New Algorithm for Neural Network Optimal Power Flow (NN-OPF) including Generator Capability Curve Constraint and Statistic-fuzzy Load Clustering

Mat Syai'in
Department of Electrical Engineering
And Surabaya Shipbuilding State Polytechnic (PPNS)
Sepuluh Nopember Institute of technology (ITS), Indonesia

Adi Soeprijanto
Department of Electrical Engineering
Sepuluh Nopember Institute of technology
(ITS)
Indonesia

Eko Mulyanto Yuniarno
Department of Electrical Engineering
Sepuluh Nopember Institute of technology
(ITS)
Indonesia

ABSTRACT
This paper presents a novel algorithm of an optimal power flow (OPF), which possible be used for real time applications. The proposed algorithm uses neural networks (NNs) to model the generator capability curves and set them as the output power constraints of the generators. In addition, it also uses NNs to replace an OPF based on the particle swarm optimization (PSO) method so as to run in real time. Also, in order for the proposed algorithm to be able to account for various load conditions, the statistic-fuzzy load clustering method is used to classify the loads based on the patterns of load curves. A similarity index is then defined to associate the similarity among different patterns of load distribution curves. This similarity index is also included in the training process of the final constructed neural networks. A 500 kV Java-Bali power system consisting of 23 buses is used as a benchmark system to validate the proposed NN-based OPF. The simulation results show that that the values obtained from the proposed algorithm are in great agreement with those calculated from the PSO-OPF.

Keywords
Generator Capability Curve, Neural Network Optimal Power Flow, Particle Swarm Optimization, Statistic-Fuzzy

1. INTRODUCTION
The objective of an optimal power flow (OPF) is usually to minimize the line losses and the total fuel cost of the generating units, which are subjected to active and reactive power, bus voltage, and line flow limits. Conventional solution techniques offer good results but when the search space is non- linear and has discontinuities, these techniques become difficult to solve and do not always get the optimal solution [1]. To solve this problem, artificial intelligence (AI) methods have been widely adopted. The most popular intelligence optimization technique already applied were genetic algorithm, fuzzy, simulated annealing (SA), expert systems, neural networks (NNs), particle swarm optimization (PSO) and the hybrid of them [1-11]. Among these, PSO based methods are the ones recently received greatest attention due to its capability in achieving global optimal solutions [7].

Normally, the constraints for a generator in an optimal power flow (OPF) are defined as rectangular constraints - curves only require two sets of inequality constraints (Pmin-Pmax, Qmin-Qmax) [12-13]**.** However, such constraints may overestimate the cost of the generation. Therefore, it is highly desirable to see how much cost would be reduced if the output power limits of generators are imposed by the actual generator capability curves (GCCs) [14]. A GCC faithfully describes the real and reactive power capabilities of a generator.

We have developed a PSO-OPF [19], which takes GCCs into account. Although, PSO can ensure convergence and lead to accurate results, its computation time is usually long and not suitable for online application. It has been recognized that NNs can be used for online application [15]. Normally, the training process of an NN is done offline due to its intensive computation. However, once the training is completed, a trained NN, like a human brain, can associate a large number of output patterns corresponding to each input patterns in an extremely fast time. Moreover, we also use NNs to approximate the developed offline PSO-OPF model, which is possible to run in real time. The training method employed in this paper is constructive backpropagation [16]. In addition, in order for the NN-based OPF to account for multitides of load conditions, the statistic-fuzzy load clustering method [17] is used to classify the loads based on the patterns of load curves. A similarity index is defined to associate the similarities among different patterns of load curves. This similarity index is also included in the training process of NNs.

The rest of this paper is arranged as follows: Section II describes the proposed solution procedures. Section III presents the simulation results of the proposed NN-based OPF method for the 500 kV Java-Bali power system, which consists of 23 buses. This system is the biggest in Indonesia, supporting the area of 7 provinces across Java and Bali Islands. The simulation results are verified with the offline PSO-OPF method. Finally, a conclusion is given in section IV

2. METHODOLOGY
This section describes the proposed solution procedure of our NN-based OPF method applicable for operation in real time. Fig.1 describes the flowchart of the overall solution procedures. As illustrated in the figure, the procedure consists of five stages, which will be described in details in the following subsections.

2.1 Stage 1: Construct GCC using NNs

The first stage is to develop a NN model for a GCC. The data used in the training process of the NN are the sample points along a GCC provided by the generator manufacture's data sheet [14]. The NN model consists of one input, one output and one hidden layer.To obtain the weighting coefficients of the NN, we first convert all the (P, Q) pairs into the polar forms, (R,θ) as shown in Fig. 2. Then, we set θ as the input and R as the output. The weighting coefficients can consequenly be obtained via constructive back propagation method. Hence, one can easily restore a GCC for a given values of θ as shown in Fig. 3

