Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information). 9/1 (2016), 52-58
DOI: http://dx.doi.org/10.21609/jiki.v9i1.366

PARTICLE SWARM OPTIMIZATION (PSO) FOR TRAINING OPTIMIZATION ON CONVOLUTIONAL NEURAL NETWORK (CNN)

Arie Rachmad Syulistyo[1], Dwi M J Purnomo[1], Muhammad Febrian Rachmadi[2], and Adi Wibowo[3]

[1] Faculty of Computer Science, Universitas Indonesia, Kampus Baru UI, Depok, 16424, Indonesia
[2] School of Informatics, The University of Edinburgh, 11 Crichton Street, Edinburgh EH8 9LE, United Kingdom
[3] Department Micro-Nano System Engineering, Graduate School of Engineering, Nagoya University, 1 Furocho, Chickusa Ward, 464-8603 Japan

E-mail: arie.rachmad@ui.ac.id

Abstract

Neural network attracts plenty of researchers lately. Substantial number of renowned universities have developed neural network for various both academically and industrially applications. Neural network shows considerable performance on various purposes. Nevertheless, for complex applications, neural network's accuracy significantly deteriorates. To tackle the aforementioned drawback, lot of researches had been undertaken on the improvement of the standard neural network. One of the most promising modifications on standard neural network for complex applications is deep learning method. In this paper, we proposed the utilization of Particle Swarm Optimization (PSO) in Convolutional Neural Networks (CNNs), which is one of the basic methods in deep learning. The use of PSO on the training process aims to optimize the results of the solution vectors on CNN in order to improve the recognition accuracy. The data used in this research is handwritten digit from MNIST. The experiments exhibited that the accuracy can be attained in 4 epoch is 95.08%. This result was better than the conventional CNN and DBN. The execution time was also almost similar to the conventional CNN. Therefore, the proposed method was a promising method.

Keywords: *deep learning, convolutional neural network, particle swarm optimization, deep belief network*

Abstrak

Jaringan syaraf tiruan menarik banyak peneliti dewasa ini. Banyak universitas-universitas terkenal telah mengembangkan jaringan syaraf tiruan untuk berbagai aplikasi baik kademik maupun industri. Jaringan syaraf tiruan menunjukkan kinerja yang patut dipertimbangkan untuk berbagai tujuan. Meskipun begitu, kinerja dari jaringan syaraf tiruan merosot dengan signifikan untuk masalah-masalah yang kompleks. Untuk menyelesaikan masalah tersebut di atas, banyak penelitian yang dilakukan untuk meningkatkan kinerja dari jaringan syaraf tiruan standar. Salah satu pengembangan yang menjanjikan untuk jaringan syaraf tiruan pada kasus yang kompleks adalah metode *deep learning*. Pada penelitian ini, diusulkan penggunaan metode Particle Swarm Optimization (PSO) pada Convolutional Neural Networks (CNNs), yang merupakan salah satu metode dasar pada deep learning. Penggunaan PSO dalam proses pelatihan bertujuan untuk mengoptimalkan hasil vektor solusi pada CNN, sehingga dapat meningkatkan akurasi hasil pengenalan. Data yang digunakan dalam penelitian ini adalah data angka yang berasal dari MNIST. Dari percobaan yang dilakukan akurasi yang dicapai dengan 4 iterasi adalah 95,08%. Hasil ini lebih baik dari CNN konvensional dan DBN. Waktu eksekusinya juga mendekati CNN konvensional. Oleh karena itu, metode yang usulkan adalah metode yang menjanjikan.

Kata Kunci: *deep learning, convolutional neural network, particle swarm optimization, deep belief network*

1. Introduction

In recent years, many researchers conducted studies on machine learning with deep hierarchical architecture. The term deep hierarchical learning was introduced by Hinton et al. [1]. They proposed a method to transform high dimensional data into low dimension data. They employed multilayer neural network with a small middle layer to reconstruct the input vector. Today, machine learning with deep hierarchical learning is named as deep learning.

