. Implementasi Akhir

Setelah proses PSO selesai, jumlah neuron optimal dalam *hidden layer* N_{optimal} akan digunakan untuk membangun JST final. JST final kemudian dilatih ulang menggunakan seluruh dataset dengan *optimizer* standar.

2.5. Evaluasi Kinerja

JST final dievaluasi pada dataset uji menggunakan metrik performa seperti *accuracy*,

precision, *recall*, dan *F*1-*Score* [19]. Hasil kinerja JST dibandingkan dengan metode *baseline* yang meliputi *AdaBoost*, *Naive Bayes*, *k-NN*, *Decision Tree*, SVM, dan *Logistic Regression (LR)*, untuk menunjukkan efektivitas pendekatan yang diusulkan.

3. Hasil Dan Pembahasan

Bab ini menyajikan hasil dari eksperimen yang dilakukan serta analisis mendalam mengenai temuan yang diperoleh. Selain itu dibahas mengenai kinerja JST yang dioptimasi menggunakan PSO dalam klasifikasi sinyal radar. Hasil ini dibandingkan dengan berbagai metode *baseline* untuk menunjukkan efektivitas pendekatan yang diusulkan. Kami juga mengevaluasi jumlah neuron optimal dalam *hidden layer* yang ditemukan oleh PSO, serta dampaknya terhadap akurasi dan efisiensi model. Diskusi ini mencakup analisis kompleksitas dan waktu komputasi, serta *robustness* model terhadap variasi data melalui *k-Fold Cross-Validation*. Visualisasi hasil, seperti kurva *accuracy*, *confusion matrix*, dan ROC *curve*, disertakan untuk memberikan gambaran yang lebih jelas tentang kinerja model. Akhirnya, temuan ini dibandingkan dengan studi-studi sebelumnya untuk menegaskan kontribusi penelitian ini dalam literatur yang ada, serta membahas implikasi praktis dari hasi yang diperoleh.

3.1. Jumlah Neuron Optimal dalam *Hidden Layer*

Algoritma PSO berhasil mengidentifikasi 32 sebagai jumlah optimal neuron pada *hidden layer* yang memberikan kinerja model pada dataset validasi. Jumlah neuron ini kemudian digunakan untuk mengkonstruksi model JST final yang akan dievaluasi lebih lanjut.

3.2. Kinerja Model pada Dataset Uji

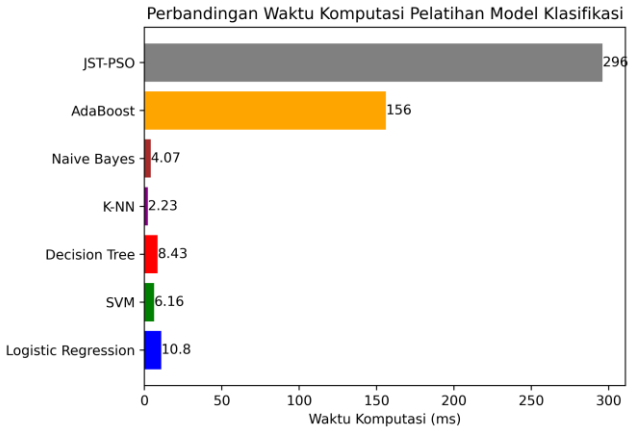
Kinerja JST dengan jumlah neuron optimal dievaluasi menggunakan metrik performa seperti *accuracy*, *precision*, *recall*, dan *F*1-*Score*. Hasil evaluasi kinerja JST dibandingkan dengan metode *baseline* yang meliputi *AdaBoost*, *Naive Bayes*, *k-NN*, *Decision Tree*, SVM, dan *Logistic Regression* seperti yang ditampilkan pada Tabel 1.

