Universität Hamburg
Department Informatik
Computer Vision Group

On The Performance of Bio-Learning Approach for Classification Tasks

Independent Study Paper

Computer Vision

Shahd Safarani
Matr.Nr. 7132798
8safaran@informatik.uni-hamburg.de

06.01.2020

Abstract

Deep learning methods have limitations in terms of generalizability and the human brain can be a source of inspiration due to its robustness and great performance across many tasks. One of the approaches is to simulate the biological learning processes that are local and unsupervised. In this paper, we investigate the performance of the bio-learning algorithm by comparing it to baseline models that perform well on a classification task. Although this approach seems to be promising and represents an attempt to conduct further research on bio-inspired techniques, results show that the bio-learning algorithm is difficult to train and inefficient when solving a simple classification task. Thus, it leads to the question that is how to simplify and improve the algorithm design.

Contents

1 Introduction

In the last decade, deep learning methods proved to achieve the state-of-the-art results on many vision-related tasks. In most cases, deep networks need a huge amount of data to solve simple tasks like classification and the training process is entirely supervised which could reduce the ability of these models to generalize better to unseen data. Thus, these methods have limitations in terms of generalizability. In comparison, biological brains that exhibit very strong and robust intelligent behavior can generalize to many new situations and complex tasks without the requirement for too much data. So, one approach to take inspiration from the brain is to try to simulate the learning process that happens in biological brains. The biological learning rules are known to be local unlike the backpropagation algorithm that adjusts the artificial networks' weights to optimize a global objective function. One recent paper [2] addresses this topic and proposes an approach for training neural networks that simulate the biological unsupervised learning. This approach is referred to as *Bio-Learning*.

This work is an attempt to reproduce the results in [2] and to investigate bio-learning from several aspects.

2 Background

- **Supervised and Unsupervised Learning** There are three types of learning: Supervised, Unsupervised and Reinforcement Learning. Supervised approach is training a model to map an input to an output. This means, each example in the dataset is a pair of an input vector and a desired output value. On the other hand, the dataset is not labeled in the case of unsupervised learning and the model has to learn the structure of the input feature space and make sense out of it, e.g. clustering is unsupervised.

- **Bio-Learning in Neural Networks** According to [2], hidden layers can be separately trained in unsupervised fashion via the bio-inspired algorithm proposed in the paper (*Bio-Learning Algorithm*). Afterwards, the output layer is trained via backpropagation to minimize a certain loss function, given the weights of the pre-trained hidden layers are frozen.

  The bio-learning algorithm has two biologically plausible properties: Locality and Selectivity. Locality means Hebbian-like learning, where the synapse strength is proportional to the co-activity of the pre- and post-synaptic cells. Furthermore, selectivity comes from lateral inhibition of other neurons within a single layer to allow for the diversity of patter selectivity across many neurons.

3 Related Work

[2] inspires the work of this paper and provides a detailed mathematical explanation of how the proposed framework works. In order to verify the performance of the suggested algorithm, experiments are run on two classification datasets: MNIST and CFAR-10. The weights trained via bio-learning are shown in 2-D plots. [2] publishes their source code on: *https://github.com/DimaKrotov/Biological_Learning*.

4 Experimental Studies

4.1 Experimental Setups

To investigate the performance of the bio-learning approach, we compare two models on MNIST dataset: the first one is trained leveraging the bio-learning while the second one is trained end-to-end in supervised fashion and referred to as *Baseline*. The source code of this work is on: *https://github.com/Shahdsaf/Biological-Learning*.

Model Configuration

The baseline model has one hidden layer of 2000 neurons and an output layer of 10 neurons. It's trained end-to-end with Adam optimizer [1] and a learning rate of 0.0002. The activation function of the hidden layer is ReLu while for the output Softmax is used. Also, the learning rate decays by a factor of 0.1 in case the validation loss does not decrease for 15 epochs. The batch size is 100 while the total training process lasts for 300 epochs.

Inspired by [2], the bio-learning model is investigated and the following subsection explains its configuration in detail. It can be divided into two parts: The unsupervised part which represents the only hidden layer of the network while the supervised part refers to training the output layer separately in supervised fashion. The architecture of the model is the same as the baseline.

- **Unsupervised Layer** As provided by [2], the model has only one hidden layer which is trained in unsupervised fashion via Bio-learning algorithm. Giving data in mini-batches, all neurons adjust their weights using a local hebbian-like update rule that enhances pattern selectivity across all neurons in this layer. The number of neurons in this layer can vary according to the dataset description. As given in [2], number of neurons is 2000.

