2022 International Symposium on Nonlinear Theory and Its Applications,
NOLTA2022, Virtual, December 12-15, 2022

Learning a simple multilayer perceptron with PSO

Riku Takato† and Kenya Jin'no†

†Graduate School of Integrative Science and Engineering, Tokyo City University
1-28-1 Tamazutumi, Setagaya, Tokyo 158-8557, Japan
Email: g2281436@tcu.ac.jp, kjinno@tcu.ac.jp

Abstract—In this article, we attempt to learn the parameters of a multi-layer perceptron (MLP) using the particle swarm optimization (PSO) method which is one of the approximate solution methods for optimization problems without using the derivative information of the objective function. We use the gradient method and PSO to learn to classify a linearly inseparable data set with an MLP in the middle layer with a small number of neurons. As a result, we experimentally confirm that PSO outperforms gradient-based learning.

1. Introduction

In recent years, multilayer perceptrons (MLP) have attracted attention for their excellent function approximation ability, and MLP is used to solve many problems in image processing and natural language processing. MLP consists of some layers, and each layer consists of plural neurons that have nonlinear activation functions. The function approximation ability of MLP is acquired by learning weight parameters using input-output data for training. Learning of the weight parameters is accomplished by minimizing the loss function, which is defined as the difference between the output of MLP and the expected output, as in other machine learning methods. Generally, the loss function of MLP is non-convex. The optimization of such a non-convex function is performed by the gradient method, which iteratively searches for a solution based on the gradient of the loss function. In particular, the stochastic gradient descent method is known to be effective, and current MLP training is based on the stochastic gradient descent method. However, the stochastic gradient descent method does not guarantee convergence to the minimum solution of the non-convex loss functions and is sensitive to the initial parameter values of the loss functions. Also, it is difficult to reach the optimal solution of the objective function by the gradient method because there are countless local solutions, plateaus, and saddle points. However, it has been suggested[2] that local solutions exist in the vicinity of the optimal solution, and it has also been reported[3] that the stochastic gradient method can converge to the optimal solution if the network consists of a sufficiently large number of neurons.

ORCID iDs Riku Takato: 0000-0003-4563-4202, Kenya Jin'no: 0000-0002-0431-5769

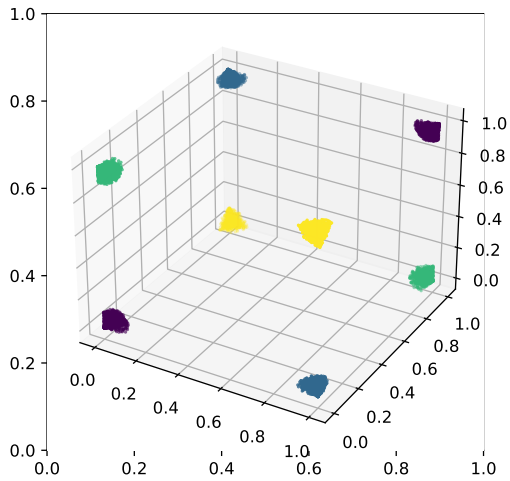

Figure 1: 3D ExOR: The input is 3-dimensional, and the number of labels is 4.

Since these results suggest that gradient-based learning is successful in systems composed of many neurons, recent MLP has become deep and the number of constituent neurons has increased. As the number of neurons increases, the amount of memory required increases and the learning time becomes longer.

On the other hand, it is essential from the edge computing point of view to realize excellent recognition functions by MLP with a small number of neurons. Therefore, to converge to the optimal solution of the objective function even with a small number of neurons, we attempt to learn the parameters of the MLP with PSO[1], which does not require information on the derivative of the objective function.