Tabel 1. Perbandingan Kinerja JST dengan Metode *Baseline* pada Dataset Uji

Models	*Accuracy*	*Precision*	*Recall*	*F*1-*Score*
JST-PSO	**0.9434**	**0.9167**	**1.0000**	**0.9565**
AdaBoost	0.9057	0.9118	0.9394	0.9254
Naive Bayes	0.8679	0.8421	0.9697	0.9014
k-NN	0.8679	0.8250	1.0000	0.9041
Decision Tree	0.9057	0.9118	0.9394	0.9254
SVM	0.8868	0.8649	0.9697	0.9143
LR	0.8679	0.8421	0.9697	0.9014

3.3. Kompleksitas dan Waktu Komputasi

Meskipun PSO memerlukan waktu komputasi yang lebih lama untuk menemukan jumlah neuron optimal, hasil yang diperoleh sepadan dengan peningkatan kinerja yang signifikan. Seperti ditunjukkan pada Gambar 2, waktu komputasi untuk JST yang dioptimasi menggunakan PSO adalah $296\,ms$, yang lebih lama dibandingkan dengan metode lain seperti *AdaBoost* ($156\,ms$), *Naive Bayes* ($4{,}07\,ms$), *k-NN* ($2{,}23\,ms$), *Decision Tree* ($8{,}43\,ms$), SVM ($6{,}16\,ms$), dan *Logistic Regression* ($10{,}8\,ms$).



Gambar 2. Perbandingan Waktu Komputasi JST dengan dan Tanpa PSO

Penggunaan metode optimasi yang lebih canggih seperti PSO memang membutuhkan sumber daya komputasi yang lebih besar, namun memberikan hasil yang lebih baik dalam hal akurasi dan kinerja secara keseluruhan. Hal ini menunjukkan bahwa investasi waktu dan sumber daya komputasi yang lebih besar pada tahap optimasi dapat memberikan hasil yang lebih baik dan lebih andal dalam aplikasi praktis.

3.4. *Robustness* Terhadap Variabilitas Data

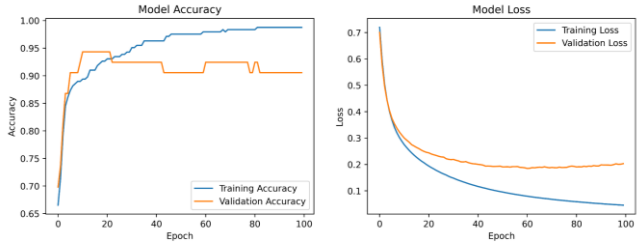
JST yang dioptimasi menggunakan PSO menunjukkan kinerja yang konsisten pada subset data yang berbeda, yang diukur menggunakan *k-Fold Cross-Validation*. Tabel 2 menunjukkan hasil 5-*Fold Cross-Validation* yang mengindikasikan bahwa model memiliki generalisasi yang baik dan tidak *overfitting* pada subset data tertentu.

Tabel 2. Hasil ***k*-*Fold Cross-Validation*** JST dengan PSO

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*
1	0.9155	0.8776	1.0000	0.9348
2	0.9286	0.9348	0.9556	0.9451
3	0.8429	0.8519	0.9388	0.8932
4	0.9143	0.8864	0.9750	0.9286
5	0.9429	0.9400	0.9792	0.9592

3.5. Visualisasi Hasil

Visualisasi hasil sangat penting untuk memahami kinerja model secara mendalam. Tiga visualisasi utama yang digunakan dalam penelitian ini adalah *confusion matrix*, ROC *curve*, dan kurva *accuracy* serta *loss* selama proses pelatihan dan validasi.



Gambar 3. Kurva *Accuracy* dan *Loss* Selama Pelatihan dan Validasi JST dengan PSO

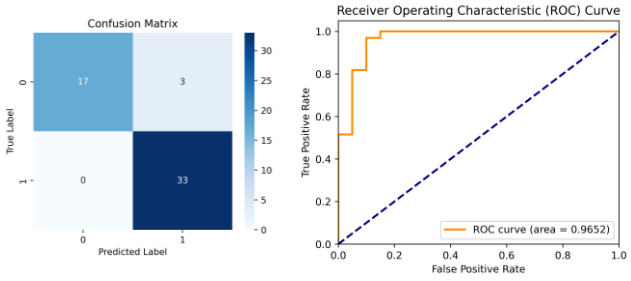
3.5.1. Kurva *Accuracy* dan *Loss*

Pada Gambar 3, kita dapat melihat perkembangan *accuracy* dan *loss* selama pelatihan dan validasi. Kurva *accuracy* menunjukkan peningkatan yang stabil, yang mengindikasikan bahwa model semakin baik dalam mengklasifikasikan data seiring bertambahnya *epoch*. Sementara itu, kurva *loss* yang menurun menunjukkan bahwa kesalahan prediksi model terus berkurang.