  Hyperparameters for training the hidden layer are set based on [2]. Initial learning rate $eps0$ is set to 0.04 while mu and $sigma$ are set to (0,1). Mini-batch size is 100 and total number of epochs is 750. Ranking parameter k equals 7, the Lebesgue norm of the weights p is 3 and Strength of the anti-hebbian learning $delta$ is set to 0.4.

The activation function is defined as x^n in case of positive x and 0 otherwise. During experimentation, we test the effect of different values of n. Notice that the activation function is equivalent to ReLu in case $n = 1$ like in the baseline model.

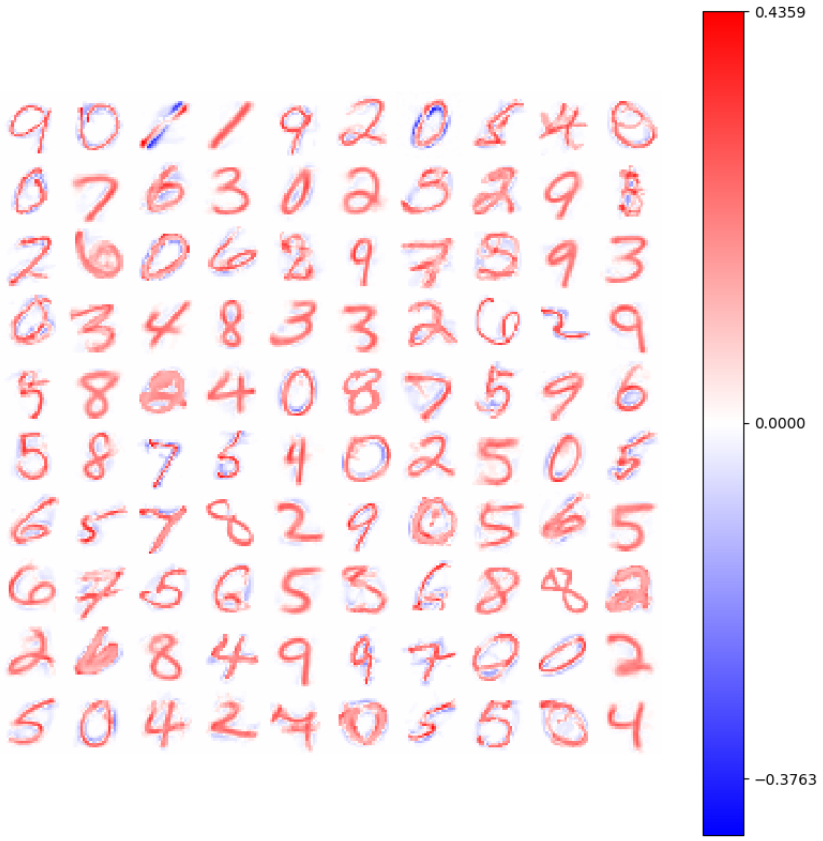

Figure 1: Reproduction of results in [2]. 100 randomly selected detectors in the hidden layer visualized after training in unsupervised fashion.

- **Supervised Output Layer** In order to output the classification probabilities of different classes, the model has an output layer consisting of neurons, whose number is equal to the number of classes, given a certain dataset (10 in this case). This layer is trained in supervised fashion via Backpropagation, after freezing the weights of the hidden layer.

  Hyperparameters for training the output layer are set to values inspired by [2] in addition to some modifications. The activation function is Softmax unlike [2]. This is due to the change of the loss function to categorical Cross-Entropy unlike what is optimized in [2]. Supervised training has the same hyperparameters as for the baseline such as the optimizer and learning rate.

4.2 Results

In Figures 2 and 3, the training and test accuracy are shown for the baseline model and bio-learning model that uses ReLu for activating the hidden layer ($n = 1$). In these figures, it is noticed that the baseline reaches 100% on the training set while the test performance reaches around 98.4%, considering that this result is obtained with no need for hyperparameter tuning, namely hyperparameter values that worked for bio-learning approach were applied to the baseline as well.

In contrast, the bio-learning performance on the training and test sets reaches around 93.5%. The performance is not as good as the baseline and obtaining these results happened after several adjustments of the training parameters such as learning rate, activation and loss functions etc.

In addition, different values of n were experimented. In Figures 4 and 5, better performances of the bio-learning approach are noticed, where the training accuracy reaches around 99% and the test accuracy around 97.5%.

