An Improved Neural Network Based on the Parasitism – Predation Algorithm for an Automatic Voltage Regulator

Widi Aribowo†, **Bambang Suprianto**, **I Gusti Putu Asto Buditjahjanto**, **Mahendra Widyartono**, and **Miftahur Rohman**, Non-members

ABSTRACT

The parasitism – predation algorithm (PPA) is an optimization method that duplicates the interaction of mutualism between predators (cats), parasites (cuckoos), and hosts (crows). The study employs a combination of the PPA methods using the cascade-forward backpropagation neural network. This hybrid method employs an automatic voltage regulator (AVR) on a single machine system, with the performance measurement focusing on speed and the rotor angle. The performance of the proposed method is compared with the feed-forward backpropagation neural network (FFBNN), cascade-forward backpropagation neural network (CFBNN), Elman recurrent neural network (E-RNN), focused time-delay neural network (FTDNN), and distributed time-delay neural network (DTDNN). The results show that the proposed method exhibits the best speed and rotor angle performance. The PPA-CFBNN method has the ability to reduce the overshoot of the speed by 1.569% and the rotor angle by 0.724%.

Keywords: Parasitism – Predation Algorithm, PPA, Cascade-Forward Backpropagation Neural Network, Automatic Voltage Regulator, Neural Network, Elman Recurrent Neural Network

1. INTRODUCTION

Currently, electricity plays a strategic role in everyday life, influenced by every piece of equipment requiring electricity in homes, offices, companies, and factories [1]. The electrical power system is designed to operate at a set nominal value. Supply voltage experiencing a shift in behavior results in uncertain behavior and impacts the lifetime of equipment. Significant changes in the system dynamics are allowed at permitted levels [2].

Irregular demand for loads that can change at any time, results in the performance of electrical power systems approaching unsafe limits. The electrical power system control is an important element in generation fulfillment. Besides, the burden also needs to increase in complexity. The generator can oscillate around a balanced state when disturbed such as load changes, turbine fluctuation, and other factors. This is extremely dangerous for the electrical system. Most synchronous generators are installed with an excitation system, controlled by an automatic voltage regulator (AVR) to maintain the dynamic stability and power quality of the power system. The AVR functions as the main controller of the excitation system and can maintain the generator terminal voltage under any conditions [3]. The basic foundation of the AVR system is stable and responsive to changes in load. An automatic AVR is a buffer for the output voltage at a pegged level under various conditions.

Complex power systems need good AVR performance. Various approaches to setting automatic voltage regulators are reported in the existing literature; the predominant two types being conventional and computational. The conventional approaches often used in the AVR arrangement are the Cohen-Coon and Zeiglar-Nicholas [4].

In conventional methods, the controller becomes a problem when adjusting the gain from light to severe conditions. This is because settings in one load condition may differ in others. Due to the complex and non-linear adjustment of the AVR, a soft computing algorithm is implemented in this study to adjust the parameter acquisition.

Several computational methods have started to be used in AVR settings such as the genetic algorithm (GA) [5, 6], teaching-learning-based optimization (TLBO) [7, 8], sine cosine algorithm [9, 10], world cup optimization [11, 12], biogeography-based optimization [14], the Jaya optimization algorithm [15], global neighborhood algorithm [16], simulated annealing optimization algorithm [17], cuckoo search algorithm [18], firefly algorithm [19], whale optimization algorithm [20], and neural network [21, 22].

This paper presents an analysis of the AVR performance, set up using a neural network based

Manuscript received on August 3, 2020 ; revised on January 22, 2020 ; accepted on January 27, 2021. This paper was recommended by Associate Editor Kaan Kerdchuen.

The authors are with the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia.