The concept of deep learning is derived from neural network research, therefore deep learning regarded as a "*new-generation of neural networks*" [2]. Deep learning is research intersection between many areas, such as neural networks, artificial intelligence, pattern recognition, signal processing, optimization, and graphical model. Feedforward neural networks or Multilayer Perceptron (MLPs) with many hidden layers, which is often called Deep Neural Networks (DNN), is a good example of a model with deep architecture. Deep learning demonstrated impressive results and has been applied in several fields, like object recognition, computer vision, voice search, conversational speech recognition, and language processing.

From a wide variety descriptions that exist, deep learning generally has two aspects [2]. The first is deep learning is a model consisting of multiple layers of nonlinear information process-ing. The second is deep learning is a supervised or an unsupervised learning method to represent the features of the bottom layer to the top layers.

Several variations of deep learning are continually being researched and many of them have been applied into some machine learning tasks. In many cases, deep learning exhibited significant improvement on results, compared to previous conventional methods. There are plenty of deep learning algorithms which have been developed such as, Deep Neural Networks (DNNs) [1], Deep Boltzmann Machines (DBMs) [3], Recurrent Neural Networks (RNNs) [4], and Deep Auto encoders [5].

From many deep learning methods, there are 3 basic models that underlie many of the deep learning methods, i.e. Deep Belief Networks (DBNs) [6], Convolutional Neural Networks (CNNs) [7], and Stacked Auto Encoder (SAE) [8]. These three models are the most prominent and become building blocks for many deep learning methods, such as Multiresolution DBN (MrDBN) [9], an extension of the DBN or Scale-Invariant Convolutional Neural Network (SiCNN) [10] developed based on the CNNs model.

In this research, we focus handwritten digit recognition problem based on data from MNIST. From three basic models, we empowered CNNs method as our basic algorithm. CNN was employed due to its high accuracy on MNIST datasets [11].

CNNs is a type of feed forward neural network inspired by the structure of visual system. CNN consists of many neurons that have weights and biases, where each neuron receives several inputs and perform dot products. In terms of architecture, CNNs composed of one or more convolutional layers with subsampling stages and one or more fully connected layers as found in a standard multi-layer neural networks.

Even though standard CNN has shown considerable accuracy, there are still a lot of space for improvement. To ameliorate performance of CNN in recognition task, we used PSO to optimize output vector from CNNs. The utilization of PSO is due to its powerful performance on the optimization problems.

PSO in an optimization method developed by Eberhart and Kennedy [12]. This method is inspired by social behavior of animals that do not have a leader in their group. PSO consists of a swarm of particles, where the particles represent a potential solution.

To assess our proposed method, we compare results obtained from proposed method with other existing algorithms results. The existing algorithms used for comparison are the original CNNs and Deep Belief Networks (DBNs). The performance criteria used in this research are error and accuracy.

The remaining of this paper is organized as follows: Section 2 describes the basic theory of DBN, CNN, and PSO. Experiment setup and results obtained in the comparison study presented in Section 3. The conclusions of this paper are given in Section 4.

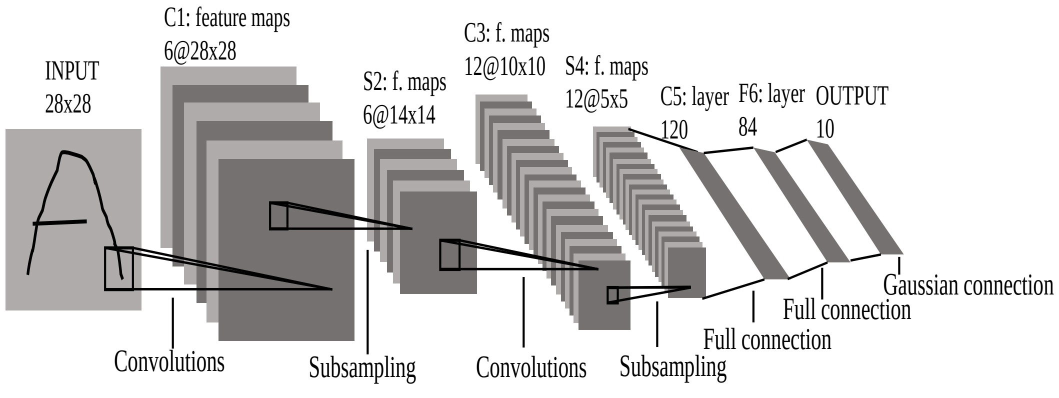

Figure 1. The original Convolutional Neural Networks (CNN) architecture [7]

