AITI: Jurnal Teknologi Informasi, Volume 20 No. 2 Agustus 2023, 125-134
ISSN 1693-8348 E-ISSN 2615-7128

JURNAL
TEKNOLOGI INFORMASI

Neuroevolution untuk optimalisasi parameter jaringan saraf tiruan

Hindriyanto Dwi Purnomo[1)], Tad Gonsalves[2)], Teguh Wahyono[3)],
Pratyaksa Ocsa Nugraha Saian[4)]
[1,3,4)]Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana
Jl. Dr. O. Notohamidjojo No 1-10, Salatiga, Indonesia
[2)]Dept. of Information and Communication Sciences, Sophia University, Japan
7-1 Kioi-cho, Chiyoda City, Tokyo, Japan
Email : hindriyanto.purnomo@uksw.edu

Riwayat artikel:
Received: 18-20-2022 Revised: 08-11-2022 Accepted: 11-11-2022

Abstract

Artificial Neural Network is a supervised learning method for various classification problems. Artificial Neural Network uses training data to identify patterns in the data; therefore, training phase is crucial. During this stage, the network weight is adjusted so that they can recognize patterns in the data. In this research, a neuroevolution approach is proposed to optimize artificial neural network parameters (weight) Neuroevolution is a combination of evolutionary algorithms, including various metaheuristics algorithms, to optimize neural network parameters and configuration. In particular, this research implemented particle swarm optimization as the artificial neural network optimizer. The performance of the proposed model was compared to backpropagation, which uses gradient information to adjust the neural network parameter. There are five datasets used as the benchmark problems. The datasets are iris, wine, breast cancer, ecoli, and wheat seeds. The experiment results show that the proposed method has better accuracy than the backpropagation in three out of five problems and has the same accuracy in two problems. The proposed method is also faster than the backpropagation method in all problems. These results reveal that neuroevolution is a promising approach to improving the performance of artificial neural networks. Further studies are needed to explore more benefits of this approach.

Keywords: *Neuroevolution, Particle swarm optimization, Neural Network, Tuning*

Abstrak

Jaringan saraf tiruan merupakan metode *supervised learning* yang telah diterapkan untuk menyelesaikan berbagai permasalahan klasifikasi. Sebagai metode *supervised learning*, jaringan saraf tiruan memerlukan data training untuk mengidentifikasi pola dalam data sehingga fase *learning* menjadi penting. Pada fase *learning*, konfigurasi bobot pada jaringan saraf tiruan diatur sehingga jaringan saraf tiruan tersebut bisa mengenali pola di dalam data. Pada penelitian ini diusulkan metode untuk mengoptimalkan nilai bobot pada konfigurasi jaringan saraf tiruan menggunakan pendekatan *neuroevolution*. *Neuroevolution* adalah pengintegrasian metode *evolutionary algorithm*; termasuk di

dalamnya adalah berbagai metode metaheuristik; dengan jaringan saraf tiruan. Secara khusus, penelitian ini menggunakan metode *particle swarm optimization* untuk mengoptimalkan bobot pada jaringan saraf tiruan. Kinerja model yang diusulkan dibandingkan dengan metode *backpropagation* dengan *stochastic gradient descent* menggunakan lima dataset: iris, *wine*, *breast cancer*, *ecoli*, dan *wheat seeds*. Hasil eksperimen menunjukkan bahwa model yang diusulkan memiliki akurasi yang lebih baik di tiga dataset dari lima dataset dan memiliki kinerja yang sama di dua dataset. Hasil penelitian ini mengindikasikan bahwa pendekatan *neuroevolution* memiliki potensi sebagai metode optimalisasi parameter pada jaringan saraf tiruan. Penelitian ini bisa dikembangkan dengan mengidentifikasi karakteristik konvergensi dari pendekatan *neuroevolution* maupun menerapkan berbagai metode *evolutionary algorithm* untuk mengoptimalkan nilai bobot pada jaringan saraf tiruan.