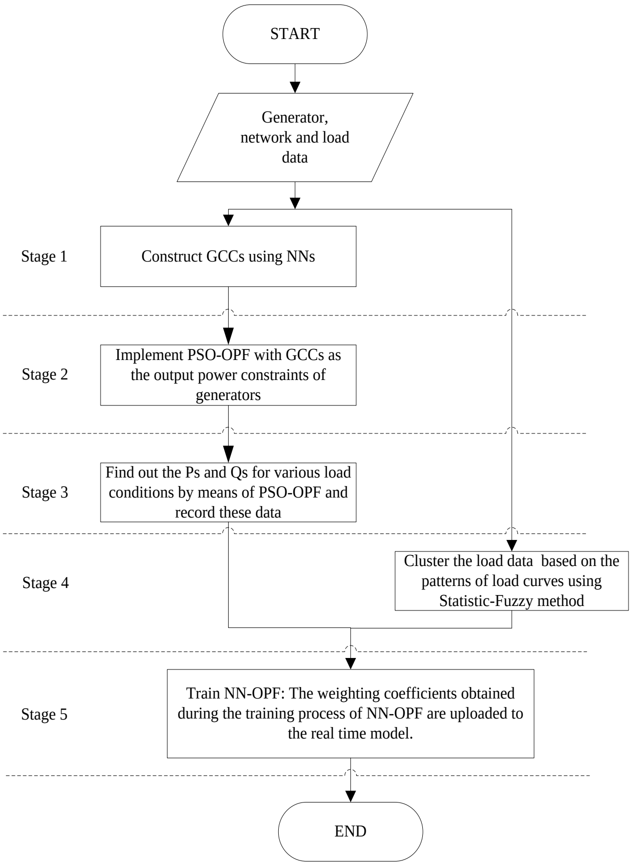

Fig. 1. Flowchart design of NN-OPF.

Fig. 3 shows the similarity between the GCCs constructed by the NN and those from the data sheet. The main advantage of the NN-based model is that it is much easier to be included in an optimal power flow. In the next subsection, we will describe how such a model can be included in the PSO-OPF.

2.2 Stage 2: Implement GCCs as the output power constraints of generators in PSO-OPF

The overall flow chart in Fig. 4 summarizes the program algorithm of implementing a PSO-OPF with GCCs as the output power constraints of generators. The generator, network and load data are first read into the program. The generator data are passed to stage 1 where the GCCs are constructed. Then generator data together with the network and load data are passed to the load flow program. However, before the load flow calculation is carried out, the initialized Ps and Qs for the PV buses need to be checked if they are within their GCC limits. The checking alogrithm can be summarized in Fig. 5. As seen from the figure, the initialized (P,Q) pairs are first converted into the (R, θ) pairs. Then ,θ is passed to the NN model, built in stage 1 to get R_{ref}. R is then compared with R_{ref}. If R is smaller than or equal to R_{ref}, it means the initialized (P,Q) pairs are within the GCC limits (see Fig. 6); otherwise , R is set to R_{ref} . After the checking process, the results are converted back to the corresponding P, Q values, which are needed for the load flow calculation. After the load flow calculation, one checks if the voltages at the PV buses are within the proper range, and the slack bus is within its GCC limits. If any of the two is violated, the values of P and Q are re-initialized as shown in Fig. 4.

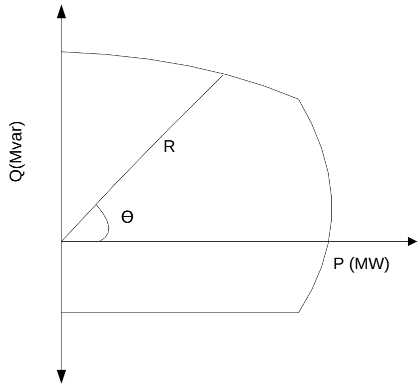

Fig. 2 Data Pair for NN Learning θ and R

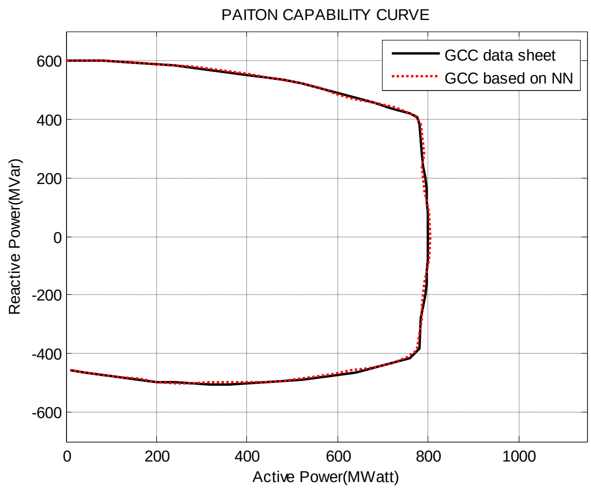

Fig. 3. Comparison between GCC data sheet and GCC based on NN

When the checking process is completed, optimization can be carried out via particle swarm optimization (PSO). PSO is an iterative process for allocating the global optimal solution by comparing the values of the objective functions for all possible combinations of feasible generation. Equations (1)-(3) describe the iterative formula of PSO.

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{1}$$

$$V_i^{k+1} = \omega V_i^k + c_1 rand_1\left(Pbest_i - X_i^k\right) + c_2 rand_2\left(Gbest_i - X_i^k\right) \tag{2}$$

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{Iter_{max}} iter \tag{3}$$

With:

V_i^k = individu velocity i at iteration k
ω = weight parameter
c_1 c_2 = acceleration coefisien
$rand_1$, $rand_2$= random value between 0 and 1
X_i^k= individu position i at iteration k
$Pbest_i$= The best position of individu i until iteration k
$Gbest_i$= The best position of community until iteration k
ω_{min} , ω_{max} = initial and final weight
$Iter_{max}$ = maximum iteration number
$iter$ = number of current iteration