†Corresponding author: widiaribowo@unesa.ac.id

Digital Object Identifier 10.37936/ecti-eec.2021192.241628

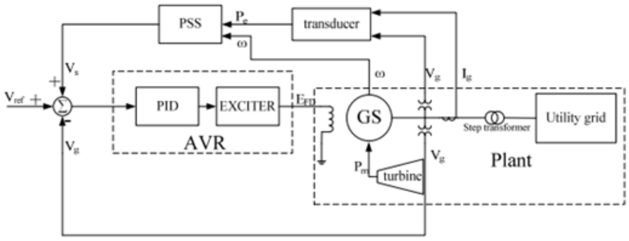

Fig. 1: *System structure [19].*

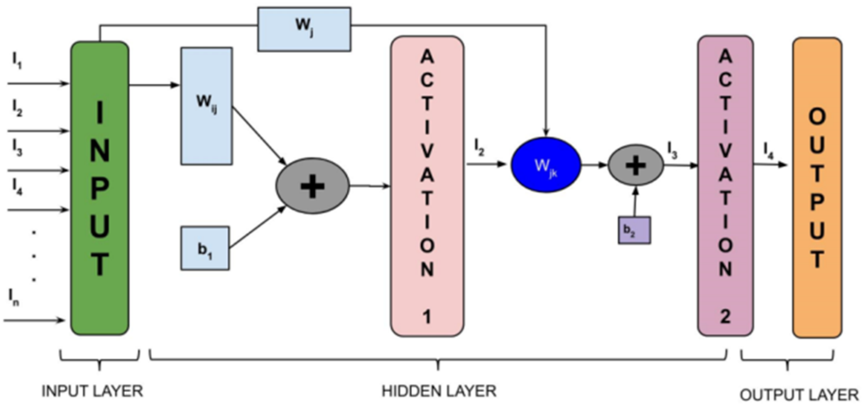

Fig. 2: *Cascade-forward backpropagation neural network structure.*

on the parasitism – predation algorithm (PPA). The implemented neural network is cascade-forward backpropagation, the performance of which is measured by focusing on the speed and rotor angle. The installed AVR is tested on a single machine, namely the Heffron-Phillips. The validation of the proposed method is compared with that for the feed-forward backpropagation neural network, cascade-forward backpropagation neural network, Elman recurrent neural network, focused time-delay neural network, and distributed time-delay neural network (DTDNN).

2. AUTOMATIC VOLTAGE REGULATOR

The generator control comprises the AVR and power system stabilizer (PSS) as shown in Fig. 1, which are used to maintain transient stability. The working principle of the AVR is to regulate the flow reinforcement (excitation) in the exciter. If the generator output voltage is below that of the nominal predetermined operator, the AVR will increase the current reinforcement (excitation) on the exciter. If the generator output voltage exceeds the nominal level, the AVR will reduce the current reinforcement (excitation) in the exciter. In the transient state, the generator affects equipment, especially over a short time, causing a clear drop in the terminal voltage of the machine.

3. CASCADE-FORWARD BACKPROPAGATION NEURAL NETWORK

The cascade feed backpropagation neural network (CFBNN) structure has input, hidden, and output layers. Characteristically, the input unit of the CFBNN is connected to the hidden unit and subsequently, the output. The weighted value of each input can be adjusted. The network between the input and the hidden layer is trained. Hidden units are added and stored on the network [23]. The weighting between the hidden unit and the output can be adjusted. The inputs of the CFBNN are I_1, I_2, …, I_j with the input data used in the training process. The structure of the CFBNN is shown in Fig. 2.

$$I_j\left(t\right) = \left(\int \sum_{i=1}^{j} W_{ij} I_i\left(t\right) + b_1\right) \cdot \sum_{i=1}^{j} W_i I_i\left(t\right) \quad (1)$$

$$I_2\left(t\right) = f\left(I_j\left(t\right)\right) = \frac{1}{1 + e^{I_j}} \quad (2)$$

In layer 2, the output from layer 1 ($I_2(t)$) is connected to neurons k with the weights in layer 2 (W_{jk}). The additional function of layer 2 is the sum of the output layer 1 ($I_2(t)$), weight (W_{jk}), and bias (b_2).

$$I_3(t) = \sum_{j=1}^{k} W_{jk} I_2(t) + b_2 \quad (3)$$

$$I_4(t) = f(I_3(t)) = \frac{1}{1+e^{I_3}} \quad (4)$$

The output unit $I_4(t)$ matches the target according to the input during training. The error is obtained by multiplying the derivative of the activation function.

$$\delta_k = t_i - I_4(t) f'(I_3(t)) \quad (5)$$

Weight improvements are used to correct the new W_{jk},

$$\Delta W_{jk} = \alpha \cdot \delta_i \cdot I_4 \quad (6)$$

4. PARASITISM – PREDATION ALGORITHM

The main role of the PPA is its relationship to the mutualism of natural life, inspired by the crow-cuckoo-cat system, known as parasitism – predation. The mutual relationship between a crow and cuckoo is apparent by the crow allowing the cuckoo to lay its eggs around those of the crows. This is very beneficial for crows since it protects the crow's eggs from predators such as cats. The cuckoo-cat-crow system is an element with its own tasks. The crow is the host, the cuckoo a parasite, and the cats a predator of the crow's nest [24].

This ecosystem system has an interrelationship. Low predation occurs when the cuckoo has a small positive effect on the crow. The cuckoo will cause the predator to become extinct (the difference between the density of the host in the presence of a parasite and the density of the host in the absence of a parasite below zero). Predation is considered to be intermediate when the interaction between the cuckoo and crow shifts from parasitism to commensalism (the difference between host density in the presence of a parasite and host density in the absence of a parasite is zero). On the other hand, predation is considered to be high when a cat causes the cuckoo to become extinct if the preventative element is weak. As with other metaheuristics, the PPA mathematical equation has a uniform initial:

$$Y_0 = Y_{\min} + rand^1 (Y_{\max} - Y_{\min}) \quad (7)$$

where $Y_{\min}$ and $Y_{\max}$ are the lower and upper limits for variables, while $rand^1$ is the random parameter taken from the uniform Gaussian distribution in the range of 0 to 1.