3.5.2. *Confusion Matrix*

Gambar 4 (a) menampilkan *Confusion matrix* dari hasil klasifikasi JST pada dataset uji. Dari *confusion matrix*, dapat dilihat bahwa jumlah prediksi benar (*true positives* dan *true negatives*) jauh lebih besar dibandingkan dengan jumlah prediksi salah (*false positives* dan *false negatives*).

Hal ini mengindikasikan bahwa model memiliki kinerja klasifikasi yang sangat baik.



(a) *Confusion matrix* (b) ROC *Curve*

Gambar 4. (a) *Confusion matrix* dan (b) ROC *Curve* untuk JST yang Dioptimalkan dengan PSO

3.5.3. ROC *Curve*

Gambar 4 (b) menunjukkan ROC *Curve* yang menggambarkan *trade-off* antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR). Area *Under the* ROC *Curve* (AUC-ROC) yang mendekati nilai 1 menunjukkan bahwa model memiliki kemampuan diskriminasi yang sangat baik antara kelas positif dan negatif. Hal ini juga tercermin dalam nilai AUC-ROC yang tinggi, menguatkan bahwa JST dengan PSO mampu menghasilkan prediksi yang akurat dan dapat diandalkan.

Visualisasi hasil ini memberikan gambaran komprehensif tentang kinerja JST yang dioptimasi dengan PSO. Kurva *accuracy* dan *loss* menunjukkan bahwa model belajar dengan baik tanpa mengalami *overfitting* yang signifikan. *Confusion matrix* memberikan bukti kuantitatif tentang jumlah prediksi benar dan salah, sementara ROC *Curve* mengonfirmasi kemampuan model untuk membedakan antara kelas yang berbeda dengan sangat baik. Secara keseluruhan, visualisasi ini mendukung temuan bahwa PSO adalah alat yang efektif untuk optimasi arsitektur JST, yang menghasilkan kinerja klasifikasi yang superior dalam aplikasi praktis seperti klasifikasi sinyal radar.

Tabel 3. Perbandingan Hasil Penelitian Ini dengan Studi Sebelumnya

Studies	*Models*	*Accuracy*	*Precision*	*Recall*	*F1-Score*
Pujari dan Jyoti [20]	Ensemble	0.91	0.88	0.96	0.89
Aung NW, dkk [21]	NB Tree	0.88	0.89	0.87	0.86
Jayshri, dkk [22]	SVM-PSO	0,82	0,82	0,78	0,82
Penelitian Ini	**JST-PSO**	**0.94**	**0.92**	**1.00**	**0.95**

3.6. Perbandingan dengan Studi Sebelumnya

Dibandingkan dengan studi-studi sebelumnya yang menggunakan dataset dan metode yang sama atau serupa, hasil penelitian ini menunjukkan peningkatan kinerja yang signifikan. Tabel 3 membandingkan hasil penelitian ini dengan beberapa studi sebelumnya, menegaskan kontribusi penelitian ini dalam literatur yang ada dan memberikan bukti tambahan bahwa PSO adalah alat yang efektif untuk optimasi arsitektur JST.

3.7. Implikasi Praktis

Hasil penelitian ini memiliki implikasi praktis yang signifikan dalam berbagai aplikasi klasifikasi sinyal radar. Peningkatan kinerja JST dalam mendeteksi dan mengklasifikasikan sinyal radar dapat meningkatkan keandalan dan efektivitas sistem radar dalam aplikasi nyata seperti pengawasan cuaca dan sistem pertahanan.