Kata kunci: *neuroevolution, particle swarm optimization*, jaringan saraf tiruan, *tuning*

Pendahuluan

Jaringan saraf tiruan merupakan metode klasifikasi yang mengimitasi cara kerja sel saraf. jaringan ini memiliki beberapa lapisan sel saraf, yaitu lapisan input, lapisan tersembunyi dan lapisan *output*. Jaringan saraf tiruan banyak dipergunakan untuk menyelesaikan berbagai permasalahan klasifikasi [1], [2]. Jaringan saraf tiruan merupakan metode *supervised learning* yang memerlukan pembelajaran untuk bisa mengenali pola yang terdapat di dalam dataset. Metode *learning* yang banyak dipergunakan adalah metode *backpropagation*. Metode *backpropagation* memanfaatkan *gradient* dari fungsi *error* (*loss function*) untuk meminimalkan *error* dengan cara melakukan penyesuaian nilai bobot pada koneksi antar *node (neuron)* di dalam jaringan saraf tiruan [3].

Fase *training* pada jaringan saraf tiruan merupakan proses yang kompleks dan panjang karena distribusi lapisan *input* berubah oleh perubahan parameter input dari lapisan sebelumnya [4]–[6]. Fase *training* merupakan fase yang sangat krusial dalam jaringan saraf tiruan karena fase ini yang akan menghasilkan konfigurasi bobot antar sel saraf. Jika proses *training* berjalan dengan baik, maka akan didapatkan konfigurasi bobot yang baik, sehingga performa jaringan saraf tiruan akan tinggi. Sebaliknya, jika proses *training* yang kurang baik, maka akan menyebabkan performa jaringan saraf tiruan menjadi tidak baik. Ada dua faktor penting yang mempengaruhi keberhasilan proses *training*, yaitu dataset dan metode *learning*. Jumlah data yang terlalu banyak dan proses *training* yang berlebihan akan menyebabkan terjadinya *overfitting*, di mana model yang dihasilkan akan merepresentasikan setiap data *training* tetapi kurang mencerminkan generalisasi dari data *training*. Sebaliknya, jika jumlah data terlalu sedikit dan proses *training* tidak banyak akan menyebabkan *underfitting*, di mana model tidak dapat merepresentasikan informasi penting di dalam data *training*.

Implementasi metode berbasis jaringan saraf tiruan yang semakin masif menghadirkan berbagai tantangan baru. Volume data yang terus meningkat

memerlukan sumber daya yang besar untuk mengolahnya. Hal ini menyebabkan metode *training* yang cepat dan efisien menjadi sangat penting dalam pengembangan dan penerapan jaringan saraf tiruan. Salah satu metode yang dipergunakan secara luas adalah metode *backpropagation* yang menggunakan metode *learning stochastic gradient descent* (SGD). Metode ini memiliki keterbatasan antara lain; sulitnya untuk mencapai konvergensi ketika jumlah lapisan dalam jaringan saraf tiruan cukup banyak [7], proses *training* yang lama, serta resiko terjadinya *vanishing gradient*. Hal tersebut akan mempengaruhi performa dari jaringan saraf tiruan secara *keseluruhan*. Ada beberapa pengembangan metode SGD yang sudah dilakukan, antara lain metode Adagrad [8], *Root Mean Square Propagation*, RMSProp [9], dan *adaptive momentum estimation*, Adam [10]. Meskipun pengembangan ini mampu meningkatkan SGD sampai pada batas tertentu, proses *training* pada jaringan saraf tiruan masih menjadi sebuah tantangan. Oleh karena itu, para peneliti berusaha mencari terobosan baru untuk mengatasi permasalahan tersebut. Salah satu pendekatan yang dilakukan adalah dengan *neuroevolution*. *Neuroevolution* adalah sebuah pendekatan yang mengkombinasikan antara *evolutionary algorithm* dengan jaringan saraf tiruan. *Neuroevolution* menjadi salah satu alternatif yang menjanjikan untuk melakukan *training* pada jaringan saraf tiruan [11].

Evolutionary algorithm merupakan metode optimalisasi yang terinspirasi dari proses evolusi makhluk hidup. Metode ini didesain untuk menemukan nilai optimal global atau *local* dalam waktu yang cepat (*acceptable search time*) dengan sumber daya komputasi yang kecil atau *reasonable computational cost* [5]. Secara umum, yang termasuk dalam metode *evolutionary algorithm* ini adalah berbagai metode *metaheuristik*, yang dapat dipakai untuk permasalahan diskrit maupun kontinyu. Beberapa metode *metaheurisitk* yang popular antara lain Algoritma Genetik (AG), *Particle Swarm Optimization* (PSO) dan *Ant Colony Optimization* (ACO). Metode *metaheuristik* merupakan metode yang handal untuk mengatasi berbagai permasalahan optimalisasi. Metaheuristik sudah banyak diterapkan pada berbagai permasalahan optimalisasai, antara lain untuk penjadwalan [12], *vehicle routing problem* [13] dan optimalisasi *assembly line* [14].