The PPA simulation uses several variable criteria such as the level of intrinsic addition for hosts with the variable r_1 set to 1. The level of parasite mortality (r_2) is set to 0.1, while the predator mortality level with variable r_3 is set to 0.3. Parasitic usefulness in changing its consumption to fitness ($\propto_1$) is set to the value of 0.2. The usefulness of a predator in shifting its predation to fitness ($\propto_2$) is set to the value of 0.25, while the number of sources consumed by the parasite (β_1) is set to a value of 0.1. The predator satiety level in predation (β_2) is set at 0.1, while half the saturation density in predation (c_1) is set at 0.1. The time wasted from predators due to parasitic resistance (c_2) is set to 0.1. The mortality rate depends on the density of the host (d_1) and is set to 0.01, while the mortality rate depending on the density of the parasite (d_2) is set to 0.01. The PPA has the following three main phases:

Phase 1: Nesting phase

The nesting phase represents the discovery of a crow's nest. Initially, the number of hosts is reduced by predators. The phase is intended to duplicate the flying of the hosts through two equations. The first equation serves to obtain the new position of the host by generating random prospective hosts.

$$Y_i^{t+1} = Y_i^t + L_f \left(Y_{r1} - Y_i^t\right) \quad \forall_i \in n_{\text{crow}} \quad (8)$$

where L_f is the Lévy flight step dimension calculated according to stable distribution (∞) with the ability to travel long distances with variations in step. The simple Lévy distribution is

$$f(q) = \sqrt{\frac{\gamma}{2\pi}} \frac{1}{(q-\mu)^{3/2}} \exp\left(-\frac{\sigma}{2(q-\mu)}\right) \quad (9)$$

with $0 < \mu < q < \infty$.

R_i produces random examples with the same flight rate as Lévy's. The step represents the measure of the scale associated with that of the subject.

$$R_i \sim \text{step} \oplus \text{Lévy}(\infty) \sim 0.01 \frac{a}{|x|^{1/z}} \quad (10)$$

$$u = N\left(0, \sigma_a^2\right) \quad (11)$$

$$y = N\left(0, \sigma_y^2\right) \quad (12)$$

$$\sigma_a = \left[\frac{\Gamma(1+a)\sin\frac{\pi\alpha}{2}}{\Gamma\left(\frac{1+\alpha}{2\alpha 2^{\left(\alpha-\frac{1}{2}\right)}}\right)}\right]^{1/\alpha} \quad (13)$$

$$\sigma_y = 1 \quad (14)$$

In the second stage, the inappropriate dimensions from the previous equation are reproduced.

$$Y_{i,o}^{\text{new}} = Y_{i,o}^{\min} + rand[0,1]\left(Y_{i,o}^{\max} - Y_{i,o}^{min}\right)$$
$$\forall_o \in \text{violated dimension} \quad (15)$$

The redesign is useful for improving the exploration capabilities and diversity of the search space.

Phase 2: Parasitism phase

In the parasitism phase, some hosts are displaced with the parasite. Initially, predators will cause the parasites to become extinct. This happens when predation is low. Parasites become extinct when predator efficiency is high. In this condition, the parasite is assumed to be efficient with the maximal limit being at the medium level.

$$Y_{i,\text{new}}^{\text{cuckoo}} = Y_{i,\text{old}}^{\text{cuckoo}} + H_g \cdot J \tag{16}$$

$$H_g = (Y_{r2} - Y_{r3})\, rand\,[0,1] \tag{17}$$

$$J = rand\,[0,1] > p_a \tag{18}$$

where $Y_{i,\text{old}}^{\text{cuckoo}}$ represents the variables selected using the roulette wheel method based on position, while H_g is a uniform Gaussian distribution step measure. Binary matrices J are used to protect the old cuckoo and retain the use of the search space, while p_a is an increasing factor.

Phase 3: Predation phase

The explosive growth of predators and a decrease in hosts causes a reduction in sufficient food sources for survival of the parasite under the assumption of high predation efficiency. This phase is based on the predator search mode. During this phase, no search mode is required because predators can track empty search spaces. The phase is composed of three steps.