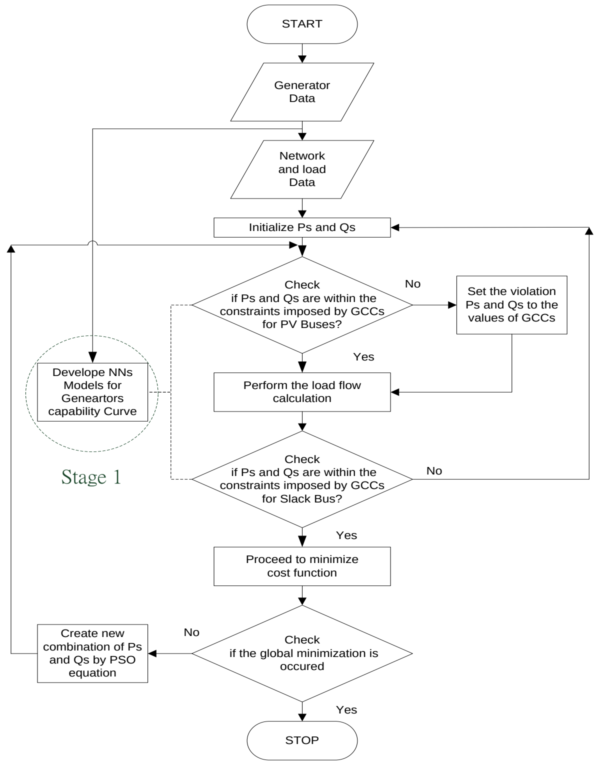

Fig. 4. Flowchart of PSO-OPF design

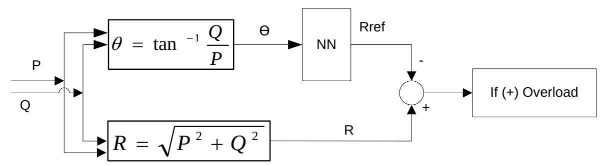

Fig. 5. Security check Algorithm.

2.3 Stage 3: Find out the P and Q for various load conditions by means of PSO-OPF and record these data

Since the PSO usually takes a long time to converge, the PSO-OPF is not suitable to be used for online applications. Therefore, we propose an NN model to replace the developed PSO-OPF for real time applications. In order for the NN-based OPF to account for various load conditions in real time, we need to train our NN model for various load conditions offline. For example, Fig. 7 shows the load distribution curves for 6 different times in an hour across 23 buses. To have the NN-based OPF to account for these load conditions in real time, we will perform offline PSO-OPFs for these load conditions. The converged Ps and Qs are then recorded for the training process of the NN. To have the NN-based OPF to account for a multitude of various load curves, load clustering is used for improving the efficiency of the training process. Stage 4 describes in details how the load clustering is implemented

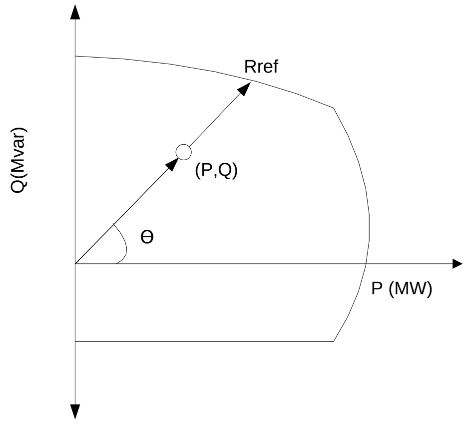

Fig. 6. Relationship between P,Q, θ, R and Rref

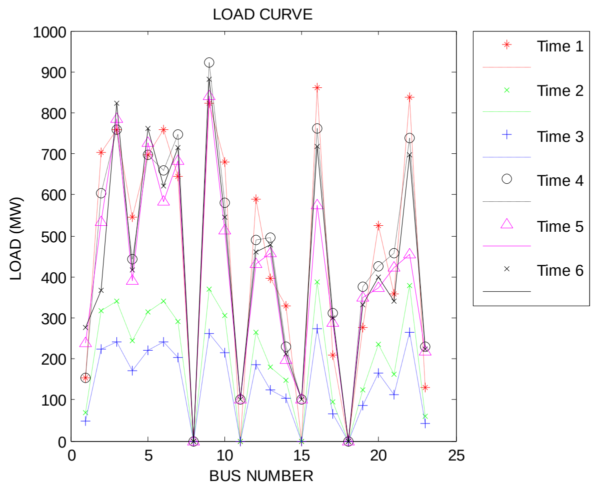

Fig.7. Load distribution curves over a 23-bus system at different time

2.4 Stage 4: Cluster the load data based on the patterns of load curves using statistic-fuzzy method

The recorded load data from stage 3 are enormous. To deal with them efficiently, we employ the concept of load clustering. The purpose of clustering is to place objects into groups such that objects in a given group have tendency to be similar to each other, and those in different cluster tend to be dissimilar. The similarity of any two load distribution curve can be measured by the similarity index. The similarity index is based on the cosine angle between the two load curve vectors [17] and is given by (4).

$$W_{ik} = \frac{\sum_{l=1}^{n} X_{il} X_{kl}}{\sqrt{\left(\sum_{l=1}^{n} X_{il}^2\right)\left(\sum_{l=1}^{n} X_{kl}^2\right)}} \quad (4)$$

W_{ik} is the similarity index between the two load distribution curve vectors (load curve *i* and load curve *k*)

X_{il} is normalized load curve *i* at node *l*
X_{kl} is normalized load curve *i* at node *k*

Note that one needs to normalize the load curves before evaluating (4). For example if all the curves in Fig. 8 are normalized with respect to their corresponding peak values, these three curves will concide to one, as shown in Fig. 9.