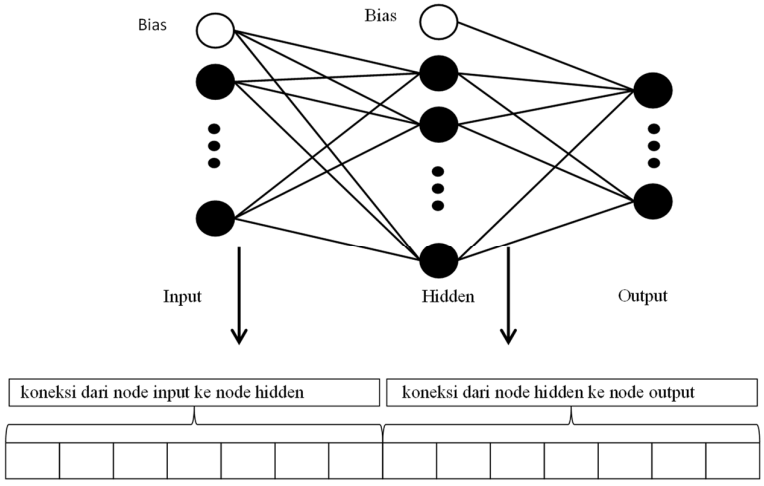
Integrasi *evolutionary algorithm* dan *deep neural network*, yang sering disebut dengan *neuroevolution*, semakin menarik para peneliti karena berbagai penelitian awal yang mengindikasikan potensi *neuroevolution* yang sangat menjanjikan. Beberapa penelitian yang sudah dilakukan terkait integrasi antara *evolutionary algorithm* dan *artificial neural network* antara lain Leung, dkk [15] mengembangkan modifikasi algoritma genetik untuk melakukan *tuning* struktur dan parameter jaringan saraf tiruan. Juang [16] menggabungkan PSO dan AG untuk melakukan *training recurrent neural network* dan *fuzzy neural network*. Rere [17] menerapkan *simulated annealing* untuk mempercepat proses *training* pada *deep learning*. *Neuroevolution* memiliki kemampuan untuk bisa mengoptimalkan

parameter serta mempelajari *building blok* solusi dan hiper parameter. Hal inilah yang kemudian menyebabkan *neuroevolution* lebih banyak dipergunakan untuk mengoptimalisasi bobot *neural network* [18], mengoptimalkan konfigurasi *neural network* [9], [19] dan mengoptimalkan *hyperparameter* [20].

Hasil-hasil penelitian awal terkait *neuroevolution* ini tentunya perlu untuk terus dilakukan, sehingga potensi *neuroevolution* bisa lebih dieksplorasi dan dimanfaatkan. Dalam penelitian ini dilakukan eksplorasi *neuroevolution* untuk mengoptimalkan parameter pada *deep neural network*. Secara khusus, penelitian difokuskan pada penerapan *particle swarm optimisation* untuk mengoptimalkan parameter bobot pada jaringan saraf tiruan. Performa model yang dihasilkan kemudian diuji pada permasalahan klasifikasi menggunakan lima dataset yaitu dataset *iris, wine*, *breast cancer*, *pima-indiana-diabetes* dan *wheat-seeds* [21].

Metode Penelitian

Model yang diusulkan dalam penelitian berfokus pada optimalisasi paramater bobot (*weight*) jaringan saraf tiruan. Oleh karena itu, konfigurasi jaringan saraf tiruan ditentukan di awal, yaitu *neural network* yang terdiri dari lapisan *input*, satu lapisan tersembunyi, dan lapisan *output*. Jumlah *node* di lapisan input disesuaikan dengan jumlah fitur dari permasalahan yang akan diselesaikan dan *node bias*. Jumlah *node* di lapisan tersembunyi dipengaruhi oleh jumlah *node* di *layer input* sedangkan jumlah *node* di lapisan *output* disesuaikan dengan jumlah kelas target dari permasalahan yang akan diselesaikan.