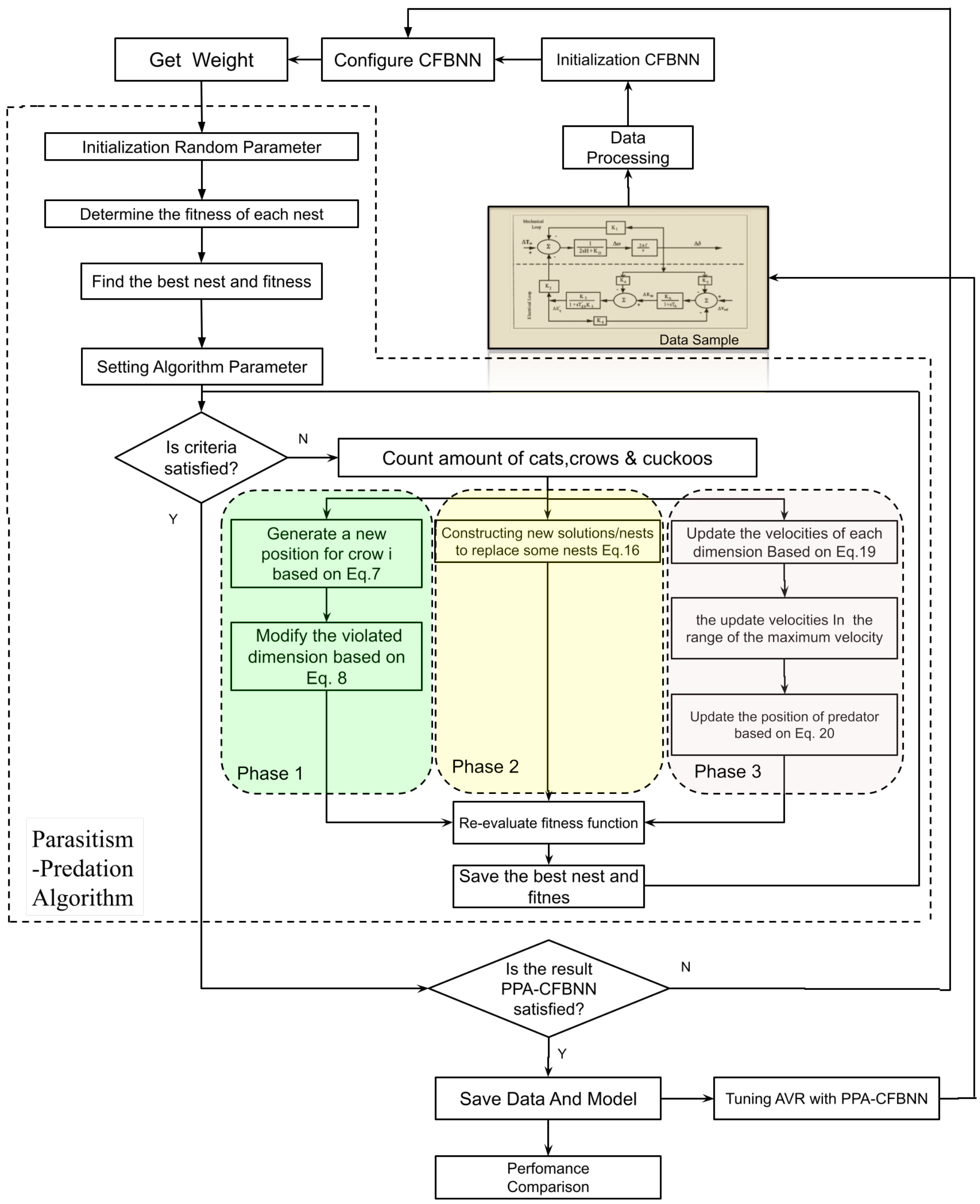

Fig. 3: *Proposed PPA-CFBNN flowchart.*

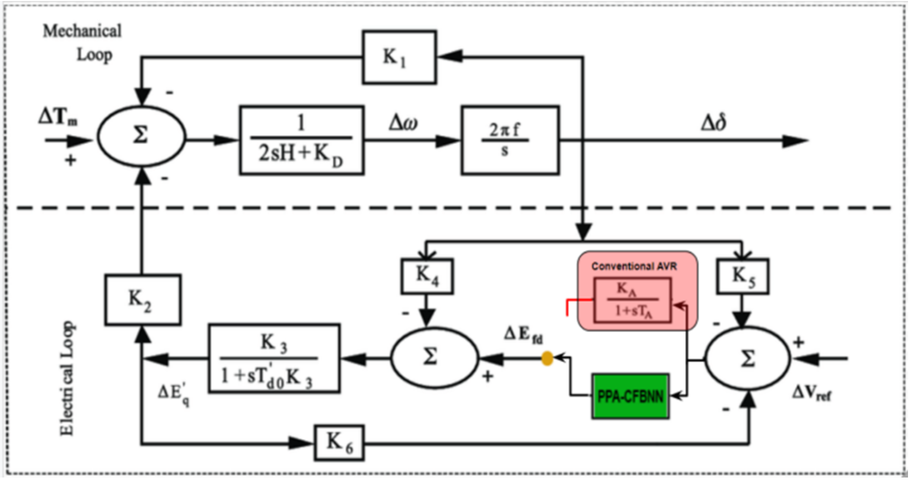

Fig. 4: *Position of the PPA-CFBNN.*

- Step 1: Regenerating the velocity variable

$$v_{i.d} = v_{i.d} + r \cdot t\,(Y_{\text{best}.d} - Y_{i.d}) \quad i = 1, \ldots, M \tag{19}$$

where $v_{i.d}$ is the velocity of the predator in i in dimension dth. $Y_{\text{best}.d}$ is the position and fitness of the predator with the highest value. $Y_{i.d}$ is the position of a predator, while r is the random variable and t is the fixed variable.

