Biologically Plausible Training of Deep Learning Models

Christian Adkins, Ankush Vasishta
Department of Computer Science
University of Georgia
Athens, Ga 30602
(caadkins,av21224)@uga.edu

Abstract

Training of deep neural networks has traditionally relied on variations of the back-propagation and gradient descent algorithms. Although this framework has shown great performance on various tasks, it also presents certain problems. These include the difficulty in parallelizing error computations and the loss of certain desired properties when gradient clipping is introduced in order to achieve differential privacy. We propose an error propagation scheme that is privacy friendly and robust to training with noise, while avoiding some of the crucial problems involved with back-propagation.

1 Introduction

Machine learning is a complex field of study, being influenced by many other fields and being used as a tool to solve problems in every field. The main way that many machine learning models are evaluated is by the accuracy with which a model can perform a task. Most machine learning problems are framed as classification or regression tasks; these tasks allow for straightforward measurements of a model's success. There are other important factors to consider in machine learning, especially when looking at powerful and complex models like neural networks. One of the most critical concerns is privacy, and how the design of a model can hinder attempts to achieve sufficiently-high levels of privacy. Differential privacy is the current state-of-the-art definition of privacy in the context of machine learning. This definition is based on limiting the impact any single individual can have on the resulting machine learning model.

1.1 Proposed solutions

Many works have shown how to successfully incorporate differential privacy guarantees with machine learning algorithms using the common error propagation technique known as back-propagation (BP). It remains an area of research to provide the same guarantees for alternative backward phase algorithms that are designed to overcome some of the drawbacks of BP. We attempt to analyze two proposed alternative error propagation schemes: direct feedback alignment (DFA) and target propagation (TP). We try to evaluate the performance of these alternatives on two different machine learning model architectures, namely a fully-connected network (FCN) and a convolutional neural network (CNN), to demonstrate the usefulness of DFA and TP for a set of common complex machine learning problems. We place an emphasis on using these methods in a private setting, striving to achieve differential privacy without the use of back-propagation; we attempt to illustrate the impact of a varying noise level on the performance of each task.

Preprint. Under review.

2 Related Work

2.1 Differential privacy

The prevalence and size of datasets available to analyze has greatly aided the development of machine learning in recent years. Along with this growth has come the increased awareness that the models produced by machine learning algorithms can provide much information about these datasets and the individual records they contain. While it is useful to learn directly from as much data as possible to create useful and accurate models, seeking that utility must be balanced by efforts to preserve the privacy of individuals involved in the data.

Over the years, the state-of-the-art definition of privacy has changed. The current state-of-the-art privacy mechanism for machine learning is differential privacy. At a high level, in order to be differentially private (DP) an algorithm must guarantee that no observer can determine whether a computation used an individual's information. Ideally the sensitivity for a given machine learning problem would be calculated to determine the exact impact an individual would have on the resulting model and act accordingly. Due to the difficulty of calculating the sensitivity for complex neural networks, a technique known as gradient-clipping is used in practice to limit individuals' contributions to the gradient used in the backward phase of machine learning. After the clipped gradients are averaged, random noise is injected to the resulting gradient based on a random distribution defined by the strictness of the privacy for a given problem. Using this clipped, noisy gradient during the training process achieves differential privacy for two neighboring databases (differing in at most one tuple).