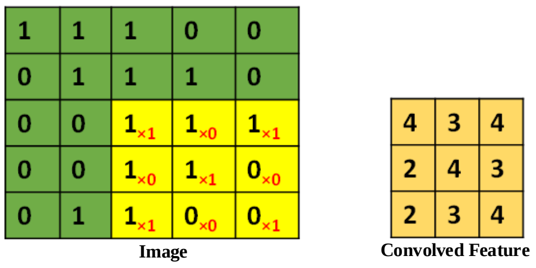

Figure 2. Convolution operation

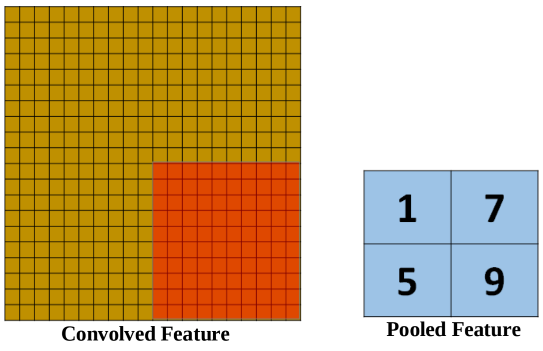

Figure 3. Illustration of pooling process

2. Methods

Convolution Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) is inspired by cat's visual cortex [12]. CNN generally consists of 3 layers which are convolutional layer, subsample/pooling layer, and fully connected layer. Convolution layer shares a lot of weights whereas pooling layer performs subsampling function to resulted output from convolution layer and reduce data rate of the layer below. The outputs from pooling layer are used as an input to several fully connected layers. Figure 1 shows the architecture of CNNs [7], and the convolution operation is illustrated on Figure 2. Convolved feature that is generally called feature maps is the result of convolving the filter/kernel on dataset.

The convolution process can be written by the following equation(1):

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \quad (1)$$

where *I* is an input image, *K* is kernel/filter used in convolution process, *m* is row of image, and *n* is column of image. The subsample or pooling is the process to reduce feature map. The concept of pooling process almost equal to convolution process that is convolving filter on input data. However, the differences pooling process on shifting filter that does not overlap on each filter compare with convolution process. The pooling illustration can be seen on Figure 2.

Deep Belief Networks (DBNs)

Deep Belief Networks (DBNs), which was introduced by Hinton, et al. [6], is a probabilistic graphical model consisting of multiple layers with hidden variables. DBNs are trained using greedy layer wise algorithm which can optimize the weight of DBNs at the time complexity that linear to the size and depth of the network. DBNs are trained to extract deep hierarchical representation on the input data. DBNs are composed of a number of Restricted Boltzmann Machines (RBMs), a special kind of Boltzmann Machines, that consisting a layer of visible unit and a layer of hidden units, with undirected and symmetrical connections between visible and hidden layer, but there is no connection among units in the same layer. Illustration of RBM models can be seen in Figure 4.

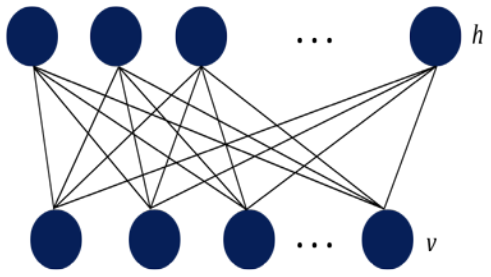

Figure 4. Restricted Boltzmann Machines (RBMs)

The lower layer *v* is the visible layer, and the top layer *h* is the hidden layer, where these two layers are stochastic binary variables. The weights between the visible layer and the hidden layer (*W*) are undirected. In addition each neuron has a bias.