Gambar 1 Pengkodean solusi

Optimalisasi bobot jaringan saraf tiruan dalam penelitian ini menggunakan metode *particle swarm optimization* (PSO). Di dalam PSO, sebuah solusi dinyatakan dengan sebuah partikel yang tersusun atas rangkaian bobot koneksi antar *node* dalam konfigurasi jaringan saraf tiruan. Konfigurasi bobot di dalam jaringan saraf tiruan diratakan (*flatten*) menjadi sebuah vektor solusi atau partikel.

Pengkodean solusi ini dapat diilustrasikan pada Gambar 1. Dalam konfigurasi jaringan saraf tiruan ini setiap *node* di lapisan *input* akan terhubung ke semua *node* di lapisan tersembunyi (kecuali dengan bias di lapisan tersembunyi) dan setiap *node* di lapisan tersembunyi akan terhubung dengan *node* di lapisan *output*. Jumlah koneksi antar *node* ini akan menjadi panjang vektor solusi dan dirumuskan dengan Persamaan (1).

$$D = (n_i + 1)n_h + (n_h + 1)n_o \quad (1)$$

di mana :

D	:	panjang vektor solusi/dimensi permasalahan
n_i	:	jumlah *node* dilapisan *input*
n_h	:	jumlah *node* di lapisan tersembunyi
n_o	:	jumlah *node* di lapisan *output*

Dalam PSO, pencarian solusi menggunakan analogi pergerakan partikel di dalam ruang solusi. Posisi partikel dinyatakan dengan Persamaan (2) dan (3) [22].

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

$$v_{id}(t+1) = i_w v_{id}(t) + c_1 r_1\big(p_{id} - x_{id}(t)\big) + c_2 r_2\big(g_d - x_{id}(t)\big) \quad (3)$$

di mana:

t	:	waktu
x_{id}	:	posisi partikel i pada dimensi ke- d
$v_{id,t}$	:	kecepatan partikel i pada dimensi ke- d
i_w	:	bobot inersia
c_1	:	koefisisen kognitif
c_2	:	koefisien sosial
p_{id}	:	posisi terbaik partikel i pada dimensi ke- d
g_d	:	posisi pertikel terbaik (*global best*) pada dimensi ke- d
r_1, r_2	:	nilai *random*

Pada saat mencari solusi terbaik, kumpulan partikel ini akan terus bergerak dalam kelompoknya (*swarm*). Mekanisme pergerakan partikel-partikel dalam mencari solusi terbaik dapat dideskripsikan pada Kode Program 1.

Kode Program 1 *Pseudocode* PSO

```
Inisialisasi_posisi_partikel ()
evaluasi_partikel()
menentukan_posisi_terbaik_partikel_i
menentukan_posisi_global_best

while solusi belum ditemukan :
   for i = 1 sampai jumlah_partikel
        update_kecepatan_partikel_i ()
        update_posisi_partikel_i ()
        evaluasi_partikel_i()
        update_posisi_terbaik_partikel_i ()
        update_posisi_global_best ()
   endFor
end
```

Ketika sebuah partikel mendapatkan posisi yang baru, maka kualitas partikel diukur dengan menerapkan nilai vektor solusi pada jaringan saraf tiruan. Nilai vektor solusi tersebut dipetakan kembali ke dalam konfigurasi jaringan saraf tiruan sebagai bobot koneksi antar *node*. Setelah itu, dilakukan proses *feedforward* untuk mendapatkan nilai *output* dari *deep neural network*. *Feedforward* dilakukan dengan memasukkan dataset ke dalam konfigurasi jaringan saraf tiruan dan dihitung perambatannya mulai dari lapisan *input* sampai lapisan *output*. Setiap *node* di lapisan tersembunyi dan lapisan *output* memiliki dua fungsi, yaitu fungsi agregasi dan fungsi aktivasi. Fungsi agregasi bertujuan untuk menghitung semua data *input* yang masuk pada suatu *node*, sedangkan fungsi aktivasi adalah fungsi yang memetakan hasil fungsi agregasi ke nilai luaran dari *node* tersebut. Fungsi agregasi didapatkan dengan menjumlah hasil perkalian antara nilai *input* dengan bobotnya. Fungsi agregasi untuk sebuah *node* di lapisan tersembunyi dinyatakan dengan Persamaan (4).