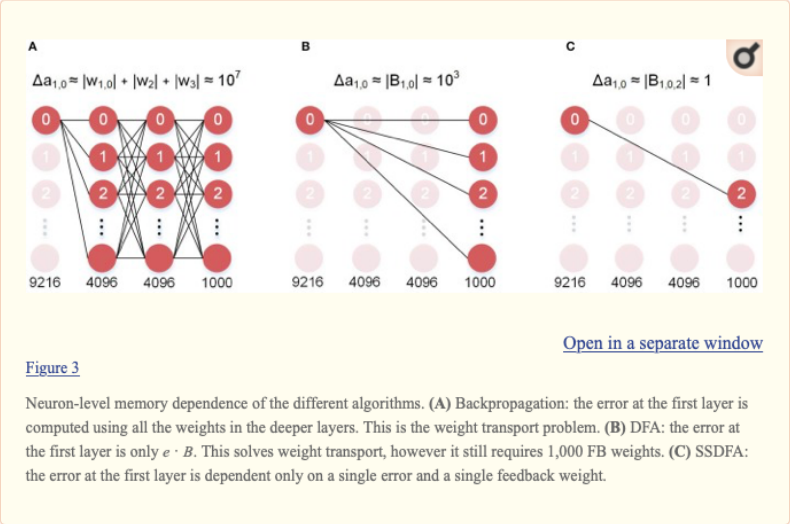

Figure 1: Comparison between Backprop and DFA

2.2 Back-propagation

There are typically two recurring phases during training of neural networks. Activations for each layer are computed by multiplying the input signal vector from the previous layer with a matrix of weights, applying nonlinearity, and then transporting the result to the next layer. This is considered the forward phase. The network computes the error of the current output when the activation (signal) reaches the output layer. The output is compared to the ground truth to derive an error signal which is then propagated back towards the input layer.

This back-propagation must be done sequentially, because computing the gradient for each subsequent layer's parameter update requires the weights used at its upper layer (closer to the final output) in the forward path. This presents two concerns. First, parallelizing these computations is extremely difficult due to the dependency between layers. The development of techniques and hardware like GPUs, designed to greatly improve repeated calculation speed, can not be properly leveraged in this error propagation scheme. The second issue is the fundamental difference between BP's behavior and the behavior of biological neurons' signal propagation.

2.3 Direct feedback alignment

This technique of error propagation involves directly connecting the feedback paths from the output layers to the neurons earlier in the pathway. Just like back-propagation, the error signal is represented as a second signal in the neurons participating in the forward path. The only difference lies in the connection of the feedback path to the neurons which, in the case of DFA, is different for each neuron that was part of the forward path.

The error signal is propagated to each layer and the gradient update for the new parameters depends upon the activation of each layer.The weights receive an update proportional to the partial derivative of the error function with respect to the current weight in each iteration of the training. The update of the weight, determined by the gradient, would be prevented since the gradient at each step would decrease or become small. This leads to the gradient diminishing problem. The direct feedback alignment is useful for preventing this problem.

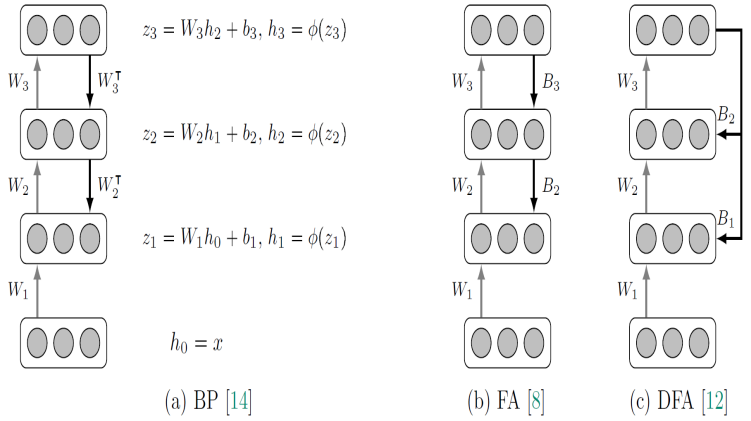

Figure 2: Comparison of different error transportation configurations for an MLP with 2 hidden layers.

2.4 Target propagation

Instead of calculating a loss gradient to associate with each feedforward unit's activation value, TP determines a 'target value'. This target should be close to the activation value, but should be likely to have provided a smaller loss if it had been obtained in the forward phase. As long as the target is very close to the feedforward value, TP should behave similarly to back-propagation. Each layer can be updated separately once a good target is computed. This avoids the parallelization problem faced by back-propagation, while also being more biologically plausible.