The join distribution function p(v,h) of the visible units v and the hidden units are defined in equation(2) in the form of an energy function.

$$p(v,h) = \frac{\exp(-E(v,h))}{Z} \quad (2)$$

where *Z* is the partition function and given by summing all possible pairs of visible and hidden units as shown by equation(3).

$$Z = \sum_{v,h} \exp(-E(v,h)) \quad (3)$$

A joint configuration of visible and hidden units has an energy that given by equation(4).

$$E(v,h) = -\sum_{i=1} \mathrm{a}_i v_i - \sum_{j=1} \mathrm{b}_j h_j - \sum_{i,j} \mathrm{v}_i h_j w_{ij} \quad (4)$$

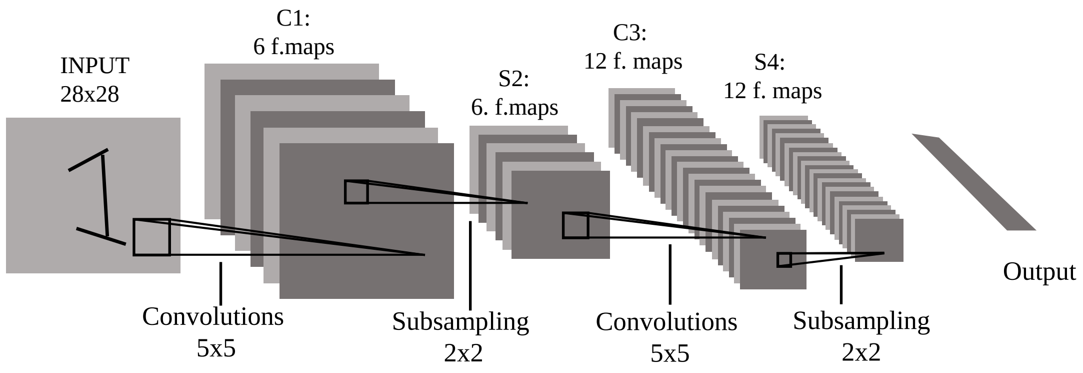

Figure 5. CNN Architecture in the Proposed Method

where v_i is binary state of visible unit i, h_j is binary state of hidden unit j, a_i is bias in visible unit, b_j is bias in hidden unit, and w_{ij} is the weight between visible and hidden units. The update rule for RBM weights are:

$$\Delta W_{ij} = E_d(v_i h_j) - E_m(v_i h_j) \qquad (5)$$

where $E_d(v_i h_j)$ is the expectation in the training data and $E_m(v_i h_j)$ is the same expectation that defined by the model. RBMs are trained using *Contrastive Divergence* (CD) algorithm to approximate the expected value.

Particle Swarm Optimization (PSO)

Particle swarm optimization (PSO) algorithm is one of evolutionary algorithm which was firstly proposed in 1995 [13]. PSO has widely been employed in miscellaneous field, to cite an instance swarm robot for odour source localization purpose [14, 15].

PSO algorithm consist several consecutive steps. First of all, initialization which randomly select the particles as searching agents (x) as well as the velocities (v). Secondly the particles then inserted into cost function to find local bests (p_{best}) and global best (g_{best}). Local best is defined as the location on which the cost is the smallest for each particle. Meanwhile, global best is the location on which the cost is smallest among the local bests. Thirdly, the particles are updated by empowering equation(6) and equation(7).

$$v_{n+1} = v_n + c_1 r_1 (p_{best} - x_n) + c_2 r_2 (g_{best} - x_n) \qquad (6)$$

$$x_{n+1} = x_n + v_{n+1} \qquad (7)$$

Where c_1 and c_2 are the constants, r_1 and r_2 are random numbers, and n is iteration. The algorithm of PSO can be written as follows:

Algorithm 1: Particle Swarm Optimization

```
% Initialization
Number_Interation_PSO;
Number_PSO_Swarm;
Determine gbest from PSO_Swarm;
Determine pbest from PSO_Swarm;
calculate fitness_old;

for 1 to Number_Interation_PSO do
 for 1 to Number_of_Particle do

    %Update Velocity
    v(n)_t = v(n)_t+1 + c1*r1.*(pbest-
    x(n) +c2*r2.*(gbest-x(n))

    %Update position
    x(n+1) = x(n) + v(n)_t

    %Evaluate the objective function
    fitness_new = f(x(n+1))
    if (fitness_new< fitness _old)
     fitness _old = fitness _new;
     x = x(n+1);
     else
     fitness_new = fitness _old;
     x(n+1) = x;
    end if
 end for;
 Index = min (fitness_new);
 pbest = x(index);
end for;
```

Proposed Method

Figure 5 shows the CNNs architecture used in the proposed method, where it is consist of an input image that will be processed using 6 convolution kernel with size 5x5 pixels, 6 sub-sampling kernel with size 2x2 pixels, 12 convolution kernel with size 5x5 pixels, 12 subsampling kernel with size 2x2 pixels and the last layer is the vector output of CNN. The proposed method process can be seen on Figure 6. In Figure 6 Y denotes the condition is met, whereas N represent the condition is not met.

PSO in this study would optimize the output vector. The output vector would be augmented by δx to acquire better value. The value of δx itself is the value which would be optimized by PSO. To calculate the fitness function, the root mean square error between output vectors after augmented with δx and the true output would be employed.

Generally, the process of the proposed method consists of several steps as shown below: 1) the first step is initializing the learning rate of the CNNs with the value is 1 based on the experiment. Batch size of CNNs is 50, the number of CNNs epoch in the range of 1 to 4, PSO iteration is 10. The convergence status of PSO is used to check the convergences of PSO, if the error value has not changed for three iterations, then the PSO is considered as convergent; 2) after setting up the experiment, the next step is run CNNs training process, where the detail of the process can be seen in section 2.1.

The result of CNNs is vector output that will be optimizing using PSO algorithm. PSO optimization in this study serves to make the value of loss function on CNN becomes minimal; 3) the output vector will be update if the solution of swarm has less error compare with old vector output; 4) the PSO will run as long as the iteration number of PSO and the convergence solution have not fulfilled; 5) after the CNN Training, the model will be tested with testing data that consist of 10000 data; 6) the result of CNN test is accuracy of CNN, it represent how precise of the CNN model can predict the actual value of testing dataset.

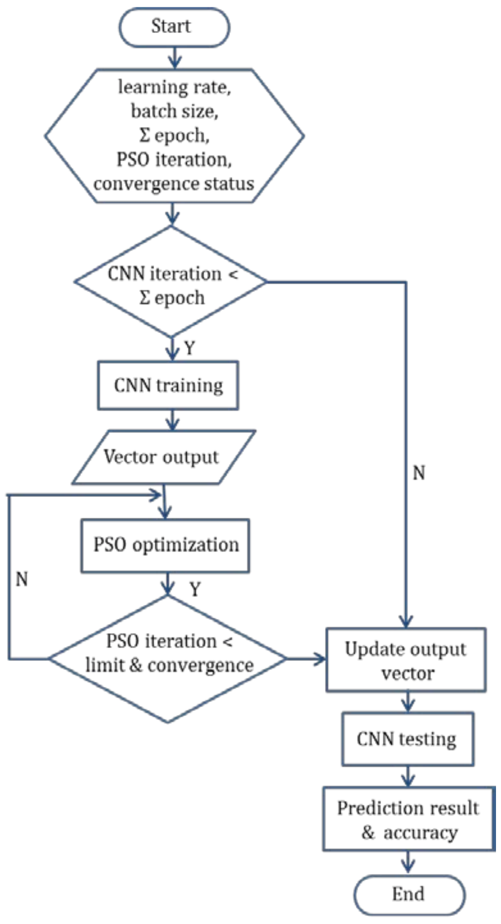
Start
learning rate, batch size, Σ epoch, PSO iteration, convergence status
CNN iteration < Σ epoch
Y
N
CNN training
Vector output
PSO optimization
N
Y
PSO iteration < limit & convergence
Update output vector
CNN testing
Prediction result & accuracy
End

Figure 6. Flowchart of the proposed method

3. Results and Analysis

Dataset

In this study we use the handwritten digits data taken from MNIST database. This dataset has 28 x 28 pixel dimension and consist of 70000 data, in which 60000 data used for training and the rest data used for testing.

