EDA based Deep Neural Network Parameter Optimization

Qingyang Xu*
School of Mechanical, Electrica & Information Engineering
Shandong University
Weihai, Shandong 264209, China
qingyangxu@sdu.edu.cn

Anbang Liu
School of Mechanical, Electrica & Information Engineering
Shandong University
Weihai, Shandong 264209, China
keithliumail@gmail.com

ABSTRACT

Deep neural network has been applied in kinds of areas due to the excellent performance. The capacity of deep neural network relies on the parameter training algorithm which is always based on the gradient information. However, for deep neural network, it is harder and harder for training due to depth of the neural network and a large number of parameters. Therefore, kinds of techniques are proposed to cope with this problem. However, the training algorithm is almost based on the gradient information with inherent defect. Evolutionary algorithm has the advantage of global optimization capability, independent of gradient information etc. Estimation of distribution algorithm (EDA) is a typical evolutionary algorithm which relies on the probability model of population. Therefore, the EDA is adopted to train the weight and bias of deep neural network in his paper. However, the large scale optimization capability of EDA is limited. Thereby, an improved strategy is proposed to enhance the large scale optimization capability of EDA. In the improved scheme, a random selection strategy is carried out to select partial variables for probability modeling instead of all of the variables, in order to reduce the computing time and the probability of combination explosion. A simulation is carried out to exhibit the validity of the improved algorithm.

CCS CONCEPTS

• Computing methodologies • Machine learning • Machine learning approaches • Neural networks

KEYWORDS

Deep neural network, Optimization, EDA, Training

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
CSAE 2019, October 22–24, 2019, Sanya, China
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6294-8/19/10$15.00
https://doi.org/10.1145/3331453.3362048

1 Introduction

Artificial neural network is inspired by the connection mechanism of biological neural network[1]. The training of ANN is based on the training algorithm, such as BP algorithm. The training algorithm makes use of the gradient information which comes from minimizing the loss function. However, the gradient information maybe disappears with the increasing depth of neural network. The gradient information may disappear in the smaller layer. Therefore, the deep learning technique is proposed and some new activation functions are proposed[2]. However, the training of deep neural network is also a challenge due to the combination of large scale parameters. Another direct way is based on the evolutionary algorithm which uses the objective function instead of gradient information for optimization[3]. The evolutionary algorithm has two types of individual updating strategy. A typical way is the gene operation which adopts selection, recombination and mutation operators to pass the genes to the next generation, such as the genetic algorithm. The parameters and architecture of the neural network are encoded as the gene for individuals. Current neural network architecture searching system is almost based on it[4]. Another evolutionary algorithm is based on the probability modeling of population, such as estimation of distribution algorithm, which has a strong global optimization ability. However, the optimization capability of EDA is limited due to the curse of dimension[5]. In this paper, an improved EDA is proposed for the parameters optimization of artificial neural network. In the improved algorithm, a random strategy is adopted for the variables. The probability model of selected variables will be updated by the statistics information of the promising individuals. And then, a new population is generated according to the sampling of probability model.

2 Stacked Autoencoder

Autoencoder is a typical unsupervised learning algorithm, and it can be used for the task of representation learning[6, 7]. It also provides a compression of knowledge representation of the original input. A reconstruction of the original input x is generated $x^{'}$, and the reconstruction error is used to train the neural network. The architecture of autoencoder is shown in Figure 1, for any new input x, we can compute the output of the hidden units a. The autoencoder often ends up learning a low-

dimensional representation which often gives a better representation of the input than the original raw input x.

Autoencoder makes use of encoding and decoding process to train the deep neural network.

$$a = f(W^T x + b) \tag{1}$$

$$y = g(Wa + b') \tag{2}$$

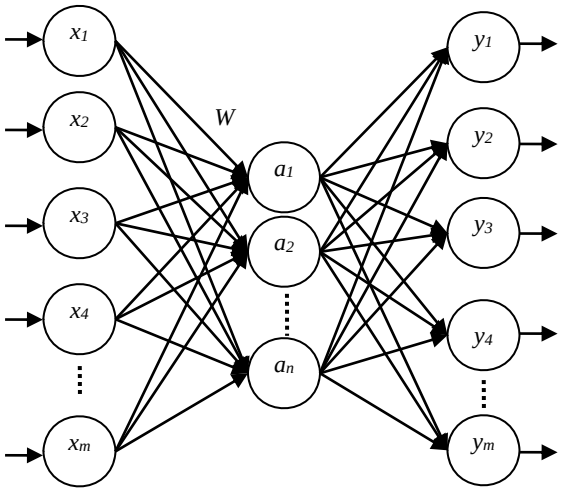

Figure 1: Autoencoder.