Auto-encoders are used to assign targets to each layer during supervised training of neural networks. To ensure robust training, a linear correction is introduced to account for the imperfectness of the encoders. These principles are applied to replace back-propagation in training algorithms.

3 Mathematical Formulation

In this section, we describe the mathematical formulation and equations pertaining to BP and DFA with a small example which has been described in et al Nøkland, Arild[11]. Let (x, y) be mini-batches of input-output vectors that we want the network to learn. For simplicity, assume that the network has only two hidden layers, and that the target output y is scaled between 0 and 1. Let the rows in W_i denote the weights connecting the layer below to a unit in hidden layer i, and let b_i be a column vector with biases for the units in hidden layer i. The activations in the network are then calculated as

$$a_1 = W_1 * x + b_1, h_1 = f(a1), \tag{1}$$

$$a_2 = W_2 * h_1 + b_2, h_2 = f(a2), \tag{2}$$

$$a_y = W_3 * h_2 + b_3, \hat{y} = f_y((a_y)) \tag{3}$$

where f() is the non-linearity used in hidden layers and fy() the non-linearity used in the output layer. If we choose a logistic activation function in the output layer and a binary cross-entropy loss function, the loss for a mini-batch with size N and the gradient at the output layer e are calculated as

$$J = -\frac{1}{N}\sum_{m,n} y_{mn}\log\hat{y}_{mn} + (1-y_{mn})\log(1-\hat{y}_{mn}) \tag{4}$$

$$e = \delta a_y = \frac{\partial J}{\partial a_y} = \hat{y} - y \tag{5}$$

where m and n are output unit and mini-batch indexes. For the BP, the gradients for hidden layers are calculated as

$$\delta a_2 = \frac{\partial J}{\partial a_2} = (W_3^T e) \odot f'(a_2),\ \delta a_1 = \frac{\partial J}{\partial a_1} = (W_2^T \delta a_2) \odot f'(a_1) \tag{6}$$

where $\odot$ is an element-wise multiplication operator and f'() is the derivative of the non-linearity. This gradient is also called steepest descent, because it directly minimizes the loss function given the linearized version of the network.

For DFA, the hidden layer update directions are calculated as

$$\delta a_2 = (B_2 e) \odot f'(a_2),\ \delta a_1 = (B_1 e) \odot f'(a_1) \tag{8}$$

where B_i is a fixed random weight matrix with appropriate dimension. If all hidden layers have the same number of neurons, B_i can be chosen identical for all hidden layers.

4 Experiments

We test different configurations of our methods on classification tasks for the following datasets: MNIST, Fashion-MNIST, and CIFAR (10 and 100). Our procedures are tested on a combination of two architectures: a fully-connected network and a convolutional neural network. The base version of the FCN consists of a single fully-connected hidden layer between the input and output layers. The base version of the CNN contains two blocks of convolution and max pooling followed by dropout before a fully-connected layer and finally the output layer. The number of hidden units in the FCN, as well as the dimensions and sizes used in the CNN, differ based on the dataset used for classification. Two different error propagation schemes are implemented: Back-propagation and Direct Feedback Alignment. We were unable to implement the third scheme, Target Propagation, effectively.

We analyze the success of the two backward-phase algorithms based on several factors. The change of loss and accuracy over epochs provides insight into each scheme's ability to create useful learning for different tasks over time. Looking at the algorithms' accuracies when combined with different model architectures reveals whether they provide good general performance or they are reliant on one specific configuration. Performance at varying network depths shows any inherent level of susceptibility to overfitting. Finally, each method's utility in the private setting is analyzed based on accuracy achieved at varying noise levels.

4.1 Implementation

The incorporation of differential privacy via gradient clipping and noise injection in Pytorch was performed by a package called "pytorch-dp". The description and implementation of this package can be found at https://github.com/facebookresearch/pytorch-dp. In practice, we provided the clipping and noise parameters to a "privacy engine" and attached it to the existing optimizer in the non-private version of each experiment. This privatization was only performed on models using back-propagation for the error propagation.