2. Learning by PSO

PSO is an algorithm[1] that searches for the parameters that give the optimal value of the objective function in the parameter space with multiple particles exchanging information with each other and is one of the meta-heuristic solution methods that do not require the derivative of the

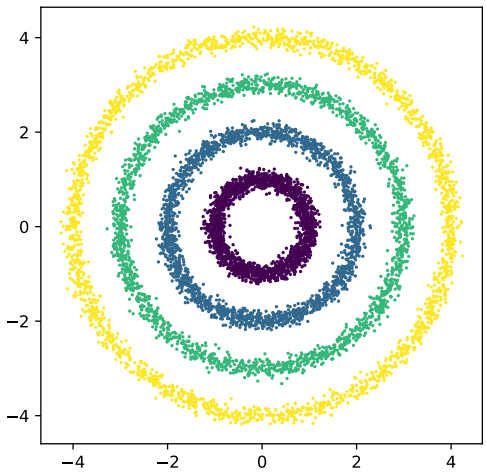

Figure 2: Quadruple Circles: The input is 2-dimensional, and the number of labels is 4.

objective function. Such PSO is used to learn the parameters of MLP. In this article, we focus on some classification problem that is not linearly separable. We consider the following three types of classification problems: 1) 3D ExOR, 2) Quadruple Circles, and 3) MNIST.

2.1. 3D ExOR

The 3D ExOR problem is a linearly inseparable problem that classifies eight different inputs in three dimensions into four classes, as shown in Fig. 1.

2.2. Quadruple Circles

The Quadruple Circles problem is a linearly inseparable problem that classifies two-dimensional inputs into four classes as shown in Fig. **??**.

2.3. MNIST

Based on reducing the image size of the 28x28x1 input image data to a 2-dimensional latent variable by convolutional operations, we make a CNN system that classifies 10 different classes from this 2-dimensional latent variable to 10 outputs using the Softmax function. This CNN is trained on 10 different handwritten digit images from MNIST [5] to a recognition accuracy of up to 100% for the training data. After training, the two latent variables in the layer consisting of only two neurons before the Softmax layer are colored for each handwritten input digit image to obtain the distribution as shown in Fig. **??**. As is clear in Fig. 3, MNIST handwritten digit images can be classified into clusters in a two-dimensional latent space. Therefore, we

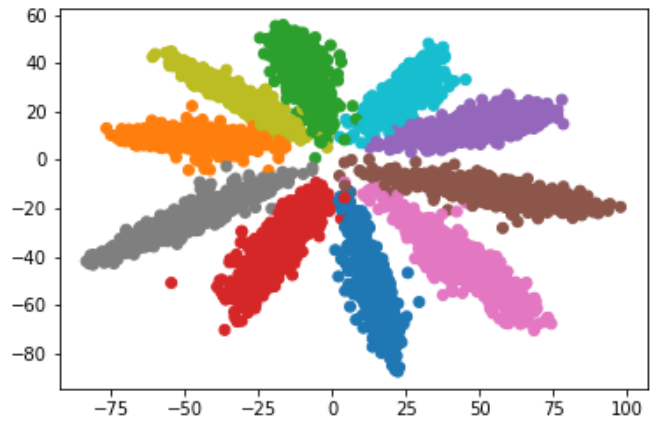

Figure 3: Quadruple Circles: The input is 2-dimensional, and the number of labels is 4.

named the problem of classifying this 2-dimensional data into 10 classes as MNIST in this article.

2.4. MLP

For the three types of problems described above, we trained the MLP with PSO so that it could perform classification.

For the 3D ExOR problem, the MLP used was 3 neurons in the input layer, 2 neurons in the middle layer, and 4 neurons in the output layer, with the ReLU function as the activation function in the middle layer and the Softmax function as the activation function in the output layer.

For the Quadruple Circles problem, the ReLU function was used for the activation function in the middle layer and the Softmax function for the activation function in the output layer for 2 neurons in the input layer, 2 neurons in the middle layer, and 4 neurons in the output layer.

For the MNIST problem, 2 neurons in the input layer and 10 neurons in the output layer, using the Softmax function as the activation function in the output layer.

The coupling between each layer was trained by PSO or gradient method.

3. PSO

PSO is one of the most famous meta-heuristic optimization methods. The original PSO have been propose by Eberhart and Kennedy in 1995.[1]. The original PSO is described by the following equation.