For the encoding layer, a is the output of hidden layer, W is the weight matrix, b is the threshold value, and f is the activation function of hidden layer. For the decoding layer, a is the output of hidden layer, W is the weight matrix, b' is the threshold value, and g is the activation function of reconstruction layer. x and y is the input and the reconstruction of x.

The loss function always consists of reconstruction error and sparse penalty as the following equation.

$$J = \|x - y\|^2 + \mu \sum a \tag{3}$$

J is the loss function, μ is the coefficient of sparse penalty. a is the output of hidden layer.

The deep neural network can be trained by autoencoer, which is always called stacked autoencoder as shown in Figure 2.

In BP neural network, only one hidden layer exists in the neural network. However, for deep neural network, there will be many hidden layers, and the training of deep neural network is invalid using BP algorithm. Therefore, the layer wised pre-training strategy is proposed for deep neural network training. Thereby, the first hidden layer 'Hidden layer 1' is trained by autoencoder of 'Hidden layer 1-Decoder 1'. After the training of hidden layer 1, the hidden layer 2 can be trained by 'Hidden layer 2-Decoder 2' which adopts the output of hidden layer 1 as the input of hidden layer 2. When the pre-training of all hidden layers is finished, a fine-tuning process can be carried out to train the whole neural network according to the labelled data, the loss function is shown as equation (4).

$$E = \|y_o - y_l\|^2 \tag{4}$$

y_o is the prediction output, y_l is the labelled data.

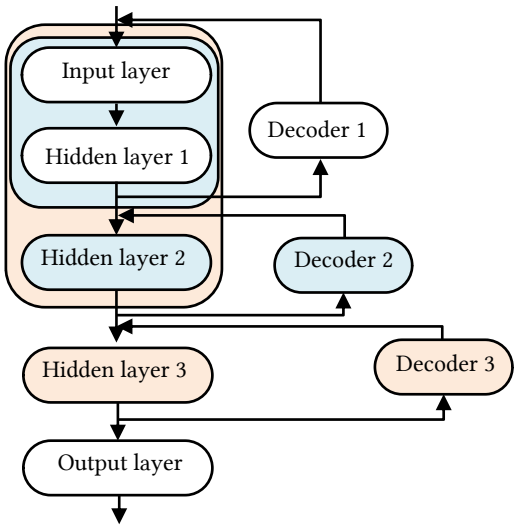

Figure 2: Diagram of Pre-training.

3 Improved Estimation of Distribution Algorithm

3.1 Traditional EDA

Compared with GA building gene blocks, EDA relies on the probability model of population evolution[8]. The probability model is built based on the statistical information of the most promising individual, and then the probability model is used for sampling to generate the new individuals. Meanwhile, the probability model is updated in each generation according to the new population. therefore, the population evolves, and finally search the optimal solutions. The diagram of the EDA is shown in Figure 3.

The most important step of EDAs is the construction of probabilistic model, and the Gaussian distribution of individuals is assumed to model and estimate the distribution of solutions[9, 10]. Therefore, mean and variance of promising individuals are computed according to the maximum likelihood. There the probability distribution $P(x_1, x_2 \cdots x_m)$ of the vector $(x_1, x_2 \cdots x_m)$ of m variables is a product of the distributions of individual variables:

$$P(x_1, x_2 \cdots x_m) = \prod_{i=1}^{m} P(x_i) \tag{5}$$

The mean and covariance parameters of the normal pdf can be estimated according to the promising individuals[11].

$$\overline{\mu}_i(k) = \frac{1}{N} \sum_{n=1}^{BN} x_i^n(k) \tag{6}$$

$$\sigma_i^2(k) = \frac{1}{N}\sum_{n=1}^{BN}(x_i^n(k) - \overline{\mu}_i(k))(x_i^n(k) - \overline{\mu}_i(k))^T \quad (7)$$

$\overline{\mu_i}(k)$ is the mean of ith variable in kth iteration, BN is the selected individuals size. $\sigma_i^2(k)$ is the covariance of ith variable in kth iteration.