- Step 2: Update and retain the velocity within the maximum speed limit (0.25 < speed limit <1).
- Step 3: Regenerate the position of predator i

$$Y_{i.d} = Y_{i.d} + v_{i.d} \tag{20}$$

5. PROPOSED PPA-CFBNN MODEL

The steps to applying the PPA and CFBNN methods in setting the AVR can be illustrated as shown in Fig. 3. The first step involves taking and processing the sample data for use in initializing and configuring the CFBNN. The initial weighting data from the CFBNN is then retrieved and processed using the PPA method; initialization of the PPA method uses random values. The results of the PPA provide the potential weight for the CFBNN applied in the network.

6. RESULTS AND DISCUSSION

The proposed method is validated by comparing it with the FFBNN, CFBNN, Elman-RNN, and FTDNN approaches, under the assumption that the system has the same parameters. Fig. 3 shows a single machine generator, employing the Heffron-Phillips model. Fig. 4 shows the proposed method installed on the system with a conventional AVR replaced by the PPA-CFBNN.

This research focuses on speed and the rotor angle, with speed and rotor angle performance being based on undershoot, overshoot, and time taken to settle. The results of the proposed method are compared with other approaches using variations in load and hidden layers. In the first experiment, the generator is given a light load of 10%.

The results of the first experiment can be seen in Figs. 5 and 6. The measurement of the ability of the proposed method focuses on the undershoot and overshoot aspects. Fig. 5 shows the speed response from the generator. The overshoot value for the proposed method is 0.381 at 45 s. The undershoot value for speed response when using the PPA-CFBNN method is better than that of the DTDNN which has a value of −0.5098. Undershoot and overshoot values using the other methods are very small compared to the DTDNN values.

Fig. 6 shows the results of the rotor angle when given a 10% load. The rotor angle of the proposed method shows an undershoot value of −2.766 and an overshoot of 0.6306. While the values of the other methods are very small with the DTDNN method. The results of each method in further detail are presented in Table 1.

Table 1 shows the overshoot value of the PPA-CFBNN method speed response at 0.3841. This value is highest using the E-RNN method. Meanwhile, the undershoot value of the speed response is best with a value of −0.5072. The results of the rotor angle employing the PPA-CFBNN method reveal that the best values for overshoot and undershoot are 0.6306 and −2.7661, respectively.

In the second experiment, the load is increased to 50%, the results of which are presented in Table 2, with the overshoot and undershoot values of the speed response using the PPA-CFBNN method being the best at 0.4293 and −0.5630, respectively.

Similarly, the best values of overshoot and undershoot for the rotor angle are obtained equating to 0.7051 and −3.1524, respectively.