$$S_h = W_{ih} I \; + \; B_{ih} \tag{4}$$

Hasil dari fungsi agregasi ini kemudian dipetakan ke nilai *output* oleh fungsi aktivasi. Fungsi aktivasi dari lapisan tersembunyi menggunakan fungsi *sigmoid*, yang dirumuskan dengan Persamaan (5).

$$Z_h(S_h) = \frac{1}{1 + e^{-S_h}} \tag{5}$$

Nilai aktivasi dari lapisan tersembunyi akan menjadi nilai input bagi lapisan *output* dan dirumuskan dengan Persamaan (6).

$$S_o = W_{ho} Z_h + \; B_{ho} \tag{6}$$

Fungsi aktivasi untuk lapisan *output* adalah fungsi *softmax* karena *output*-nya menggunakan *one hot encoding*. Fungsi *softmax* diformulasikan dengan Persamaan (7).

$$\sigma(S_o) = \frac{e^{S_o}}{\sum_{k=1}^{K} e^{S_o}} \tag{7}$$

di mana :

S_h	:	fungsi agregasi di lapisan tersembunyi
S_o	:	fungsi agregasi di lapisan *output*
I	:	vektor *input*
W_{ih}	:	matrik bobot antara lapisan *input* dengan lapisan tersembunyi
W_{ho}	:	matrik bobot antara lapisan tersembunyi dengan lapisan *output*
B_{ih}	:	vektor bias *input*
$B_i o$	:	vektor bias lapisan tersembunyi
Z_h	:	fungsi aktivasi di *layer* tersembunyi
σ	:	vektor luaran di *layer output*
K	:	jumlah kelas *output*

Karena bobot pada jaringan saraf tiruan didapatkan dari partikel dalam PSO, maka nilai W_{ih} dan W_{ho} didapatkan dari nilai *x* pada PSO sesuai dengan pemetaannya.

Hasil dan Pembahasan

Performa model yang diusulkan dalam penelitian ini dibandingkan dengan metode *backpropagation stochastic gradient descent* (SGD), menggunakan lima *benchmark problems* yang didapatkan dari UCI *machine learning repository* [11]. Metode Kelima *benchmark problems* tersebut adalah: *iris, wine, breast_cancer, ecoli*, dan *wheat_seeds*. Metadata terkait dengan dataset yang dipergunakan dalam penelitian ini ditunjukkan pada Tabel 1.

Tabel 1 Dataset

No.	Dataset	Tipe kelas	Fitur	#kelas	#sample
1	Iris	*multiclass*	4	2	150
2	Wine	*multiclass*	13	3	178
3	Breast cancer	*binary class*	30	2	569
4	Ecoli	*multiclass*	7	7	336
5	Wheat seeds	*multiclass*	7	3	210

Konfigurasi jaringan saraf tiruan yang dipergunakan dalam penelitian ini terdiri dari satu lapisan *input*, satu lapisan tersembunyi, dan satu lapisan *output*. Jumlah *node* di lapisan *input* disesuaikan dengan jumlah variabel *input* ditambah dengan bias. Jumlah *node* di lapisan tersembunyi adalah tiga kali jumlah *node* di lapisan *input* dan node bias. Hal ini didasarkan pada uji empiris yang dilakukan, di mana jumlah tersebut sudah cukup memadai dan mendapatkan hasil yang baik. Sedangkan jumlah *node* di lapisan *output* sesuai dengan jumlah kelas *output*. Untuk setiap *benchmark problem*, model dijalankan 10 kali, dan setiap kali dijalankan diukur akurasinya. Supaya perbandingan yang dilakukan setara, maka jumlah *epoch* satu *benchmark problem* dibuat sama untuk setiap model. Hasil eksperimen ini kemudian dihitung nilai rata-rata, nilai terbaik, dan standar deviasinya. Untuk melihat signifikasi perbedaan hasil kedua model, dilakukan uji t-test. Kedua model diimplementasikan menggunakan Python dan dijalankan pada komputer dengan spesifikasi AMD Ryzen Threadripper 3970X 32-core processor, 3.69 GHz, RAM 32 GB, and GPU RTX 3090. Akurasi untuk setiap *benchmark problem* berdasarkan eksperimen diperlihatkan pada Tabel 2.