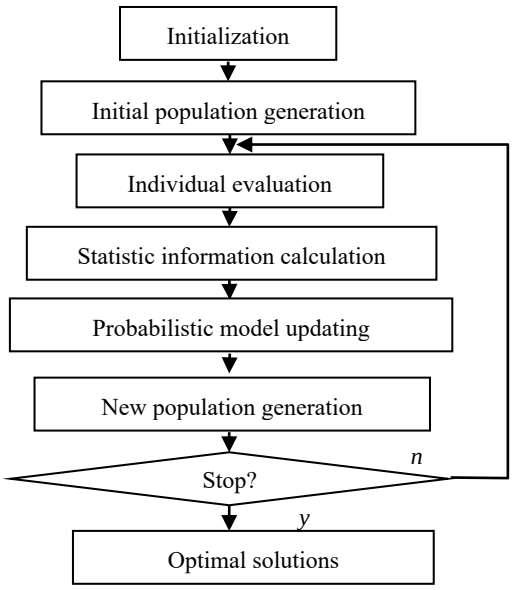

Figure 3: Flowchart of Traditional EDA.

The normal pdf $N(\mu_i, \sigma_i)$ is defined as equation (8).

$$N(x_i,\ \mu_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}} \quad (8)$$

The probability distribution $P(x_1, x_2 \cdots x_m)$ can be described as equation (9).

$$P(x_1, x_2, \cdots x_m) = \prod_{i=1}^{m} \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}} \quad (9)$$

The (μ_i, σ_i) have been estimated according to the promising individuals, and they are updated in every iteration.

3.2 Random Variable Selection Based EDA

For the neural network parameter optimization system, the number of the parameter is huge which is a typical large scale optimization problem. For traditional EDA, it is hard to obtain a solution due to the curse of dimensionality[12]. A large scale of dimensionality will increase the sampling time and the probability of combinational explosion. Therefore, a random strategy is proposed. For the random strategy, a group of variables are selected and the probability model of theses variables are updated by the statistics information of the promising individuals. In the improved EDA, the updating of variables is sampled by the probability model of corresponding variable partially instead of generating all variables by sampling as traditional EDA. The flowchart of the algorithm is shown in Figure 4.

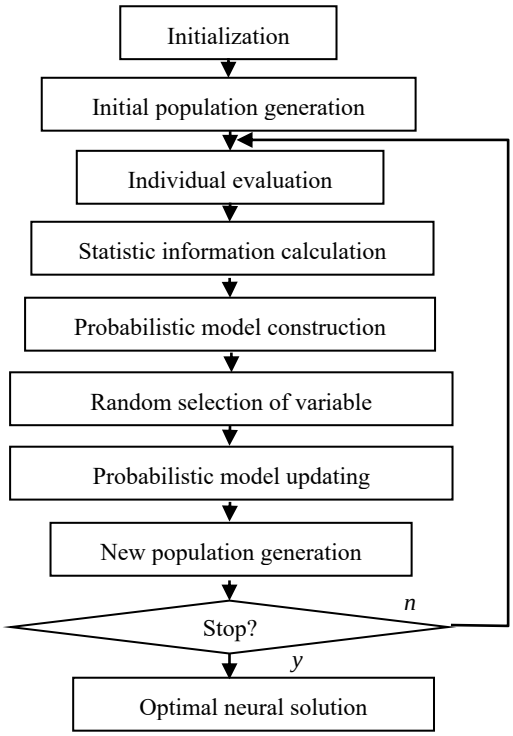

Figure 4: Flowchart of Improved EDA.