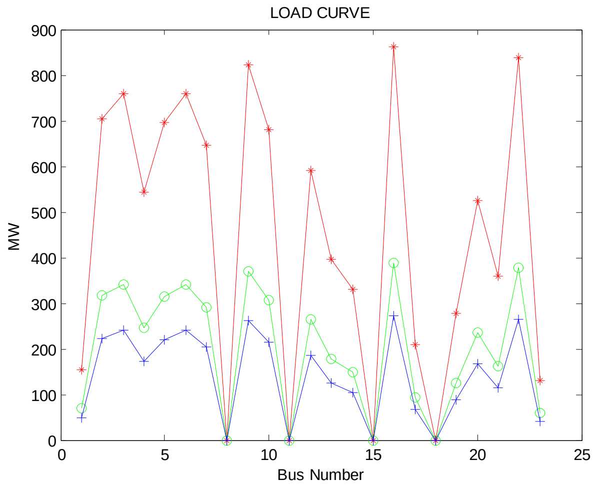

Fig.8. Example of three load curve surfaces in MW.

The similarity relationship among all the load curves form a matrix whose elements are the similarity incices W_{ik} between two load curves. The diagonal W_{ik} are always 1 because the load curves are compared to themselves. The off-diagonal W_{ik} are between the value of 0 and 1. In order to W_{ik} can used to cluster load curve, it needs to process in fuzzy system. The process of fuzzyfication usually uses triangle, trapezoids, or normal curve distribution. In this paper we used statistic equation (4) to process fuzzification. The fuzzy rule base used in this paper is min-max system and combined with equation (5).

$$FW_{ik} = \sum_{l=1}^{n} W_{il} W_{kl} \tag{5}$$

FW_{ik} The nearness index between the two load curve vectors (load curve i and load curve k) after fuzzification process.

W_{il} The similarity index between the two load curve vectors (load curve i and load curve l)

W_{kl} The similarity index between the two load curve vectors (load curve k and load curve l)

FW_{ik} is calculated using equation (5) but the product of two elements is replaced by taking the minimum one and the addition of two products by taking the maximum one. For load curves whose FW_{ik} are greater than 0.9, they are assumed to be one cluster. Inside each of the cluster, the average of all the load curve patterns are obtained in order to serve as a representative for that cluster. These representatives are used for computing the values of FW_{ik} for new sets of load curves.

2.5 Stage 5: Train NN-OPF: The weighting coefficients obtained during the training process of NN-OPF are uploaded to the real time model

To have PSO-OPF to be able to run in real time, an NN is constructed to replace the PSO-OPF. The training process of NN requires 3 sets of input, and they are the total apparent power of the load, total active/reactive power of the load, and the nearness index (FW_{ik}), As shown in Fig. 10. The activation functions [18] of the hidden layers were chosen to be tansig and logsig. The training method used was based on the constructive back propagation method [16].

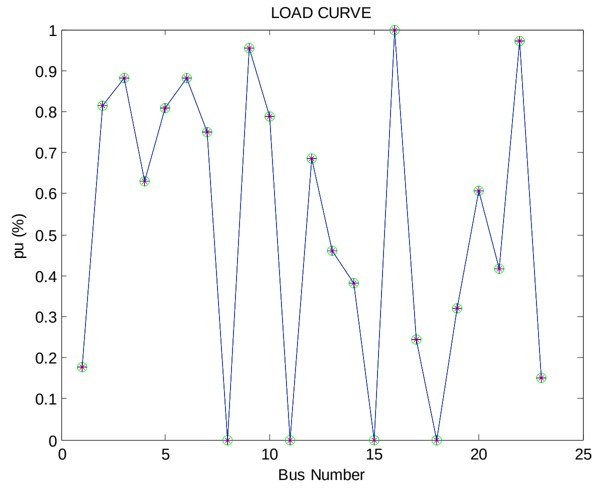

Fig. 9. Same load curve surface in per unit of three different load curve in MM

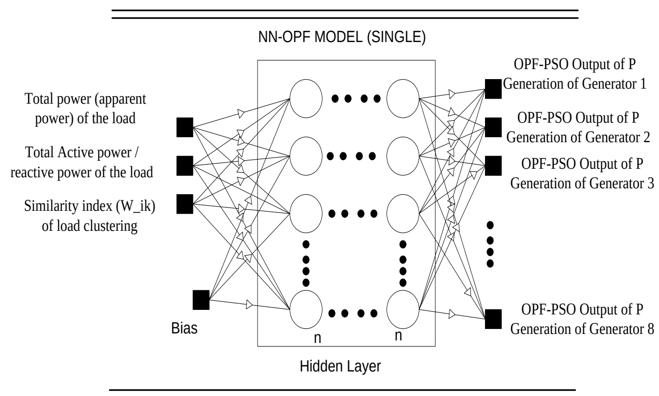

Fig.10. NN-OPF Model (single) for the active power optimization.

The range of the load variations can be very wide. If we only use one NN to account for all changes of the load, it may take a great number of neurons, and consequently a long time to train the NN. Instead, it is better to construct several smaller NNs working in parallel for different ranges of the loads as shown in Fig. 11. These NN have the same number of neurons. However, the weightings for each network may be different, and their values depend on its corresponding range. In this paper, 15 NNs were constructed because the load variation was ranged from 25% to 100%, and each NN is responsible for 5% range of the load. The number of hidden layer in each NN is three. The first layer consists of 11 neurons, the second 25, and the third 25.