$$\begin{aligned} V_i(t+1) &= wV_i(t) + c_1R_1(t)V_i^{p_a}(t) + c_2R_2(t)V^{g_a}(t) \\ X_i(t+1) &= X_i(t) + V_i(t+1) \\ V_i^{p_a}(t) &= Pbest_i(t) - X_i(t) \\ V^{g_a}(t) &= Gbest(t) - X_i(t) \end{aligned} \tag{1}$$

where $X_i(t)$ and $V_i(t)$ denote the location and the velocity of the i-th particle at t-th iteration, respectively. $Gbest(t)$ is the global best, which is the parameter information that gives the best value of the output accuracy within the swarm at t-th iteration, and $Pbest_i(t)$ is the personal best, which is the parameter information that gives the best value of the output accuracy for the i-th particle at t-th iteration. $V_i^{p_a}(t)$, $V^{g_a}(t)$ are vectors toward $Pbest_i(t)$ and $Gbest(t)$, respectively. w represents an inertia parameter. c_1 and c_2 are acceleration parameters, and $R_1(t)$ and $R_2(t)$ are time-variant random number parameters.

PSO is a very simple system, but it can efficiently search for the optimal solution without gradient information of the objective function. In practice, however, it is known that although the search to the vicinity of the optimal solution is very fast, it is difficult to improve the accuracy of the solution after that.

To solve these problems in this paper we propose two types of PSO. One is to perturb the positional information of each particle, and the other is to add noise to the normal distribution before giving the velocity. These methods are aimed at improving the ability of each particle to search in the neighborhood of the best solution it has found, and at solving the problem of all particles converging at the end of the search, which limits the scope of the search.

3.1. PSO_dist

To prevent premature convergence of the PSO, a perturbation is applied to particles of the PSO[4]. The perturbation is given by Eq. (2).

$$\begin{aligned} V_i(t+1) &= wV_i(t) + c_1R_1V_i^{p_a}(t) + c_2R_2V^{g_a}(t) \\ X_i(t+1) &= X_i(t) + V_i(t+1) + A(t+1)\cos\theta_i(t+1) \\ \theta_i(t+1) &= \theta_i(t) + \gamma\cos\theta_i(t) + C \\ A(t+1) &= A(t) + B \end{aligned} \tag{2}$$

where $\theta(t)$ represents the internal state variable and $A(t)$ is a function that controls the magnitude of the micro-perturbation. B is a parameter that controls the magnitude of tangential motion so that $A(t)$ increases over time but does not become too large.

3.2. PSO_q

As another way to prevent premature convergence of the PSO, we consider adding perturbations to the velocity. The perturbation is given by Eq. (3).

$$\begin{aligned} V_i(t+1) &= wV_i(t) + c_1R_1V_i^{p_a}(t) + c_2R_2V^{g_a}(t) \\ X_i(t+1) &= X_i(t) + R_3 + V_i(t+1) \\ R_3 &\sim N(0, 0.0625) \end{aligned} \tag{3}$$

where R_3 is a random parameter that follows a normal distribution.

We observe the effect of adding normally distributed noise before giving velocity.

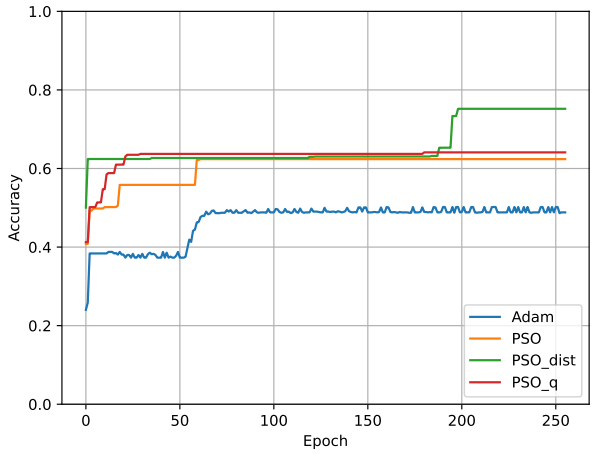

Figure 4: Change in output accuracy at each training step when using "3D ExOR"

4. Experiments

To compare the results of applying PSO and the normal gradient method for training MLP we carry out three experiments by using above problems. First, we experiment with 3D ExOR. The accuracy results for each epoch during learning are shown in Fig. 4. The blue line represents the results of learning with the gradient-based method (adam). The remaining three graphs are the results of training MLP with PSO. The orange line represents the results of regular PSO, the green line represents the results of PSO_dist, and the red line represents the results of PSO_q. PSO produced the best results in this experiment. The gradient method resulted in no update in accuracy because it may be initially trapped in the local solution.