4 Simulation

In this paper, the MNIST database is adopted to verify the validation of the proposed algorithm, which contains 60,000 handwritten digits training images and 10,000 testing images [13]. In order to do a comparison, the layer by layer pre-training and fine-tuning based on traditional SGD algorithm is carried out, and then the layer by layer pre-training, two layers pre-training meanwhile and fine-tuning based on the proposed algorithm are testified. The structure of the neural network is 784-250-100-10, which contains two hidden layers with 250 and 100 neurons. The neurons of input and output layer are the pixels of input image and categories out. The configuration of the hardware is i5-8400 CPU, 8G RAM, GTX 1080TI GPU. NVIDIA GPU is adopted to accelerate the calculation. The code is programmed under MATALB. Some comparisons are carried out to exhibit the validation of the proposed algorithm.

Firstly, a layer by layer pre-training is adopted traditional style as the SGD based algorithm. The EDA is adopted to pretrain the hidden layer 1, and then the hideden layer 2 is pre-trained. Compared with the traditional SGD based algorithm as Figure 5, the performance of EDA is better than SGD based algorithm. Although, the performance of EDA in layer by layer

pre-training is promising, the time consumption is unsatisfactory. Therefore, the two hidden layers are considered to pre-train meanwhile. The results in shown in Figure 6. We can see that the performance of the two-layer pre-training meanwhile is promising. Hence, the EDA has the capability of searching the parameters of deep neural network meanwhile without layer by layer pre-training way. In Figure 7, we also compare the testing results in the fine-tuning process. We can see that the final classification accuracy is similar. Therefore, we can say that the proposed algorithm is effective to solve the large scale optimization problem of deep neural network. In order to testify the time consumption of the prosed algorithm, the consumption in each iteration is recoded which is shown in Figure 8. The two layers pre-training meanwhile consumes little longer than the layer by layer pre-training time. However, the sum of layer by layer pre-training time consumption is larger than EDA. Therefore, the time consumption of EDA is favorable in the training of deep neural network.

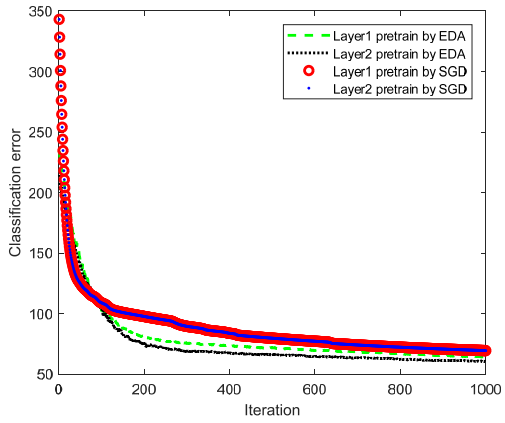

Figure 5: Layer by Layer Pre-training based on EDA and SGD.

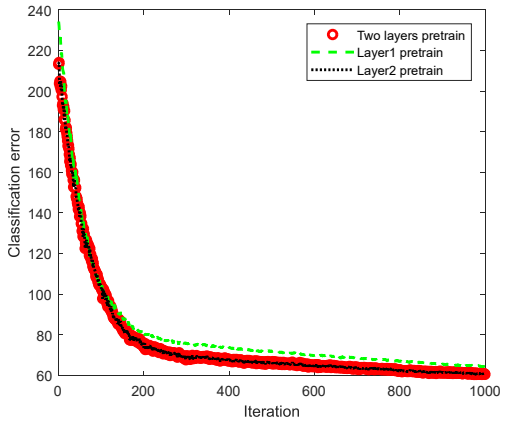

Figure 6: Layer by Layer Compared with Two Layer Meanwhile Pre-training based on EDA

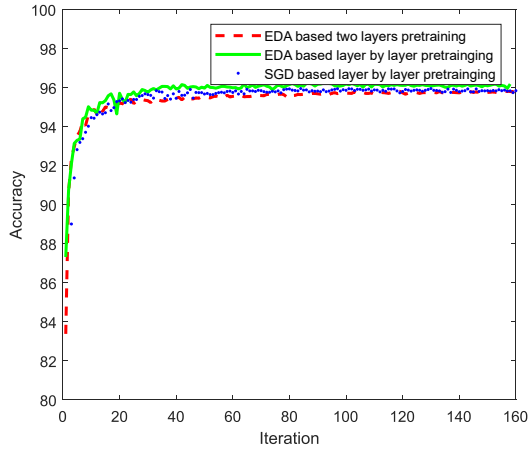

Figure 7: Testing Results in Fine-tuning based on EDA and SGD.