3. SIMULATION AND ANALYSIS

The system used for simulation is the 500 kV Java-Bali Power System (see Fig. 12). The cost function for each generator is shown in TABLE I. The network data are listed in the Appendix. The performance comparison between the NN-based OPF and the PSO-OPF can be seen in TABLE II. As seen from the table, the difference between NN-based OPF and PSO-OPF on the operation cost and power generation is 0.16 % and 0.0%, respectively, which is very small.

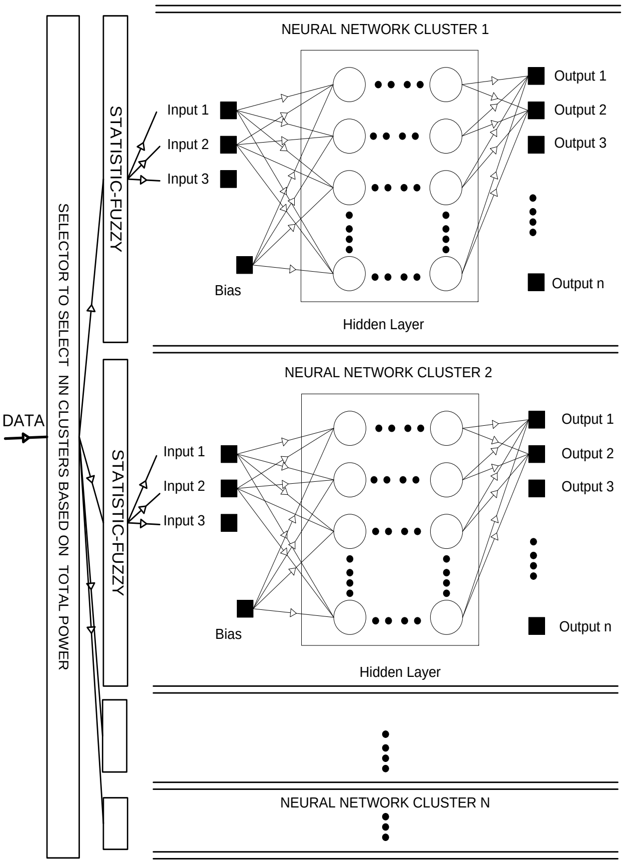

Fig.11. NN-OPF Model for Large Load Variation

Table 1. Generator data

UNIT	CHARACTERISTIC FUNCTION OF GENERATOR	PRODUCTION COST(RP/KWH)
SURALAYA (1)	$65.94P_1^2 + 395668.05P_1 + 31630.21$	0.138
MUARA TAWAR (8)	$690.98P_2^2 + 2478064.47P_2 + 107892572.17$	1.450
CIRATA (10)	$0 + 6000.00P_3 + 0$	1.000
SAGULING (11)	$0 + 5502.00P_4 + 0$	0.917
TANJUNG JATI (15)	$21.88P_5^2 + 197191.76P_5 + 1636484.18$	0.077
GRESIK (17)	$132.15P_6^2 + 777148.77P_6 + 13608770.9$	0.378
PAITON (22)	$52.19P_7^2 + 37370.67P_8 + 8220765.38$	0.030
GRATI (23)	$533.92P_8^2 + 2004960.63P_8 + 31630.21$	1.067

Fig. 13 -20 show the optimization results of the P and Q of each generator by using the NN- based OPF and PSO-OPF. Figs. 13, 15, 17, 18 and 20 show that the values obtained by these two methods are almost identical. For Generator Saguling (Fig. 16), the values of Qs are almost identical and those of Ps are slightly different. On the other hand, for Generator Muara Tawar and Paiton (Figs. 14 and 19) the differences between these two OPF are observed for the value of Q. Their difference can be improved by

1. Including more data for NN training.
2. Increasing the threshold value during the process of clustering the load curves.

Note that the optimized P and Q values of Generators Cirata, Saguling, Tanjungjati, Gresik, Paiton and Grati (Figs. 15, 16, 17, 18, 19 and 20) exactly coincide with the GCC. Operation under this condition is still very safe because GCCs used in our simulation include security factors.