The implementation of Directed Feedback Alignment we use is found at https://github.com/bcrafton/ssdfa . Note that this implementation is done in Tensorflow, not Pytorch. We tried to create a working implementation of DFA in Pytorch, but were unsuccessful.

The general structure of our architectures and the default values of certain parameters were inspired by tutorials found at https://pytorch.org/tutorials/ .

Both authors assisted in writing and editing the report, as well as developing and executing various experiments.

4.2 Performance over Epochs

Figure 3 demonstrates accuracy changes over 100 epochs of training for a non-private fully-connected network using Direct Feedback Alignment on the CIFAR10 dataset.

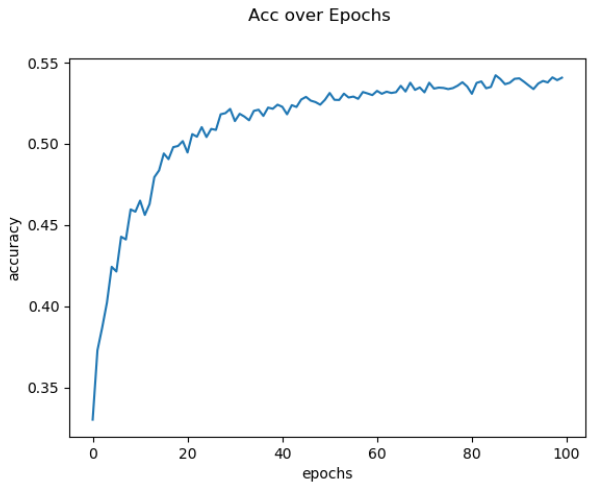

Figure 3: Change in accuracy during training. Non-private FCN using DFA on CIFAR10 dataset.

Figure 4 shows the change in accuracy over epochs for a FCN using BP for CIFAR10 dataset

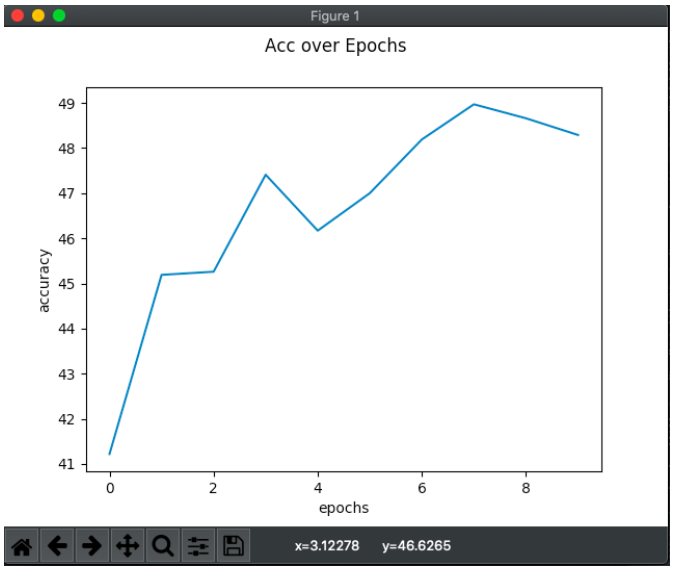

Figure 4: Change in accuracy during training. Non-private FCN using BP on CIFAR10 dataset.

Next, we show in Figures 5 and 6 how the accuracy and loss change with training over 50 epochs for the private version (noise level set to 0.1, clipping threshold set to 4.0) of the Back-Propagation FCN

on the MNIST dataset. Classifying MNIST images is known to be an easier task than classifying CIFAR images, and that is reflected in the overall lower accuracy level in Figure 3. It is important to note in the latter two figures that the accuracy steadily declines even as the loss fluctuates; this represents an observed pattern of performance degradation when using the privacy engine during the training phase. This pattern is discussed further in the Conclusion section.