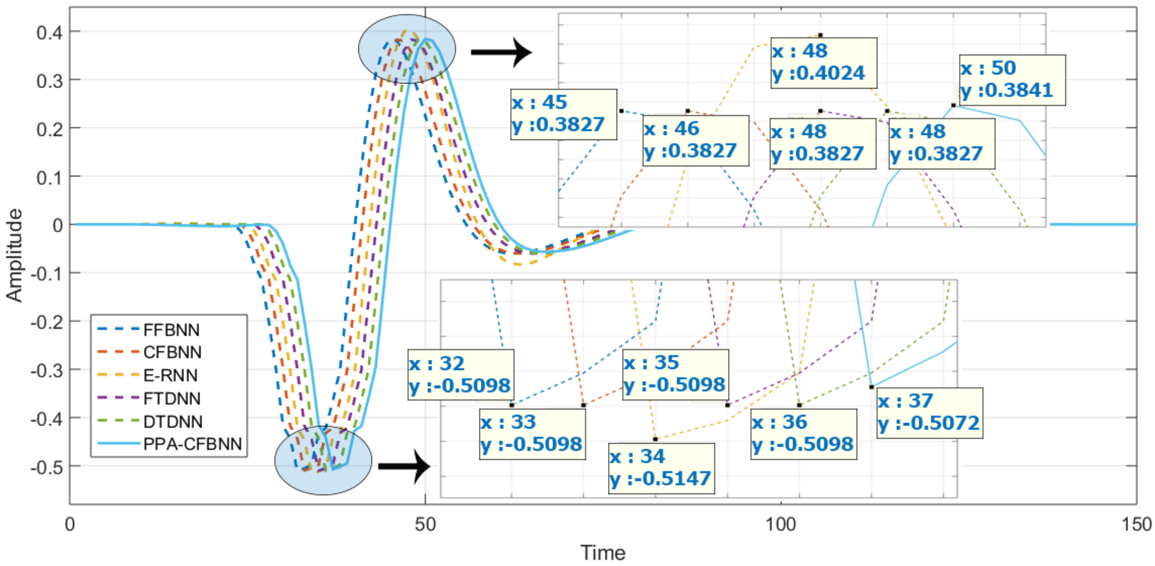

Fig. 5: *Speed response with 10% of the load.*

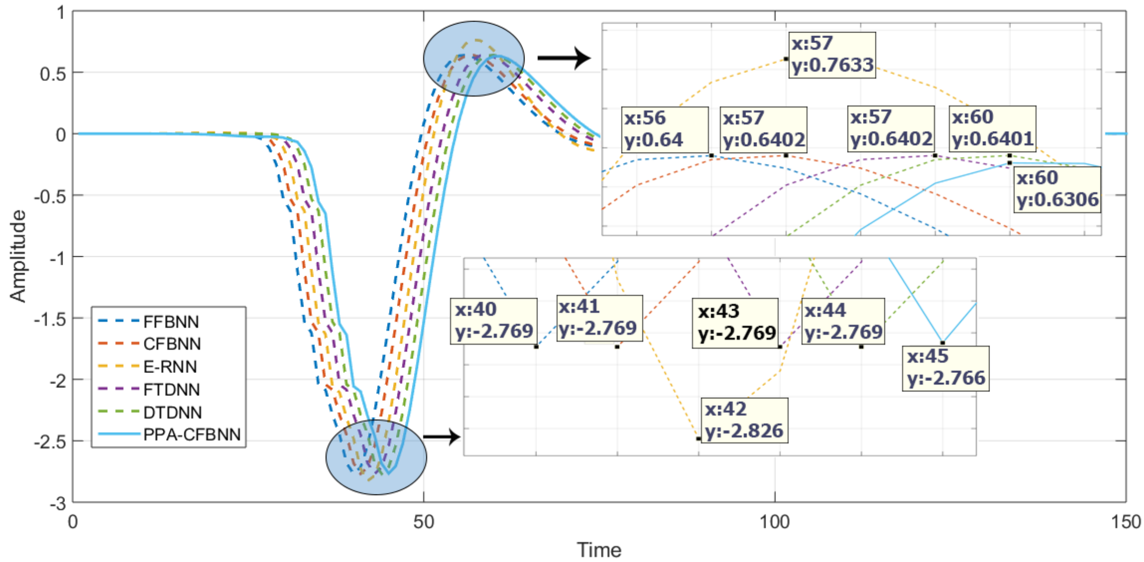

Fig. 6: *Rotor angle response with 10% of the load.*

For the third experiment, the load is increased to 80%, the results of which are presented in Table 3. The undershoot and overshoot for the speed response using the PPA-CFBNN method obtain the best values of -0.5979 and 0.4596, respectively. Meanwhile, the rotor angle response values for undershoot and overshoot are -3.4159 and 0.7821, respectively.

7. CONCLUSION

This research examines the parasitism – predation algorithm method combined with the cascade-forward backpropagation, collectively called PPA-CFBNN. This combination of methods is used to set the AVR on the generator. This study employs three different loads to measure performance. The hidden layer value of the neural network is 4 for all methods. In this study, the speed value and rotor angle using the neural network algorithm reveal almost the same results. The CFBNN method exhibits slightly better results compared to other neural network methods. However, the capabilities of the CFBNN are still below that of the proposed PPA-CFBNN method. The results of the study reveal that the proposed method has the best average value. In this paper, the proposed approach is tested with the use of a simple, single machine system. To determine the performance and durability of the PPA-CFBNN method further research should be conducted using a larger and more complex system.

Table 1: *Results of training with 10% of the load.*

Method	Speed Response				Rotor Angle Response			
	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)
FFBNN	−0.5098	0.3827	0.544	85.5572	−2.7687	0.6400	4.6299×10^{-8}	83.2848
CFBNN	−0.5098	0.3827	0.5197	85.5823	−2.7687	0.6402	4.5794×10^{-8}	83.2913
E-RNN	−0.5147	0.4024	0.7095	85.5424	−2.8263	0.7633	1.9659×10^{-6}	81.5118
FTDNN	−0.5098	0.3827	0.4688	85.5805	−2.7687	0.6402	4.5881×10^{-8}	83.2910
DTDNN	−0.5098	0.3827	0.7234	85.5778	−2.7687	0.6401	4.5764×10^{-8}	83.2897
PPA-CFBNN	−0.5072	0.3841	2.48×10^{-10}	87.0607	−2.7661	0.6306	0.8081	83.8766

Table 2: *Results of training with 50% of the load.*

Method	Speed Response				Rotor Angle Response			
	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)
FFBNN	−0.5711	0.4296	0.4819	88.1445	−3.1694	0.7161	2.6061×10^{-7}	85.6717
CFBNN	−0.5710	0.4294	0.5937	88.1705	−3.1692	0.7165	2.7709×10^{-7}	85.6926
E-RNN	−0.5795	0.4555	0.6714	88.0012	−3.2385	0.8592	2.4764×10^{-6}	83.8869
FTDNN	−0.5710	0.4294	0.3998	88.1720	−3.1692	0.7164	2.7593×10^{-7}	85.6923
DTDNN	−0.5710	0.4294	0.6879	88.1769	−3.1693	0.7164	2.7710×10^{-7}	85.6948
PPA-CFBNN	−0.5630	0.4293	0.0065	93.3397	−3.1524	0.7051	0.2729	88.7103