Secara keseluruhan, penelitian ini menunjukkan bahwa PSO adalah alat yang efektif

untuk optimasi arsitektur JST, khususnya dalam menentukan jumlah neuron optimal dalam *hidden layer*. Penerapan metode ini dapat diterapkan pada berbagai domain lain yang memerlukan optimasi arsitektur JST untuk tugas-tugas klasifikasi yang kompleks.

3.8. Penelitian Lebih Lanjut

Untuk penelitian di masa depan, beberapa area dapat dieksplorasi lebih lanjut untuk memperluas dan meningkatkan temuan penelitian ini. Pertama, pengujian metode optimasi lain seperti *Genetic Algorithm* atau *Differential Evolution* untuk membandingkan kinerja dan efisiensi dengan PSO dapat memberikan wawasan tambahan. Selain itu, penerapan PSO dalam optimasi arsitektur jaringan yang lebih kompleks, seperti *Convolutional Neural Networks* (CNN) atau *Recurrent Neural Networks* (RNN), bisa menjadi langkah berikutnya yang menarik.

Selain itu, integrasi teknik regularisasi lanjutan seperti *Dropout* atau *Batch Normalization* dalam JST yang dioptimasi oleh PSO dapat membantu lebih lanjut dalam mengurangi *overfitting* dan meningkatkan generalisasi model. Eksplorasi terhadap penggunaan berbagai fungsi aktivasi dan struktur lapisan yang berbeda juga dapat memberikan pemahaman yang lebih dalam tentang bagaimana berbagai konfigurasi arsitektur JST mempengaruhi kinerja model.

Penerapan metode yang diusulkan pada dataset yang lebih besar dan lebih beragam juga akan membantu untuk menguji skalabilitas dan keandalan pendekatan ini dalam kondisi yang lebih menantang. Terakhir, mengembangkan pendekatan ini menjadi *framework* yang dapat digunakan oleh komunitas peneliti dan praktisi dapat mempercepat adopsi dan pengembangan lebih lanjut dari metode ini.

Dengan berbagai arah penelitian di masa depan ini, diharapkan pendekatan PSO dalam optimasi JST akan terus berkembang dan memberikan kontribusi yang signifikan dalam bidang pembelajaran mesin dan aplikasinya dalam berbagai domain.

4. Kesimpulan Dan Saran

Penelitian ini berhasil menunjukkan bahwa penggunaan PSO untuk menentukan jumlah neuron optimal dalam *hidden layer* dari JST memberikan peningkatan kinerja yang signifikan dalam klasifikasi sinyal radar dibandingkan dengan metode *baseline* tradisional seperti *Logistic Regression*, *Support Vector Machine* (SVM), *Decision Tree*, *Random Forest*, *k-Nearest Neighbors* (*k*-NN), *Naive Bayes*, dan *Multi-Layer Perceptron* (MLP).

PSO terbukti efektif dalam menemukan arsitektur JST yang optimal, mengatasi keterbatasan metode optimasi tradisional dengan mengeksplorasi ruang solusi yang lebih luas dan menghindari jebakan solusi lokal. Hasil penelitian menunjukkan bahwa JST yang dioptimasi menggunakan PSO memiliki performa superior dalam hal *accuracy*, *precision*, *recall*, *F*1-*Score*, dan AUC-ROC, serta menunjukkan *robustness* terhadap variasi data melalui evaluasi *k-Fold Cross-Validation*.

Meskipun PSO memerlukan waktu komputasi

yang lebih lama untuk menemukan jumlah neuron optimal, hasil yang diperoleh sepadan dengan peningkatan kinerja yang signifikan. Analisis waktu komputasi menunjukkan bahwa penggunaan metode optimasi yang lebih canggih dapat memberikan hasil yang lebih baik meskipun memerlukan sumber daya komputasi yang lebih besar.

Penelitian ini juga mengindikasikan bahwa JST yang dioptimasi menggunakan PSO memiliki kinerja yang konsisten dan tidak *overfitting* pada subset data tertentu, menunjukkan generalisasi yang baik. Perbandingan dengan studi-studi sebelumnya yang menggunakan dataset dan metode serupa menunjukkan bahwa penelitian ini memberikan kontribusi yang signifikan dalam literatur yang ada dan memberikan bukti tambahan bahwa PSO adalah alat yang efektif untuk optimasi arsitektur JST.