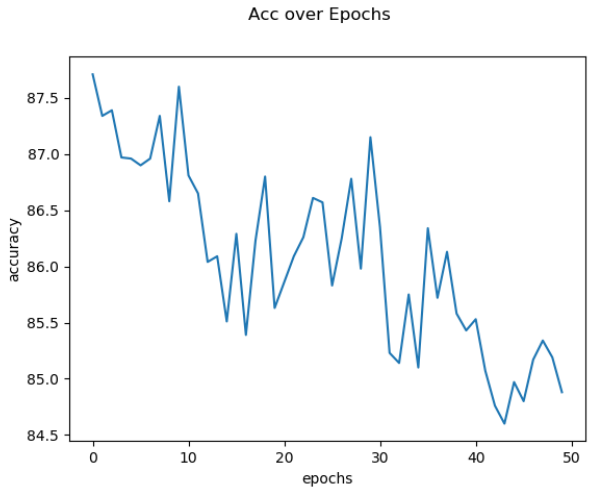

Figure 5: Change in accuracy during training. Private FCN on MNIST dataset.

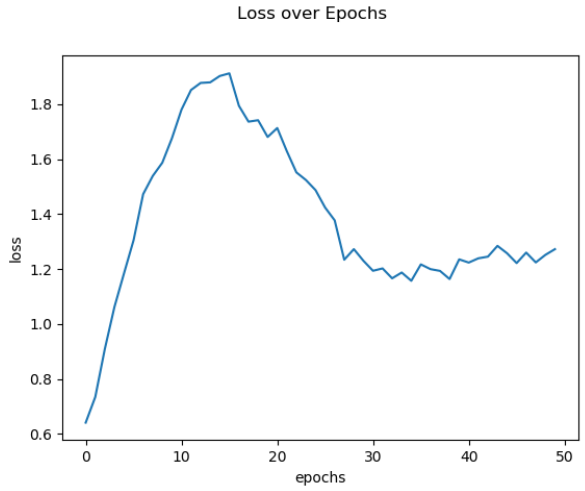

Figure 6: Change in loss during training. Private FCN on MNIST dataset.

4.3 Accuracy across Architectures

Between Figures 7 and 8, we show that Direct Feedback Alignment can be successfully incorporated with various structures of machine learning models. The eventual decline in accuracy on the MNIST dataset is within expectations for the given number of epochs on this simple classification task.

4.4 Accuracy over Network Depth

Table 1 shows two examples of the effects that network depth has on model performance. In the case of the privatized FCN, the shallower version was designed first and achieved reasonable accuracy; the addition of an extra hidden layer degraded the accuracy. The non-private convolutional neural network was less impacted by such changes. The original version of the CNN was deeper, with two

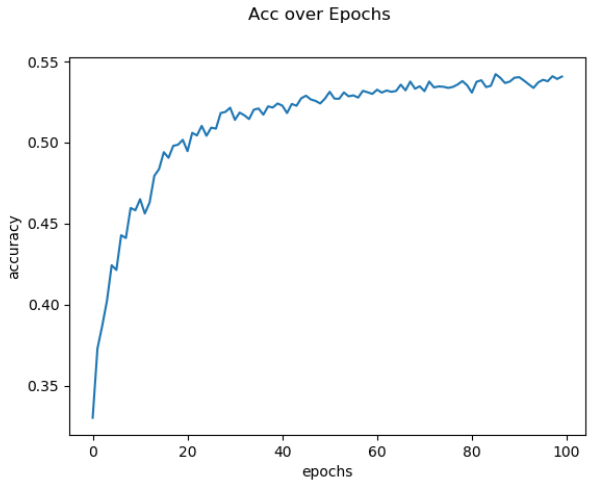

Figure 7: Accuracy for non-private FCN using DFA on CIFAR10 dataset.