Experiment Result

In this chapter we will show the experiment result on deep learning method, such as Convolutional Neural Network, Deep Belief Network, and Convolutional Neural Network Optimize with Simulated Annealing (SA) [16] and also compare the performance with proposed method. The experiment use handwritten Digit dataset from MNIST. And the running time of all the deep learning method will compare each other to know how long the deep learning method can predict the test dataset.

Experiment on Handwritten Dataset

The overall experimental results on error and accuracy can be seen in Figure 7 and Table 1 respectively. The results of experiments showed the proposed method has a better accuracy than another deep learning method. Although the difference accuracy value is not high but it important be-cause the CNN does not need a lot of training epoch to get good accuracy. CNNPSO has better accuracy than CNNSA, it could be the advantages of PSO fast searching ability with minimum iteration of CNN. The error also exhibit the similar behavior. These due to the fact that PSO is a powerful algorithm for optimizing. Thus if com-pared to simulated annealing in particular, it has better performance.

As shown in Table 1, with only 4 epochs the accuracy of CNNPSO has reached 95.08%. This value was only slightly different from simulated annealing optimized CNN (95.19%). Meanwhile, CNN and DBN occupy the two last place with 94.81% and 91.14% respectively.

In execution time, DBN has fastest execution time, which architecture 50 hidden node and 25 hidden layer, because on training process using contrastive divergence that very fast [17].

CNNPSO consume longer time than CNN. This due to the addition iteration in the PSO algo-

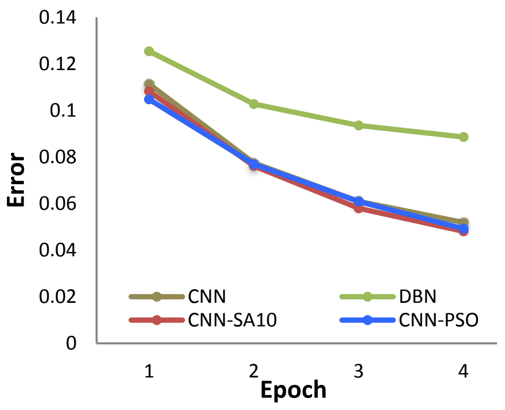

Figure 7. Error comparison of several algorithm

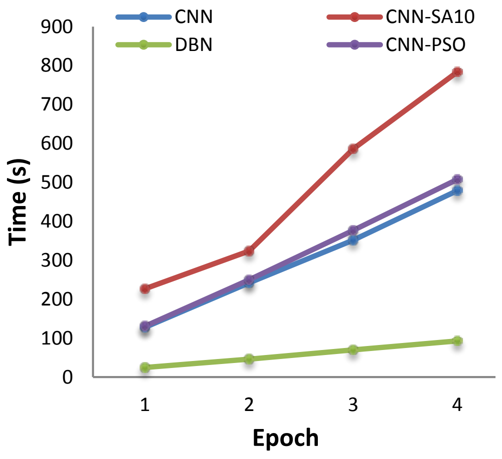

Figure 8. Time Comparison Based On Epoch Value

TABLE 1
ACCURACY PERFORMANCE ON ALL DEEP LEARNING METHOD

Method	Number of Epoch			
	1	2	3	4
CNN	88.87	92.25	93.9	94.81
DBN	87.46	89.72	90.64	91.14
CNN-SA10	89.18	**92.38**	**94.2**	**95.19**
CNN-PSO	**89.52**	92.31	93.91	95.08

rithm as well as the particles calculation. Nevertheless, if compared to CNNSA10, CNNPSO was still faster.