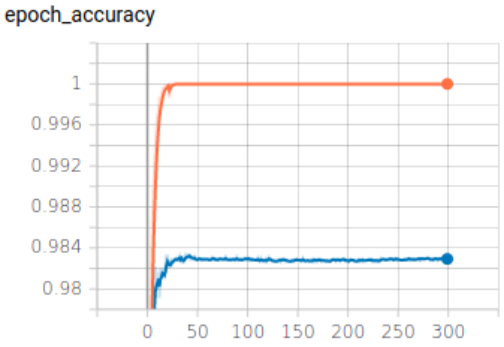

Figure 2: Baseline Model, x-axis is epochs and y-axis is accuracy. Training accuracy in red and validation in blue.

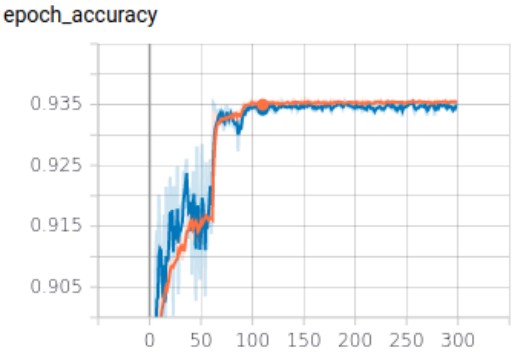

Figure 3: Bio-learning Model, x-axis is epochs and y-axis is accuracy. Training accuracy in red and validation in blue. The activation function of the hidden layer is ReLu ($n = 1$).

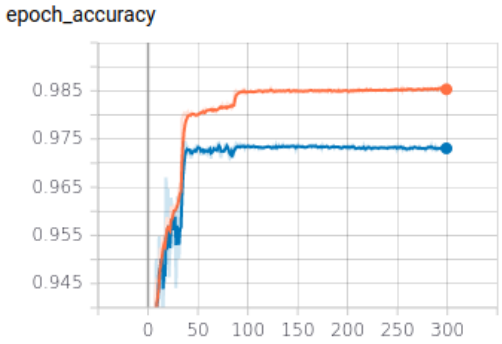

Figure 4: Bio-learning Model, x-axis is epochs and y-axis is accuracy. Training accuracy in red and validation in blue. The activation function of the hidden layer is x^3 for positive x.

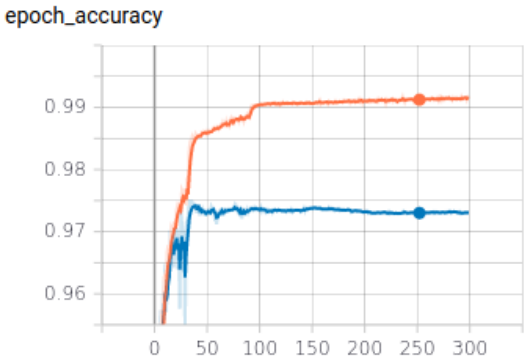

Figure 5: Bio-learning Model, x-axis is epochs and y-axis is accuracy. Training accuracy in red and validation in blue. The activation function of the hidden layer is x^6 for positive x.

4.3 Discussion

From the results, we see in general that the approach does not perform as well as the baseline that leverages backpropagation end-to-end. While training the bio-learning model, it is noticed that training needs a lot of trials and hyperparameter experimentation so that the model can solve the task well, e.g. tuning the learning rate and optimizer etc.

Softmax is used generally for classification in readout layers, while it is noticed that [2] uses Tahn and a loss function different from Cross-Entropy. Using such configurations were not justified in the paper clearly. Furthermore, the activation function of the hidden layer x^n for positive x (0 otherwise) is not justified either, where the authors could simply use ReLu as a default function in hidden layers. Hence, having to tune a lot of hyperparameters such as n and m (used in their loss function) leads to inefficiency when training the network in the supervised phase.

Not to mention that the unsupervised training has also a lot of hyperparameters to tune and try out.

In summary, this approach makes it more difficult to train a simple network on a simple task and it does not obtain the state-of-the-art accuracy results (efficiently).

5 Conclusion

In this paper, we investigate the bio-learning approach suggested in [2] and compare it to a baseline model that is trained end-to-end in supervised fashion. According to the results shown above, the approach has many hyperparameters to tune and training the hidden layer separately from the output layer seems to be very inefficient in terms of time-consuming hyperparameter tuning and accuracy performance.

References

[1] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[2] Dmitry Krotov and John J. Hopfield. Unsupervised learning by competing hidden units. *CoRR*, abs/1806.10181, 2018.