Backforward Propagation (Student Abstract)

George Stoica, Cristian Simionescu

"Alexandru Ioan Cuza" University
sgeorge.sstoica99@gmail.com,cristian@nexusmedia.ro

Abstract

In this paper we introduce Backforward Propagation, a method of completely eliminating Internal Covariate Shift (ICS). Unlike previous methods, which only indirectly reduce the impact of ICS while introducing other biases, we are able to have a surgical view at the effects ICS has on training neural networks. Our experiments show that ICS has a weight regularizing effect on models, and completely removing it enables for faster convergence of the neural network.

Internal Covariate Shift

Internal Covariate Shift was first defined in (Ioffe and Szegedy 2015), but is originating from (Shimodaira 2000), which introduced *covariate shift*. ICS is a phenomena which happens during the training of neural networks as a result of applying the back-propagated gradients on a layer with regards of the weights of the previous layers before the update. Due to the update of the previous layers in the same back-propagation step, the distribution of the input for the current layer changes. This causes a shift in the step of the model for the latter layers and the model does not learn the loss of a batch but an approximation of it. It is presumed that as a consequence of this, smaller learning rates are needed in order to reach convergence. In this paper we study the effect of removing the ICS on the training process by introducing a novel method, Backforward Propagation.

Given a model architecture with a single layer and a given batch of samples, if the layer is wide enough, applying a Stochastic Gradient Descent (SGD) step with a learning rate of 1.0 should result in the model perfectly learning the current batch. For an architecture with one or more hidden layers, only the update on the first layer would be exactly in the direction of the gradient descent. The update on the second and latter layers would be based on the weights of the previous layer before the update, whose distribution would have changed during the same step, causing the update direction to deviate, making its update to be slightly obsolete.

Related Work

Research was done in the direction of eliminating ICS, starting from (Ioffe and Szegedy 2015) which introduced the widely used Batch Normalization (BatchNorm). The method was shown to reduce the impact of the ICS by normalizing layer inputs based on mini-batch statistics using BatchNorm layers, and therefore by fixing the distribution of the layer inputs the effect of the ICS is minimized allowing the use of higher learning rates.

(Arpit et al. 2016) introduced Normalization Propagation (NormProp), another technique of combating ICS through normalization. However, unlike BN, it does not use mini-batch statistics for normalization but has the assumption that the activations follow the Gaussian distribution.

In (Awais, Iqbal, and Bae 2020) ICS is further studied and the authors claim that the success of BatchNorm is strongly tied to reducing the effect of ICS, combating voices that reject the ICS hypothesis as to why BatchNorm is successful.

As opposed to the approaches above, which reduce the effect of the ICS using normalization, we study the complete elimination of it by recalculating the update for each layer except the first, at the cost of more computation.

Backforward Propagation

Our method of completely removing ICS consists of recalculating the update for each layer, taking into account the change of distribution for the previous layers. This can be achieved by redoing the forward and backpropagation steps and updating one layer each time, starting from the first layer until the last layer.

The naive implementation of the Backforward Propagation algorithm described above can be achieved by applying n forward and backpropagation steps on the whole model (where n is the number of layers without the input layer), applying the update only on the layer with the index of the current iteration. While this process adds complexity and greatly decreases the training speed, the purpose of this is to study the effect of eliminating ICS and whether this process has any potential of further use.

A better implementation of the algorithm would be to apply the forward and the backpropagation step on portions of the model (and not the entire model), however this requires custom created models to support this operation. For a general solution, access to the computation graph of the gradients is required, which isn't easily available in Deep Learning frameworks.