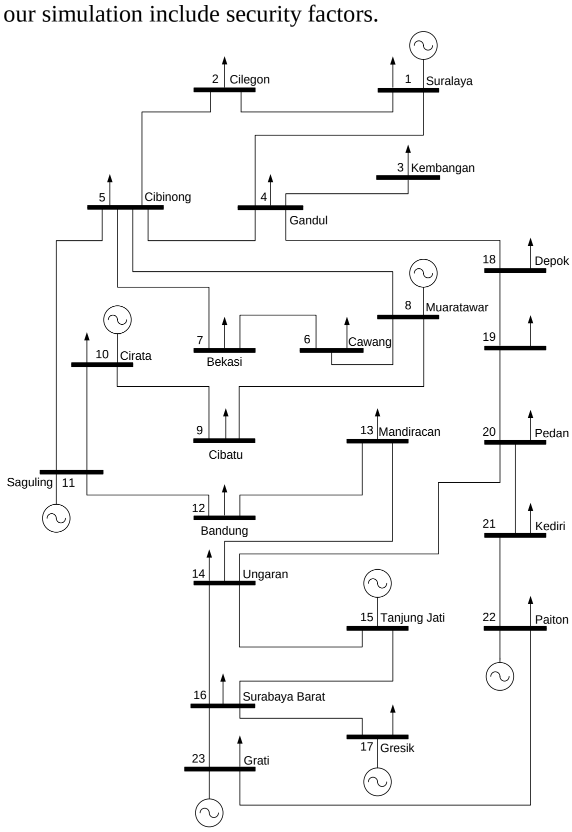

Fig. 12. 500 kV Java Bali power system

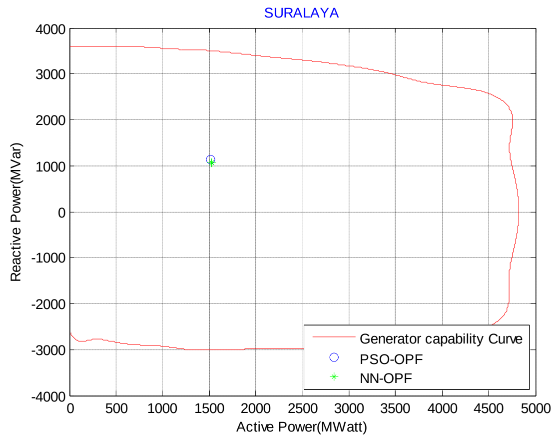

Fig.13. NN-OPF and PSO-OPF at Suralaya Generator

Table 2. Cost generation

	NN-OPF			OPF-PSO		
	P(MW)	**Q(Mvar)**	**Cost (Rp/Kwh)**	**P(Mvar)**	**Q(MW)**	**Cost (Rp/Kwh)**
Suralaya (bus 1)	1531.22	1012.57	760 491 673.1	1519.46	1145.83	753 473 072.9
Muara tawar (bus 8)	1040.00	803.08	3 432 443 589.0	1040.00	586.93	3 432 443 589.0
Cirata (bus 10)	787.27	371.89	4 723 602.4	779.82	391.93	4 678 905.3
Saguling (bus 11)	648.00	464.48	3 565 305.9	670.51	458.12	3 689 167.3
Tanjung jati (bus 15)	743.31	432.26	160 299 920.6	748.42	428.88	161 475 023.8
Gresik (bus 17)	392.71	294.73	339 180 169.7	392.21	294.94	338 740 163.4
Paiton (bus 22)	4728.85	1177.88	1 352 013 925.0	4721.22	1417.70	1 347 965 310.0
Grati (bus 23)	149.99	670.98	399 294 066.6	149.71	671.01	398 695 696.8
Total Generation	**10021.35**			**10021.35**		
Total Cost (Rp/Kwh)			**6 452 012 252.3**			**6 441 160 928.5**