Next we carry out the experiment with Quadruple Circle. The results of the experiment are shown in Fig. 5. In this case, PSO gives the best results, too. We can observe some oscillations in the results with the gradient method, but the accuracy of the solution is not significantly updated.

Finally, we carry out the experiment using MNIST. This problem can be viewed as learning only the final stage part of the MNIST handwritten digit recognizer in the CNN. The training CNN that created this dataset was able to train this part using the gradient method, and the accuracy, in this case, is 100%. Again, PSO produced the best results in this case. However, the gradient method did not achieve 100% accuracy when this part of the dataset was extracted and trained with the gradient method. This point needs to be clarified in the future, as it has not been fully discussed.

5. Conclusions

We used PSO to learn the parameters of an MLP consisting of a small number of neurons. As a result, we con-

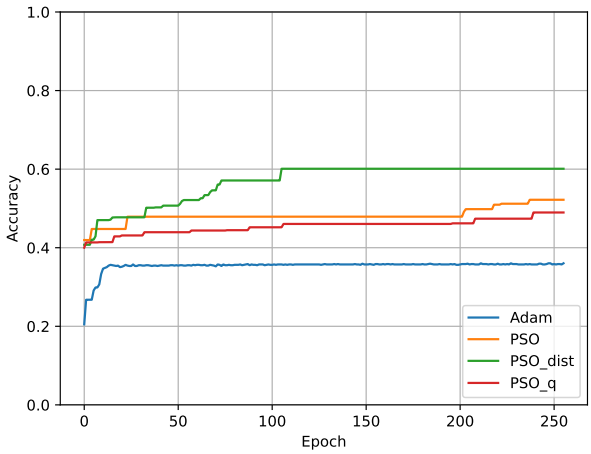

Figure 5: Change in output accuracy at each training step when using "Quadruple Circle"

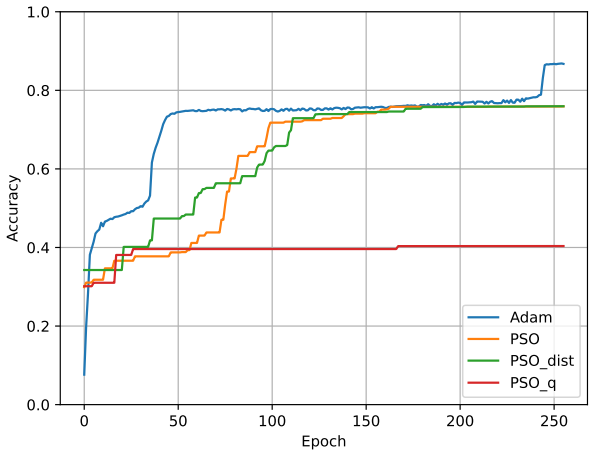

Figure 6: Change in output accuracy at each training step when using "MNIST"

firmed that PSO_dist can learn the parameters of neural networks better than the conventional gradient method such as adam when the number of neurons in the hidden layer is small and there are many local solutions to the loss function. Also, we confirmed that PSO can improve by giving perturbation and using loss function value. Although we have focused on the simplest two-class classification problems, our next task is to apply PSO to multi-class classification problems and confirm its performance.

Acknowledgments

This work was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research (C) Number: 20K11978. Part of this work was carried out under the Cooperative Research Project Program of the Research Institute of Electrical Communication, Tohoku University.

References

[1] J. Kennedy, R. Eberhart, "Particle Swarm Optimization". Proc. ICNN 1995, pp. 1942–1948, 1995. doi:10.1109/ICNN.1995.488968

[2] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, Y. LeCun, "The Loss Surfaces of Multilayer Networks," Proc. AISTATS 2015, pp. 192-204, 2015. https://arxiv.org/abs/1412.0233

[3] Z. A-Zhu, Y. Li, Z. Song, "A Convergence Theory for Deep Learning via Over-Parameterization," Proc. ICML, PMLR vol. 97, pp. 242-252, 2019. https://arxiv.org/abs/1811.03962

[4] K. Jin'no, "Analysis of particle swarm optimization by dynamical systems theory," NOLTA, vol. 12, no. 2, pp. 118-132, 2021. https://doi.org/10.1587/nolta.12.118

[5] THE MNIST DATABASE of handwritten digits, http://yann.lecun.com/exdb/mnist/