Tabel 2 Akurasi

Problems	**epoch**	**SGD**			**PSO**			**t-test**
		best	**mean**	**stdev**	**best**	**mean**	**stdev**	
iris	10k	0.967	0.957	0.016	1.000	**0.993**	0.014	+
wine	50k	0.472	0.372	0.057	0.917	**0.742**	0.099	+
breast_cancer	10k	0.711	0.535	0.148	0.930	**0.891**	0.022	+
ecoli	25k	0.912	**0.837**	0.038	0.912	0.829	0.071	*
wheat_seeds	25k	0.952	0.900	0.049	0.952	**0.902**	0.044	*

Berdasarkan hasil percobaan yang dilakukan terlihat bahwa optimalisasi parameter jaringan saraf tiruan menggunakan PSO memiliki rata-rata akurasi yang lebih baik dari pada metode *backpropagation-SGD* di 4 dari 5 *benchmark problem* serta memiliki akurasi yang lebih buruk di 1 *benchmark problem*. Berdasarkan uji signifikansi, metode *tuning parameter* menggunakan PSO lebih baik dari pada metode *backpropagation* SGD di tiga dari lima *benchmark problem* serta memiliki akurasi yang kurang lebih sama di dua *benchmark problem*. Untuk dataset *wine* dan *breast-cancer*, selisih nilai rata-rata antara metode SGD dan PSO relatif tinggi dibandingkan ketiga dataset lainnya. Kedua dataset tersebut memiliki jumlah fitur yang lebih tinggi dibandingkan dataset lainnya, sehingga relasi antara atribut bebas dan atribut terikatnya menjadi lebih kompleks. Relasi yang kompleks ini menyebabkan metode SGD, yang cenderung bersifat *local optimizer*, kesulitan dalam mengoptimalkan parameter yang ada di jaringan saraf tiruan. Sementara *particle swarm optimization*, yang bersifat *global optimizer*, mampu mengenali struktur jaringan saraf tiruan secara menyeluruh, sehingga mampu memberikan performa yang lebih baik.

Hasil eksperimen yang didapatkan menunjukkan bahwa metode *tuning* parameter menggunakan PSO secara umum memiliki kemampuan yang lebih baik dari pada *backpropagation*. Hal ini mengindikasikan bahwa pendekatan dengan *neuroevolution* memiliki potensi yang besar dalam melakukan *tuning parameter* jaringan saraf tiruan. Tentunya penelitian lanjutan sangat diperlukan untuk mengeksplorasi kemampuan berbagai metode *evolutionary algorithm* seperti; algoritma genetik dan *differential evolution*; dalam meningkatkan performa jaringan saraf tiruan.

Simpulan

Jaringan saraf tiruan merupakan salah satu metode klasifikasi yang banyak diimplementasikan untuk menangani berbagai permasalahan. Salah satu fase krusial dalam jaringan saraf tiruan adalah fase *tuning parameter*. Pendekatan *backpropagation* dengan memanfaatkan *gradient* yang selama ini dipergunakan memiliki berbagai keterbatasan, sehingga diperlukan terobosan baru untuk meningkatkan proses *tuning parameter* pada jaringan saraf tiruan. Salah satu pendekatan yang dilakukan adalah dengan *neuroevolution*, yang mengkombinasikan antara metode *evolutionary algorithm* dengan *artificial neural network*. Dalam penelitian ini dilakukan pemodelan *tuning parameter* jaringan saraf tiruan menggunakan metode *particle swarm optimization*. Model yang diusulkan dibandingkan dengan metode *backpropagation*-SGD dan diterapkan pada lima *benchmark problems*.

Hasil penelitian menunjukkan bahwa *tuning parameter* jaringan saraf tiruan menggunakan metode *particle swarm optimization* memiliki akurasi yang lebih baik di tiga dari lima *benchmark problems* serta memiliki akurasi yang setara

dengan *backpropagation* di dua *benchmark problems*. Hal ini mengindikasikan bahwa pendekatan *neuroevolution* memiliki potensi yang besar dalam meningkatkan performa jaringan saraf tiruan. Penelitian lanjutan sangat diperlukan untuk mengeksplorasi berbagai kemungkinan penerapan *evolutionary algorithm* untuk meningkatkan performa *deep neural network*, seperti mengimplementasikan metode algoritma genetik dan *differential evolution* untuk *tuning parameter* pada jaringan saraf tiruan dengan banyak lapisan (*deep neural network*) maupun untuk *tuning hyper-parameter*.