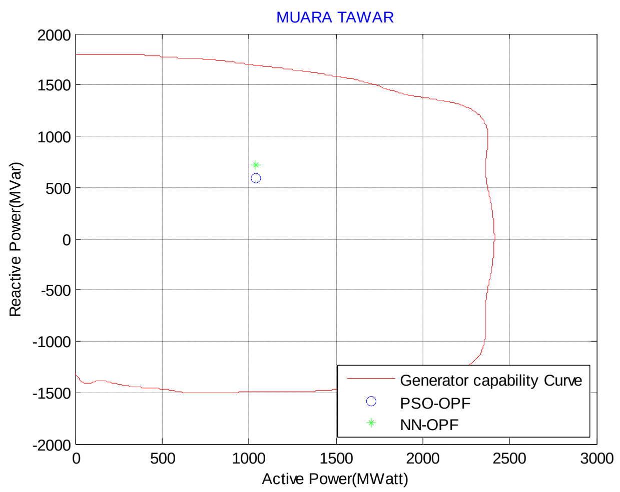

Fig.14. NN-OPF and PSO-OPF at Muara Tawar Generator

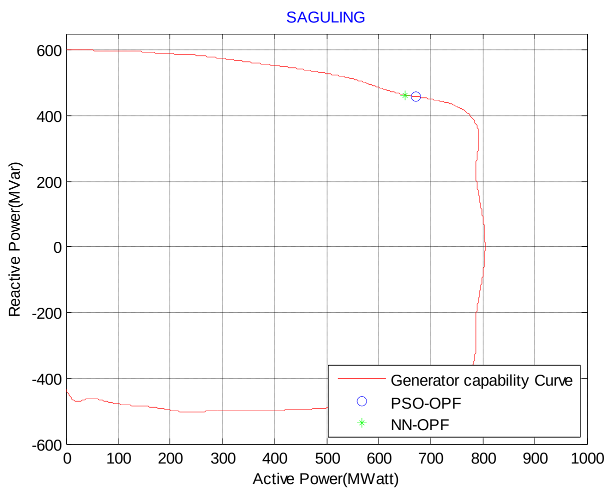

Fig.16. NN-OPF and PSO-OPF at Saguling Generator

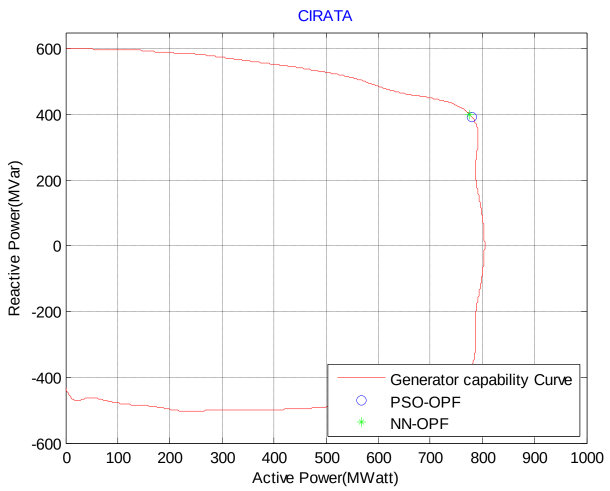

Fig.15. NN-OPF and PSO-OPF at Cirata Generator

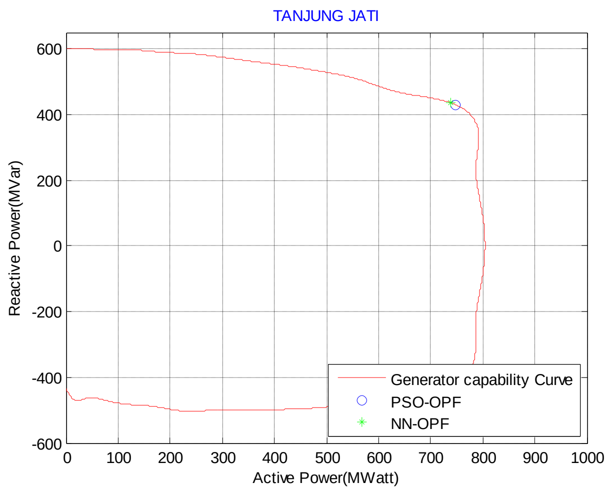

Fig.17. NN-OPF and PSO-OPF at Tanjung Jati Generator

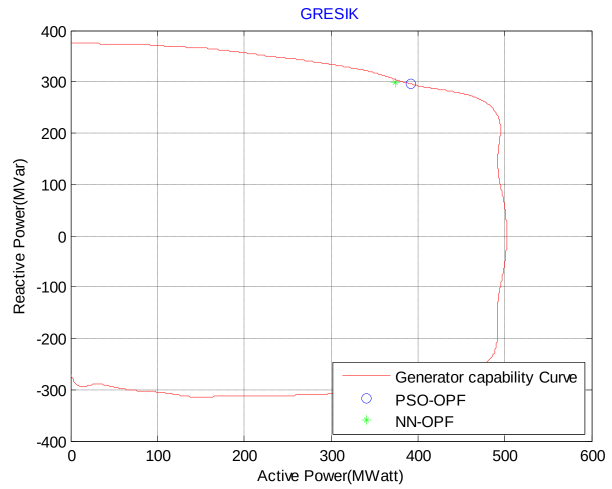

Fig.18. NN-OPF and PSO-OPF at Gresik Generator

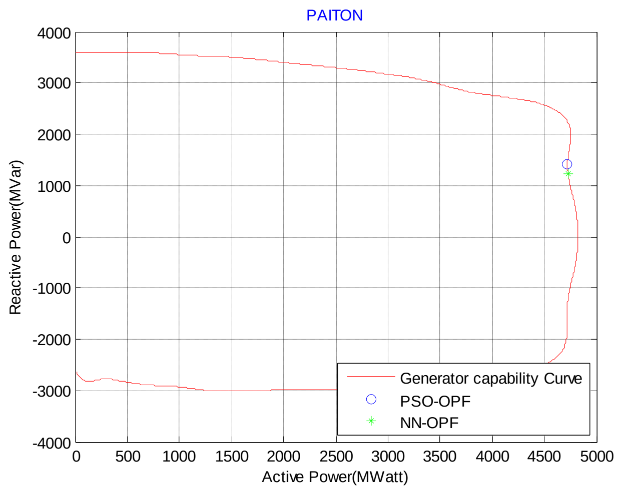

Fig.19. NN-OPF and PSO-OPF at Paiton Generator

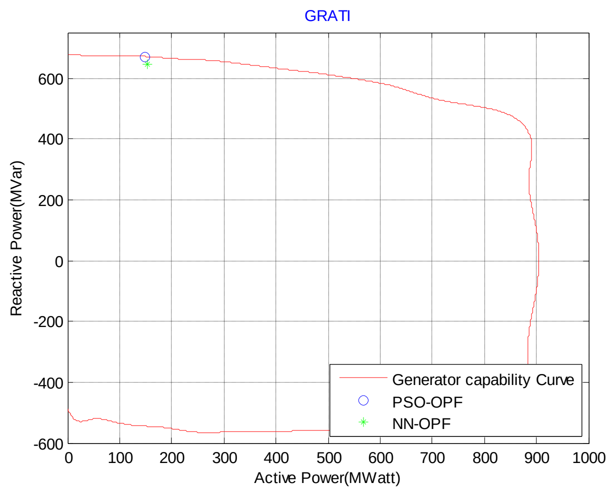

Fig.20. NN-OPF and PSO-OPF at Grati Generator

4. CONCLUSION

An NN-based OPF is proposed in this paper. The proposed OPF include three unique features. Firstly, instead of using the rectangular constraints, the more realistic GCC constraints are used in the algorithm. To overcome the mathematical difficulty in modeling a GCC, we used NN to model it. Secondly, to be able to account for various load conditions, the statistic-fuzzy load clustering method is used to classify the loads based on the patterns of load curves. A similarity index is then defined to associate the similarity among different patterns of load distribution curves. Thirdly, the proposed overall NN is trained to imitate the PSO-OPF. Therefore, the results, obtained by the proposed OPF, are very close to those by the PSO-OPF.