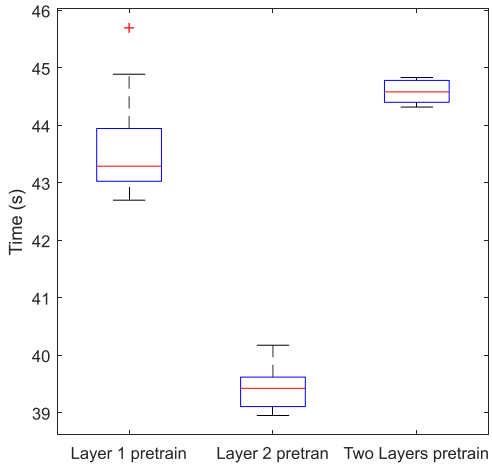

Figure 8: Time Consumption of Layer by Layer and two Layers Meanwhile Pre-training.

5 Conclusions

According to the simulation, the proposed scheme is valid for the optimization of deep neural network. The layer-wised pre-training is carried out for the gradient based training algorithms. For the EDA based algorithm, we can see that the optimization of the whole deep neural has the same performance as the layer-wised pre-training scheme. Therefore, the EDA algorithm can be used for the optimization of deep neural network conveniently, which also can overcome the disadvantage of gradient dependence.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under Grants (61603214, 61573213, 61673245, 61803227), National Key Research and Development Plan of China under Grant 2017YFB1300205, Shandong Province Key Research and Development Plan under Grants (2018GGX101039,2016ZDJS02A07), China Postdoctoral Science Foundation under Grant 2018M630778 and Independent Innovation Foundation of Shandong University under Grant 2018ZQXM005.

REFERENCES

[1] Y. LeCun, Y. Bengio and G. Hinton (2015). Deep learning, Nature, 521, 436--444.

[2] G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. Science, 313, 504--507.

[3] F. A. W. M. Mohr (2018). ML-Plan: Automated machine learning via hierarchical planning. Machine Learning, 107, 1495--1515.

[4] M. Wistuba, A. Rawat and T. Pedapati (2019). A Survey on Neural Architecture Search. arXiv e-prints, 1905.01392.

[5] W. Dong, T. Chen, P. Tiňo, and X. Yao (2013). Scaling Up Estimation of Distribution Algorithms for Continuous Optimization. IEEE Transactions on Evolutionary Computation, 17, 797-822.

[6] Y. Qiu, W. Zhou, N. Yu, and P. Du (2018). Denoising Sparse Autoencoder Based Ictal EEG Classification. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 1-1.

[7] Y. Bengio, A. Courville and P. Vincent (2013). Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence, 35, 1798--1828.

[8] M. Hauschild and M. Pelikan (2011). An introduction and survey of estimation of distribution algorithms. Swarm and Evolutionary Computation, 1, 111 - 128.

[9] Q. Xu, C. Zhang, J. Sun, and L. Zhang (2016). Adaptive Learning Rate Elitism Estimation of Distribution Algorithm Combining Chaos Perturbation for Large Scale Optimization. Open Cybernetics & Systemics Journal, 10, 20-40.

[10] Q. Xu, C. Zhang and L. Zhang (2014). A Fast Elitism Gaussian Estimation of Distribution Algorithm and Application for PID Optimization. The Scientific World Journal, 2014, 1-14.

[11] W. Dong, T. Chen, P. Tiňo, X. Yao (2013). Scaling Up Estimation of Distribution Algorithms for Continuous Optimization, IEEE Transactions on Evolutionary Computation 17, 797-822.

[12] A. Kabán, J. Bootkrajang and R. J. Durrant (2016). Toward Large-Scale Continuous EDA: A Random Matrix Theory Perspective. Evolutionary Computation, 24, 255-291.

[13] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. Manzagol (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, The Journal of Machine Learning Research 11, 3371--3408.