Table 3: *Results of training with 80% of the load.*

Method	Speed Response				Rotor Angle Response			
	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)	Under-shoot	Over-shoot	Rise Time (s)	Settling Time (s)
FFBNN	−0.6151	0.4668	0.589	88.65	−3.4696	0.7743	0	85.9820
CFBNN	−0.6151	0.4667	0.649	88.56	−3.4696	0.7748	0	85.9367
E-RNN	−0.6254	0.4952	0.6408	88.3191	−3.5531	0.9341	2.881×10^{-6}	84.1216
FTDNN	−0.65151	0.4667	0.3391	88.5645	−3.4696	0.7747	2.880×10^{-7}	85.9379
DTDNN	−0.6151	0.4667	0.6584	88.5585	−3.4696	0.7747	2.885×10^{-7}	85.9358
PPA-CFBNN	−0.5979	0.4596	0.2179	96.0734	−3.4159	0.7821	0.5	90.5663

REFERENCES

[1] W. Aribowo, S. Muslim, and I. Basuki, "Generalized Regression Neural Network For Long-Term Electricity Load Forecasting," in *2020 International Conference on Smart Technology and Applications (ICoSTA)*, Surabaya, Indonesia, 2020, pp. 1–5, doi: 10.1109/ICoSTA48221.2020.1570611361.

[2] S. Majumdar, K. Mandal, and N. Chakraborty, "Performance Study of Mine Blast Algorithm for Automatic Voltage Regulator Tuning," in *Annual IEEE India Conference (INDICON)*, Pune, India, 2014, pp. 1–6, doi: 10.1109/INDICON.2014.7030488.

[3] H. Saadat, *Power System Analysis*. New York, USA: McGraw-Hill, 1999.

[4] M. W. Iruthayarajan and S. Baskar, "The Evolutionary Algorithms Based Design of Multivariable PID Controller," *Expert Systems with Applications*, vol. 36, pp. 9159–9167, 2009

[5] N. K. Yegireddy, S. Panda, T. Papinaidu, and K. P. K. Yadav, "Multi-Objective Non Dominated Sorting Genetic Algorithm-II Optimized PID Controller For Automatic Voltage Regulator Systems," *Journal of Intelligent & Fuzzy Systems*, vol. 35, no. 5, pp. 4971–4975, 2018.

[6] N. Y. Bulatov and A. V. Kryukov, "Optimization of Automatic Regulator Settings of The Distributed Generation Plants on The Basis Of Genetic Algorithm," in *2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, Chelyabinsk, Russia, 2016, pp. 1–6, doi: 10.1109/ICIEAM.2016.7911456.

[7] S. Priyambada, P. K. Mohanty, and B. K. Sahu, "Automatic Voltage Regulator using TLBO Algorithm Optimized PID Controller," in *9th International Conference on Industrial and Information Systems (ICIIS)*, Gwalior, India, 2014, pp. 1–6.

[8] S. Chatterjee and V. Mukherjee, "PID Controller For Automatic Voltage Regulator Using Teaching–Learning Based Optimization Technique," *International Journal of Electrical*

Power & Energy Systems, vol. 77, pp. 418–429, May 2016.

[9] B. Hekimoğlu, “Sine-Cosine Algorithm-Based Optimization For Automatic Voltage Regulator System,” *Transactions of the Institute of Measurement and Control*, vol. 41, no. 6, pp. 1761–1771, 2019.

[10] J. Bhookya and R. K. Jatoth, “Optimal FOPID/PID Controller Parameters Tuning For The AVR System Based On Sine–Cosine-Algorithm,” *Evolutionary Intelligence*, vol. 12, no. 4, pp. 725–733, 2019.

[11] N. Razmjooy, M. Khalilpour, and M. Ramezani, “A New Meta-Heuristic Optimization Algorithm Inspired by FIFA World Cup Competitions: Theory and Its Application in PID Designing for AVR System,” *Journal of Control, Automation and Electrical Systems*, vol. 27, no. 4, pp. 419–440, 2016.

[12] U. Güvenç, T. Yiğit T, A. H. Işik, and İ. Akkaya, “Performance Analysis Of Biogeography-Based Optimization For Automatic Voltage Regulator System,” *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 24, pp. 1150–1162, 2016, doi:10.3906/elk-1311-111.

[13] T. A. Jumani *et al.*, “Jaya Optimization Algorithm For Transient Response And Stability Enhancement Of A Fractional-Order PID Based Automatic Voltage Regulator System,” *Alexandria Engineering Journal*, vol. 59, no. 4, pp. 2429–2440, 2020, doi: 10.1016/j.aej.2020.03.005.

[14] H. Gözde, M. C. Taplamacioğlu, and M. Ari, “Simulation Study For Global Neighborhood Algorithm Based Optimal Automatic Voltage Regulator (AVR) System,” in *5th International Istanbul Smart Grid and Cities Congress and Fair (ICSG)*, Istanbul, Turkey, 2017, pp. 46–50, doi: 10.1109/SGCF.2017.7947634.