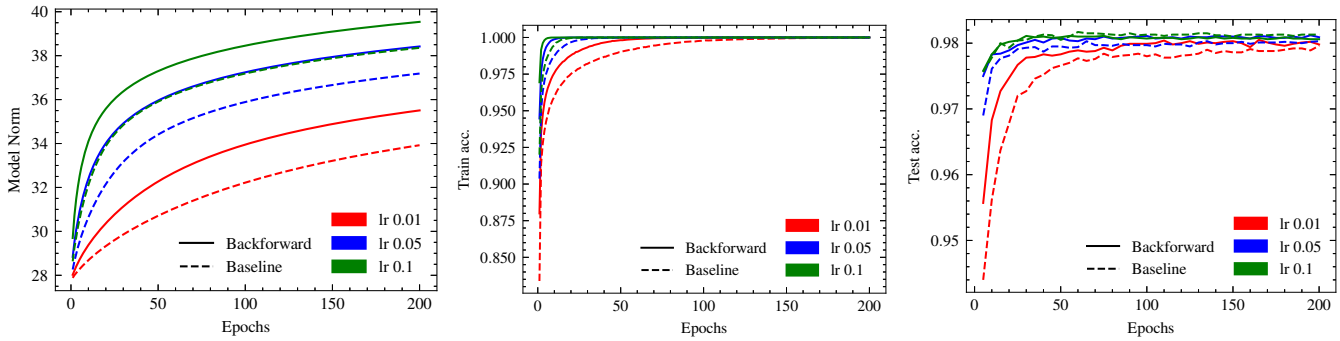

Figure 1: Backforawrd results using different learning rates

Experiments

In order to validate that Backforward Propagation has the effect we would expect, we setup a toy experiment using a batch of 4 samples from the MNIST dataset (LeCun et al. 1998). Using a Multilayer Perceptron (MLP) with one hidden layer, we do a single SGD update step using that batch, and test to see the loss value before and after the update. With a large enough hidden layer, if we use a learning rate of 1.0, the loss on that batch should be 0 if ICS would not be a factor. We found that a hidden layer of size 256 is enough for Backforward to reduce the loss to $\epsilon \approx 0$, meaning the model was able to almost completely learn the batch in a single update. However, when using the baseline in the same condition, the loss of the given batch in the second epoch is very large, enough to show that the discrepancy is due to ICS.

Since our method of removing ICS comes with a high computational cost, our experiments were performed on MNIST using a MLP with one hidden layer of 256 neurons. We trained the MLP using the Backforward Propagation algorithm described above and a standard SGD baseline for comparison. In order to reduce any impact from other methods, we chose to not use any data augmentations, optimizer momentum and no weight decay.

We compared the results of the Backforward Propagation algorithms with a baseline, with multiple learning rates of 0.1, 0.05 and 0.01. We have observed that the norm of the weights is larger in the absence of ICS, which implies that ICS has a regularizing effect on the training process.

Our results in Figure 1 show that that the baseline has similar results to the Backforward Propagation when using a learning rate of 0.1. This is due to the fact that Backforward Propagation has higher difficulties of escaping a plateau due to its direction being better than the baseline's. We hypothesize that this is due to the baseline being able to get out of such plateaus easier because of oscillations in its update direction. This problem can be mitigated by decreasing learning rate over time, a common method during training.

However, our algorithm converges faster and performs better when using learning rates of 0.05 or 0.01, a consequence of removing ICS which enables the model to make steps in the exact gradient direction, eliminating any error in the approximation.

Conclusion and Future Work

We have designed a method for completely eliminating ICS and have studied it's effect, showing that it can enable better performance and faster convergence, while also identifying that ICS has a regularization effect. We want to extend the experiments, applying our method for deeper architectures where ICS has greater impact. Because of the prohibitive computational cost of using Backforward Propagation, we also want to investigate if there are any useful applications where such a cost would be acceptable, for example model warmup or fine-tuning.

Furthermore, we want to compare Backforward Propagation and BatchNorm performance on similar architectures and combine the two methods, in order to study how does the BatchNorm behaves in the absence of ICS, which may reinforce or disprove arguments with regards to how does BatchNorm work.

References

Arpit, D.; Zhou, Y.; Kota, B.; and Govindaraju, V. 2016. Normalization propagation: A parametric technique for removing internal covariate shift in deep networks. In *International Conference on Machine Learning*, 1168–1176. PMLR.

Awais, M.; Iqbal, M. T. B.; and Bae, S.-H. 2020. Revisiting internal covariate shift for batch normalization. *IEEE Transactions on Neural Networks and Learning Systems*, 32(11): 5082–5092.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. PMLR.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Shimodaira, H. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2): 227–244.