4. Conclusion

Based on the experiment result that have been conducted it can be conclude that the proposed method CNNPSO has good accuracy. The considerable accuracy (95.08%) was attained with only 4 epoch. Moreover the proposed method exhibited better performance than CNN, DBN. Even though its accuracy is lower than CNNSA, to obtain the accuracy nearby, CNNPSO consumed shorter time. If compared to the conventional CNN, CNN-PSO consumed only slightly longer time. However, it has to be improved. The improvement can be focused on how to give restricted range on delta x so the proposed method get optimal vector output faster.

Acknowledgement

This work was supported by Directorate Research of Universitas Indonesia funding in 2015. The title of the research is Laboratory Infra-structure. This grant number is 1831/UN2.R12/ HKP.05.00/2015.

References

[1] G. E. Hinton and Ruslan R. Salakhutdinov, "Reducing the dimensionality of data with neural networks." *Science Journals* 313.5786 (2006): 504-507.

[2] L. Deng and D. Yu, “Deep Learning: Methods and Applications”, Foundations and Trends® in Signal Processing, Vol. 7, Nos. 3–4 (2013) 197-387.

[3] R.R. Salakhutdinov and G. E. Hinton, “Deep Boltzmann Machines”, *In Proceeding of International Conference of Artificial Intelligence and Statistics*, vol. 12, 2009.

[4] J. Martens and I. Sutskever, “Learning Recurrent Neural Networks with Hessian-free Optimization.” *In Proceedings of International Conference on Machine Learning (ICML)*, 2011.

[5] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, “Exploring Strategies for Training Deep Neural Networks.” *J. Machine Learning Research*, vol. 10, pp. 1-40, 2009.

[6] G.E. Hinton, S. Osindero, and Y.W. Teh, “A Fast Learning Algorithm for Deep Belief Nets”, *Neural Computation*, vol. 18, no. 7, pp. 1527-1554, 2006.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based Learning Applied to Document Recognition”. *In Proceedings of the IEEE*, 86:2278–2324, 1998.

[8] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” *in Advances in Neural Information Processing Systems 19* (NIPS’06), (B. Scholkopf, J. Platt, and T. Hoffman, eds.), pp. 153–160, MIT Press, 2007.

[9] Y. Tang and A.R Mohamed, “Multiresolution Deep Belief Networks”, *In Proceeding of Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS) 2012*, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX, 2012.

[10] Y. Xu, T. Xiao, and J. Zhang, Scale-Invariant Convolutional Neural Networks, *in arXiv*: 1411.6369v1 [cs.CV], 2014.

[11] Ciresan, Dan; Meier, Ueli; Schmidhuber, Jürgen, Multi-column deep neural networks for image classification, IEEE Conference on Computer Vision and Pattern Recognition, 2012.

[12] Hubel, D. and Wiesel, T. (1968). Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology (London), 195, 215–243.

[13] J. Kennedy and R. Eberhart, "Particle swarm optimization", *In Proceedings of the IEEE International Conference Neural Network (ICNN),* Nov. 1995, vol. 4 pp. 1942-1948.

[14] W. Jatmiko, K. Sekiyama, and T. Fukuda, A mobile robots pso-based for odor source localization in dynamic advection-diffusion environment, in Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on. IEEE, 2006, pp. 4527–4532.

[15] W. Jatmiko, A. Nugraha, R. Effendi, W. Pambuko, R. Mardian, K. Sekiyama, and T. Fukuda, Localizing multiple odor sources in a dynamic environment based on modified niche particle swarm optimization with flow of wind, WSEAS Transactions on Systems, vol. 8, no. 11, pp. 1187–1196, 2009.

[16] L. M. R. Rere, M. I. Fanany, A. M. Arymurthy, Simulated Annealing Algorithm for Deep Learning, Procedia Computer Science, no. 72, pp. 137–144, 2015.

[17] Hinton, Geoffrey E. and Osindero, Simon and Teh, Yee-Whye, A Fast Learning Algorithm for Deep Belief Nets, Journal Neural Computing,no.7, pp 1527-1554, 2006.