Secara keseluruhan, penelitian ini menunjukkan bahwa PSO adalah alat yang efektif untuk optimasi arsitektur JST, khususnya dalam menentukan jumlah neuron optimal dalam *hidden layer*. Penerapan metode ini dapat diperluas ke berbagai domain lain yang memerlukan optimasi arsitektur jaringan syaraf tiruan untuk tugas-tugas klasifikasi yang kompleks.

5. Ucapan Terima Kasih

Ucapan terimakasih penulis sampaikan kepada keluarga, dan kepada seluruh pihak yang turut membantu, membimbing dan mendukung pelaksanaan kegiatan penelitian ini.

Daftar Pustaka

[1] J. A. Zhang *et al.*, "An Overview of Signal Processing Techniques for Joint Communication and Radar Sensing," *IEEE J Sel Top Signal Process*, vol. 15, no. 6, pp. 1295–1315, 2021, doi: 10.1109/JSTSP.2021.3113120.

[2] Z. Seddighi, M. R. Ahmadzadeh, and M. R. Taban, "Radar signals classification using energy-time-frequency distribution features," *IET Radar, Sonar & Navigation*, vol. 14, no. 5, pp. 707–715, 2020.

[3] R. Y. Choi, A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang, and J. P. Campbell, "Introduction to machine learning, neural networks, and deep learning," *Transl Vis Sci Technol*, vol. 9, no. 2, p. 14, 2020.

[4] L. Wang, J. Tang, and Q. Liao, "A study on radar target detection based on deep neural networks," *IEEE Sens Lett*, vol. 3, no. 3, pp. 1–4, 2019.

[5] C. C. Aggarwal, "Machine Learning with Shallow Neural Networks," in *Neural Networks and Deep Learning: A Textbook*, Cham: Springer International Publishing, 2018, pp. 53–104. doi: 10.1007/978-3-319-94463-0_2.

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[7] Z. Geng, H. Yan, J. Zhang, and D. Zhu, "Deep-learning for radar: A survey," *IEEE Access*, vol. 9, pp. 141800–141818, 2021.

[8] W. Jiang, Y. Ren, Y. Liu, and J. Leng, "Artificial neural networks and deep learning techniques applied to radar target detection: A review," *Electronics (Basel)*, vol. 11, no. 1, p. 156, 2022.

[9] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization.," *Journal of machine learning research*, vol. 13, no. 2, 2012.

[10] Y. Zhang, S. Wang, and G. Ji, "A comprehensive survey on particle swarm optimization algorithm and its applications," *Math Probl Eng*, vol. 2015, no. 1, p. 931256, 2015.

[11] M.-P. Song and G.-C. Gu, "Research on particle swarm optimization: a review," in *Proceedings of 2004 international conference*

on machine learning and cybernetics (IEEE Cat. No. 04EX826), 2004, pp. 2236–2241.

[12] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[13] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[14] J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *International workshop on artificial neural networks*, 1995, pp. 195–201.

[15] C. M. Bishop, "Pattern recognition and machine learning," *Springer google schola*, vol. 2, pp. 1122–1128, 2006.

[16] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[17] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[18] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, 1995, pp. 1942–1948.

[19] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation." 2020.

[20] P. Pujari and J. Gupta, "Improving classification accuracy by using feature selection and ensemble model," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, no. 2, pp. 380–386, 2012.

[21] A. N. Oo, "Classification of Radar Returns from Ionosphere Using NB-Tree and CFS," *International Journal of Trend in Scientific Research and Development (IJTSRD)*, vol. 2, no. 5, pp. 1640–1642, 2018.

[22] J. D. Dhande and D. R. Dandekar, "PSO Based SVM as an Optimal Classifier for Classification of Radar Returns from Ionosphere," *International Journal of Emerging Technology and Advanced Engineering*, vol. 21, 2011.