[15] R. Lahcene, S. Abdeldjalil, and K. Aissa, “Optimal Tuning Of Fractional Order PID Controller For AVR System Using Simulated Annealing Optimization Algorithm,” in *5th International Conference on Electrical Engineering - Boumerdes (ICEE-B)*, Boumerdes, Algeria, 2017, pp. 1–6, doi: 10.1109/ICEE-B.2017.8192194.

[16] Z. Bingul and O. Karahan, “A Novel Performance Criterion Approach To Optimum Design Of PID Controller Using Cuckoo Search Algorithm For AVR System,” *Journal of The Franklin Institute*, vol. 355, no. 13, pp. 5534–5559, 2018.

[17] M. N. Britto, S. Sivakumar, K. Ompriyadharshini, R. Mrinalini, and S. Nagarajan, “Tuning Of Finest Controller For AVR Unit – An Examination With Firefly-Algorithm,” in *IEEE International Conference on System, Computation, Automation and Networking (ICSCA)*, Pondicherry, India, 2018, pp. 1–5, doi: 10.1109/ICSCAN.2018.8541164.

[18] A. M. Mosaad, M. A. Attia, and A. Y. Abdelaziz, “Whale Optimization Algorithm To Tune PID And PIDA Controllers On AVR System,” Ain Shams Engineering Journal, vol. 10, no. 4, pp. 755–767, 2019

[19] W. Aribowo, “Focused Time Delay Neural Network For Tuning Automatic Voltage Regulator,” *EMITTER International Journal of Engineering Technology*, vol. 7, no. 1, pp. 34–43, 2019

[20] S. Alkhalaf, “Modeling the Automatic Voltage Regulator (AVR) Using Artificial Neural Network,” in *International Conference on Innovative Trends in Computer Engineering (ITCE)*, Aswan, Egypt, 2019, pp. 570–575, doi: 10.1109/ITCE.2019.8646450.

[21] A. A. Bhutto, F. A. Chachar, M. Hussain, D. K. Bhutto, and S. E. Bakhsh, “Implementation Of Probabilistic Neural Network (PNN) Based Automatic Voltage Regulator (AVR) For Excitation Control System In Matlab,” in *2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Sukkur, Pakistan, 2019, pp. 1–5.

[22] G. Bal, O. Kaplan, and S. S. Yalcin, “Artificial Neural Network Based Automatic Voltage Regulator For A Stand-Alone Synchronous Generator,” in *8th International Conference on Renewable Energy Research and Applications (ICRERA)*, Brasov, Romania, 2019, pp. 1032–1037, doi: 10.1109/ICRERA47325.2019.8997098.

[23] W. Aribowo, S. Muslim, Munoto, B. Suprianto, U. T. Kartini, and I. G. P. Asto Buditjahjanto, “Tuning of Power System Stabilizer Using Cascade Forward Backpropagation,” in *2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE)*, Surabaya, Indonesia, 2020, pp. 1–5, doi: 10.1109/ICVEE50212.2020.9243204.

[24] A.-A. A. Mohamed, S. A. Hassan, A. M. Hemeida, S. Alkhalaf, M. M. M. Mahmoud, and A. M. B. Eldin, “Parasitism – Predation algorithm (PPA): A novel approach for feature selection,” *Ain Shams Engineering Journal*, vol. 11, no. 2, pp. 293–308, 2020.

Widi Aribowo received his Bachelor of Engineering and Master of Engineering from Sepuluh Nopember Institute of Technology (ITS), Surabaya, Indonesia, in 2005 and 2009, respectively. He is currently a lecturer at the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia. His research interests include power system, load forecasting, control, and artificially intelligent.

Bambang Suprianto received his Bachelor degree in Electronic Engineering Education from Universitas Negeri Surabaya, Surabaya, Indonesia, in 1986. He received his Master of Engineering and Doctor of Electrical Engineering from Sepuluh Nopember Institute of Technology (ITS), Surabaya, Indonesia, in 2001 and 2012, respectively. He is currently a lecturer at the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia. His research interests include power system, control, and electronic.

I Gusti Putu Asto Buditjahjanto received his Bachelor of Engineering, Master of Engineering, and Doctor of Electrical Engineering from Sepuluh Nopember Institute of Technology (ITS), Surabaya, Indonesia, in 1998, 2003, and 2011, respectively. He is currently a lecturer at the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia. His research interests include control, computation, optimization, and artificially intelligent.

Mahendra Widyartono received his Bachelor of Engineering from Sepuluh Nopember Institute of Technology (ITS), Surabaya, Indonesia, in 2006, and his Master of Engineering from Brawijaya University, Indonesia, in 2012. He is currently a lecturer at the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia. His research interests include power system and renewable energy.

Miftahur Rohman received his Bachelor of Engineering and Master of Engineering from Sepuluh Nopember Institute of Technology (ITS), Surabaya, Indonesia, in 2011 and 2014, respectively. He is currently a lecturer at the Department of Electrical Engineering, Universitas Negeri Surabaya, Indonesia. His research interests include power system, electronic, and control.