Acknowledgements

Penelitian ini didanai oleh Hibah Riset UKSW tahun 2022 no 190/Pen./Rek./6/V/2022.

Daftar Pustaka

[1] O. Kwon, et al., "A deep neural network for classification of melt-pool images in metal additive manufacturing", *Journal of Intelligent Manufacturing*, vol 31, pp. 375-389, 2020

[2] H. H. Sultan, et al., "Multi-classification of Brain Tumor Images using Deep Neural Network", *IEEE Access*, vol 7, pp. 69215-69225, May 2019

[3] Hecht-Nielsen, "Theory of the backpropagation neural network", *International Joint Conference on Neural Network (IJCNN)*, pp. 598-605, 1989

[4] S. Ioeffe and C. Szegedy., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", https://arxiv.org/pdf/1502.03167.pdf, diakses 20 Juli 2022.

[5] S. Fong, et al., "How Metaheuristics Algorithm Contribute to Deep Learning in the Hype of Big Data Analytics", in *Proceeding in Intelligent Computing Techniques: Theory, Practice and Applications, Advances in Intelligent System and Computing*, vol 518, pp. 3-25, 2017

[6] J. Schmidhuber., "Deep learning in neural networks: An overview", Neural Networks, vol 61, pp. 85-117, Jan 2015

[7] Q. Meng, et al., "Convergence analysis of distributued stochastic gradient descent with shuffling", *Neurocomputing*, vol 337, pp. 46-57, April 2019

[8] J. Duchi, et al., "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization", *Journal of Machine Learning Research*, vol 12, pp. 2121-2159, July 2011

[9] E. Real, et al., "Regularized Evolution for Image Classifier Architecture Search", *Proceeding of the AAAI Conference on Artificial Intelligence*, vol 33, no 1, pp. 4780-4789, 2019

[10] D. P. Kingma and J. Ba., "Adam : A Method for Stochastic Optimization", *International conference on learning representation (ICLR)*, poster presentation, May 2015

[11] K.O. Stanley, et al., "Designing neural networks through neuroevolution", *Nature Machine Learning*, vol 1, pp 24-35, Jan 2019

[12] R. Pellerin, et al., "A survey of hybrid metaheuristics for the resource-constrained project scheduling problem", *European Journal of Operation Research*, vol 280, no 2, pp. 395-416, Jan 2020

[13] R. Elshaer and H. Awad, "A taxonomy review of metaheuristics algorithms for solving the vehicle routing problem and its variants", *Computer & Industrial Engineering*, vol 140, pp. 106242, Feb 2020

[14] H. D. Purnomo and H. M. Wee., "Maximizing production rate and workload balancing in two-sided assembly line using harmony search", *Computer & Industrial Engineering*, vol 76, 222-230, Oct 2014

[15] F.H.F. Leung, et al., "Tuning of the structure and parameters of a neural network using an improved genetic algorithm", *IEEE Transactions on Neural Network*, vol 14, no 1, pp. 79-88, Jan 2003

[16] C. F. Juang., "A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Desing", *IEEE Transactions on System, Man and Cybernetics*, vol 34, no 2, pp. 997-1006 April 2004

[17] L. M. R. Rere, et al., "Simulated Annealing Algorithm for Deep Learning", *Procedia Computer Science*, vol 72, pp. 137-144, 2015

[18] T. Salimans, et al., "Evolution strategies as a scalable alternative to reinforce learning", https://arxiv.org/abs/1703.03864, diakses 5 Juli 2022

[19] C. Liu, et al., "Auto-Deeplab: Hierarchical Neural Architecture Search for Semantic Image Segmentation", *Proceeding of the IEEE conference on computer vision and pattern recognition*, pp. 82-92, June 2019

[20] P. Lim, et al., "Evolutionary Cluster-Based Synthetic Oversampling Ensemble (ECO-Ensemble) for Imbalance Learning", IEEE transaction on cybernetics, vol 47, no 9, pp. 2850-2861, June 2016

[21] -, UCI machine learning repository, https://archive.ics.uci.edu/ml/datasets.php, diakses 10 Juni 2022

[22] Y. Shi and R. Eberhart, "A modified particle swarm optimization", IEEE International Conference on Evolutionary Computation Proceeding. IEEE World Congress on Computational Intelligence, 1998