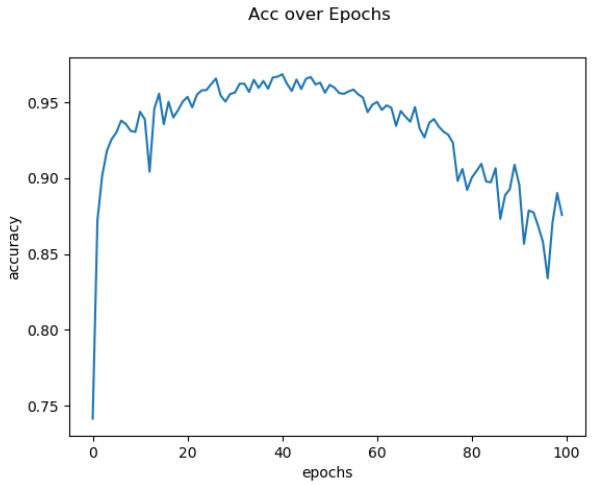

Figure 8: Accuracy for non-private CNN using DFA on MNIST dataset.

blocks of convolution and pooling; removing one of the layers (and adjusting parameters accordingly) only resulted in a very minor drop in accuracy.

Architecture	Accuracy @ Greater Depth	Accuracy @ Lesser Depth
Private FCN	64	73
Non-Private CNN	97	95

Table 1: Resulting accuracy of two models at different depths on MNIST dataset with back-propagation.

4.5 Accuracy over Noise Level

In Table 2, the value of "None" for sigma indicates a non-privatized test. The results show the loss of accuracy accompanying stricter privacy requirements.

Sigma	Accuracy
None	97
0.1	78
0.5	73
1.0	61

Table 2: Resulting accuracy of FCN on MNIST dataset using back-propagation, with varying noise levels.

5 Conclusion

The vast majority of research and development in neural networks has defaulted to using back-propagation in the backward phase of training, despite the difficulty in parallelization this induces. Direct feedback alignment and target propagation are two proposed biologically-inspired alternatives to BP. Our experiments attempt to demonstrate the feasibility of BP and DFA as error schemes, including in the differentially private context. Limitations in incorporating the privacy engine implementation with our own neural network developments seemingly hindered the performance of the private versions of our back-propagation models beyond the expected performance degradations and prevented experiments looking into privacy effects on DFA models. Moving forward, we will refine this evaluation strategy to more accurately assess how these schemes operate when combined with differential privacy, and we will expand our analysis to include Target Propagation.

References

[1] Abadi, Martin, et al. "Deep learning with differential privacy." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. 2016.

[2] Bottou, Léon. "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT'2010. Physica-Verlag HD, 2010. 177-186.

[3] Dwork, Cynthia, et al. "Our data, ourselves: Privacy via distributed noise generation." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2006.

[4] Dwork, Cynthia, et al. "Calibrating noise to sensitivity in private data analysis." Theory of cryptography conference. Springer, Berlin, Heidelberg, 2006.

[5] Hinton, Geoffrey, Nitish Srivastava, and Kevin Swersky. "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent." Cited on 14.8 (2012).

[6] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[7] Lee, Dong-Hyun, et al. "Difference target propagation." Joint european conference on machine learning and knowledge discovery in databases. Springer, Cham, 2015.

[8] Lillicrap, Timothy P., et al. "Random synaptic feedback weights support error backpropagation for deep learning." Nature communications 7.1 (2016): 1-10.

[9] Luo, Liangchen, et al. "Adaptive gradient methods with dynamic bound of learning rate." arXiv preprint arXiv:1902.09843 (2019).

[10] Mironov, Ilya. "Rényi differential privacy." 2017 IEEE 30th Computer Security Foundations Symposium (CSF). IEEE, 2017.

[11] Nøkland, Arild. "Direct feedback alignment provides learning in deep neural networks." Advances in neural information processing systems. 2016.

[12] Robbins, Herbert, and Sutton Monro. "A stochastic approximation method." The annals of mathematical statistics (1951): 400-407.

[13] Rumelhart, David E., et al. "Backpropagation: The basic theory." Backpropagation: Theory, architectures and applications (1995): 1-34.

[14] Crafton, Brian et al. "Direct Feedback Alignment With Sparse Connections for Local Learning." Frontiers in neuroscience vol. 13 525. 24 May. 2019, doi:10.3389/fnins.2019.00525