5. ACKNOWLEDGMENTS

Thank you for the Indonesian Government Electrical Company for supporting all the data and the Indonesian Government for supporting financial needed in this research.

6. REFERENCES

[1] Roa-Sepulveda, C.A. and B.J. Pavez-Lazo. “A solution to the optimal power flow using simulated annealing”. in Power Tech Proceedings, 2001 IEEE Porto. 2001.

[2] B. Venkatesh, M. K. George, and H. B. Gooi, "Fuzzy OPF incorporating UPFC," IEE Proceedings-Generation, Transmission and Distribution, 2004. 151(5): p. 625-629.

[3] Mori, H. and T. Horiguchi, "A genetic algorithm based approach to economic load dispatching," ANNPS '93.

[4] Yurong, W., L. Fangxing, and W. Qiulan. "Reactive power planning based on fuzzy clustering and multivariate linear regression," 2010 IEEE Power and Energy Society General Meeting.

[5] M. A. Abido, "Multiobjective particle swarm optimization for optimal power flow problem," MEPCON 2008.

[6] L. dos Santos Coelho and V. C. Mariani. "Economic dispatch optimization using hybrid chaotic particle swarm optimizer," IEEE International Conference on Systems, Man and Cybernetics, 2007.

[7] P. Jong-Bae, et al., "An Improved Particle Swarm Optimization for Nonconvex Economic Dispatch Problems," IEEE Transactions on Power Systems, 2010, 25(1): p. 156-166.

[8] L. Weibing, L. Min, and W. Xianjia. "An improved particle swarm optimization algorithm for optimal power flow," IPEMC '09.

[9] R. R. B. Aquino, et al., "Recurrent neural networks solving a real large scale mid-term scheduling for power plants," The 2010 International Joint Conference on Neural Networks (IJCNN)

[10] K. K. Swarnkar, S. Wadhwani, and A. K. Wadhwani, "Optimal Power Flow of large distribution system solution for Combined Economic Emission Dispatch Problem using Partical Swarm Optimization. in Power Systems," ICPS '09

[11] S. Panta, and S. Premrudeepreechacharn, "Economic dispatch for power generation using artificial neural network," ICPE '07.

[12] G. Zwe-Lee, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," IEEE Transactions on Power Systems, 2003. 18(3): p. 1187-1195.

[13] P. E. Onate Yumbla, J. M. Ramirez, and C.A. Coello Coello, "Optimal Power Flow Subject to Security Constraints Solved With a Particle Swarm Optimizer," IEEE Transactions on Power Systems, 2008. 23(1): p. 33-40.

[14] P. E. Sutherland, "Generator capability study for offshore oil platform," IEEE Industrial & Commercial Power Systems Technical Conference , 2009.

[15] Li, W. and Z. Xia., "Online monitoring and fault diagnosis system of Power Transformer," APPEEC 2010.

[16] N. Gunaseeli, and N. Karthikeyan. "A Constructive Approach of Modified Standard Backpropagation Algorithm with Optimum Initialization for Feedforward Neural Networks," International Conference on Computational Intelligence and Multimedia Applications, 2007.

[17] L. Wenyuan, et al., "A Statistic-Fuzzy Technique for Clustering Load Curves," IEEE Transactions on Power Systems, 2007. 22(2): p. 890-891.

[18] F.O. Karray,et al., Soft Computing and Intelligent Systems Design Pearson Education Limited 2004.

[19] Mat Syai'in , Adi Soeprijanto,, and Takashi Hiyama,, Generator Capability Curve Constraint for PSO Based Optimal Power Flow; International Journal of Electrical and Electronic Engineering, 2010. 4(6): p. 371-376..

7. APENDIX

Table 3. Network data

From Bus.	To Bus	R (pu)	X (pu)	B (pu)
1	2	0.0006264960000	0.0070087680000	0
1	4	0.0065132730000	0.0625763240000	0.005989820
2	5	0.0131333240000	0.1469257920000	0.003530571
3	4	0.0015131790000	0.0169283090000	0
4	5	0.0012464220000	0.0119750100000	0
4	18	0.0006941760000	0.0066692980000	0
5	7	0.0044418800000	0.0426754000000	0
5	8	0.0062116000000	0.0596780000000	0
5	11	0.0041113800000	0.0459950400000	0.004420973
6	7	0.0019736480000	0.0189618400000	0
6	8	0.0056256000000	0.0540480000000	0
8	9	0.0028220590000	0.0271129540000	0
9	10	0.0027399600000	0.0263241910000	0
10	11	0.0014747280000	0.0141684580000	0
11	12	0.0019578000000	0.0219024000000	0
12	13	0.0069909800000	0.0671659000000	0.006429135
13	14	0.0134780000000	0.1294900000000	0.012394812
14	15	0.0135339200000	0.1514073600000	0.003638261
14	16	0.0157985600000	0.1517848000000	0.003632219
14	20	0.0090361200000	0.0868146000000	0
15	16	0.0375396290000	0.3606623040000	0.008630669
16	17	0.0013946800000	0.0133994000000	0
16	23	0.0039863820000	0.0445966560000	0
18	19	0.0140560000000	0.1572480000000	0.015114437
19	20	0.0153110000000	0.1712880000000	0.016463941
20	21	0.0102910000000	0.1151280000000	0.011065927
21	22	0.0102910000000	0.1151280000000	0.011065927
22	23	0.0044358230000	0.0496246610000	0.004769846