ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

Experimental study of the neuron-level mechanisms emerging from backpropagation

Simon Carbonnelle and Christophe De Vleeschouwer *

Université catholique de Louvain - ICTEAM
Louvain-La-Neuve - Belgium

Abstract. The backpropagation algorithm [1] is the most successful learning algorithm for training deep artificial neural networks. Its inner workings are in stark contrast with other learning rules, as it is based on a global, black box optimization procedure rather than the repetition of a local, neuron-level procedure (e.g. like hebbian learning [2]). In this paper, we present preliminary evidence suggesting that local, neuron-level mechanisms are in fact emerging during backpropagation-based training of neural networks and describe what could be key components of it.

1 Introduction

It has often been assumed that in a trained network, the activation of hidden neurons reflects the presence/absence of a semantic concept in the input signal. This resulted for example in the usage of activation functions with a binary threshold since the origin of artificial neural networks (e.g. [1]). This intuition is probably due to the fact that artificial neural networks have been inspired by neurons from the brain which, we suspect, operate in such way [3]. It has also been strengthened by several works showing experimentally that activations in a network trained through backpropagation sometimes correlate with the presence of semantic concepts in the input signals (e.g. [4, 5]).

What lacks around this intuition, however, is a proper understanding of how such neuron-level behaviour would emerge during backpropagation-based training. Indeed, it is interesting to note that backpropagation performs (stochastic) gradient descent, which is a very general optimization algorithm that was not specifically designed for neural network training. In contrast with other learning algorithms, which embrace the neural network structure by repeating neuron-level mechanisms, gradient descent considers neural networks as black box differentiable functions containing an unstructured collection of parameters that all try to optimize one global loss function. In other words, the decomposition of neural networks in hidden neurons is completely ignored by gradient descent's learning rule.

Intrigued by the considerable gap between backpropagation's basic principles and the commonly held view that hidden neurons learn to detect semantic concepts, this paper's goal is to explore experimentally what kind of neuron-level

*Both authors are FNRS research fellows.

dynamics emerge during backpropagation-based training. Instead of studying training from the perspective of the network's parameters, as is commonly done, we'll study it from the perspective of a neuron. More precisely, we'll monitor both the partial derivatives of the loss w.r.t the neurons' outputs (because they reveal how the neurons' outputs should be modified for each sample), and the neurons' pre-activations (to see if the neuron, through training, is able to realize the modifications provided the inputs it receives). These two signals are illustrated in Figure 1.

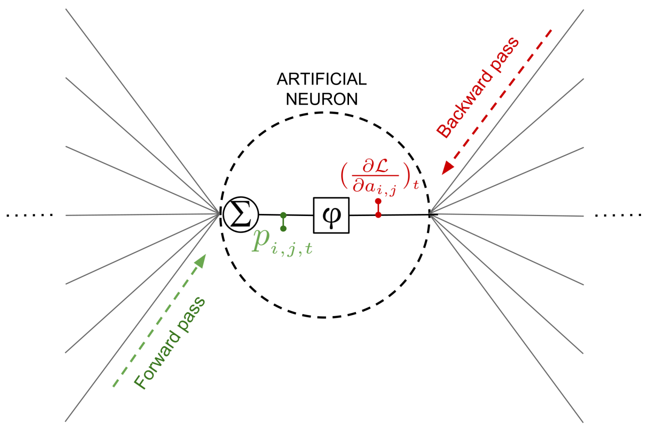

Fig. 1: Illustration of an artificial hidden neuron inside a network. φ and $\mathcal{L}$ denote the activation function and the training loss respectively. The indices i, j, t stand for the neuron, input sample and training step index respectively. In this paper, we monitor both the partial derivative of the loss w.r.t the neuron's activation $(\frac{\partial \mathcal{L}}{\partial a_{i,j}})_t$, and the neuron's pre-activation $p_{i,j,t}$.

2 Experimental setup

The studied network is a 2-layer multi-layer perceptron (MLP) with 512 neurons in both hidden layers, which is trained on MNIST [6]. We moreover use a 0.5 dropout rate [7] after each layer. In addition to the ReLU activation function [8], we also analyse a version of the MLP with the sigmoid activation function and a version without any non-linear activation function. Through the paper, we will repeatedly refer to specific layers of the network. We simply refer to the two fully-connected layers as dense1-act and dense2-act, act being replaced by the used activation function (relu, sigmoid or linear).

We train the network for 50 epochs, using a learning rate of $1e^{-1}$ for the ReLU variant, $1e^{-3}$ for the linear variant and 1 for the sigmoid variant. The batchsize is 128 and we use the normalized initialization scheme described in [9].

3 Derivatives suggest that neurons classify samples into two categories

During training, but in a separate process, we record for 30.000 input samples the partial derivative of the loss with respect to the output of randomly selected neurons (cfr. $(\frac{\partial \mathcal{L}}{\partial a_{i,j}})_t$ in Figure 1) on a regular basis (every 10 batches, leading to 2350 recordings). For each (input sample, neuron) pair, we compute the average sign of the partial derivatives over the recordings at the different training steps. This value tells us whether an increased activation generally benefits (negative average) or penalizes (positive average) the classification of the sample.[1]

Figure 2 shows, for six randomly selected neurons from different layers, the histograms of the computed average signs. As one can see, the average partial derivative sign is either 1 or -1 for most of the samples, which indicates that the derivative sign doesn't change at all through the training. The activation of a sample, in a specific neuron, should thus be either always increased or always decreased throughout training to improve its prediction. This is exactly the behaviour you would expect in the output of a binary classifier trying to separate two categories. Since around half of the activations have positive derivatives and the other half negative ones, a neuron seemingly tries to classify the inputs into two categories of nearly equal size.

4 Watching neurons learn

The derivatives strongly indicate that a neuron, during backpropagation-based training, tries to separate two categories of inputs. Does this effectively happen during training? We assign each sample to a category based on its average activation partial derivative sign, and see how both categories' pre-activations evolve across the recordings. Categories are named 'low' and 'high' for positive and negative average derivatives respectively. Figure 3 shows the results for a neuron in dense2-relu, dense2-sigmoid and in dense2-linear. We observe that indeed, both categories are being separated during the training procedure. Supplementary visualizations are available in video format on `https://www.youtube.com/channel/UC5VC20umb8r55sOkbNExB4A`.

5 The hidden neuron's ability to create subtasks

Our analysis currently ignores the impact of activation functions on the neuron-level dynamics. Here, we present one role activations functions might play,

[1]Due to the use of float32 precision, zero partial derivatives appear at some point in training when the sample is correctly classified with high confidence. Since the signs of these values are not relevant, they are ignored when the average sign is calculated.

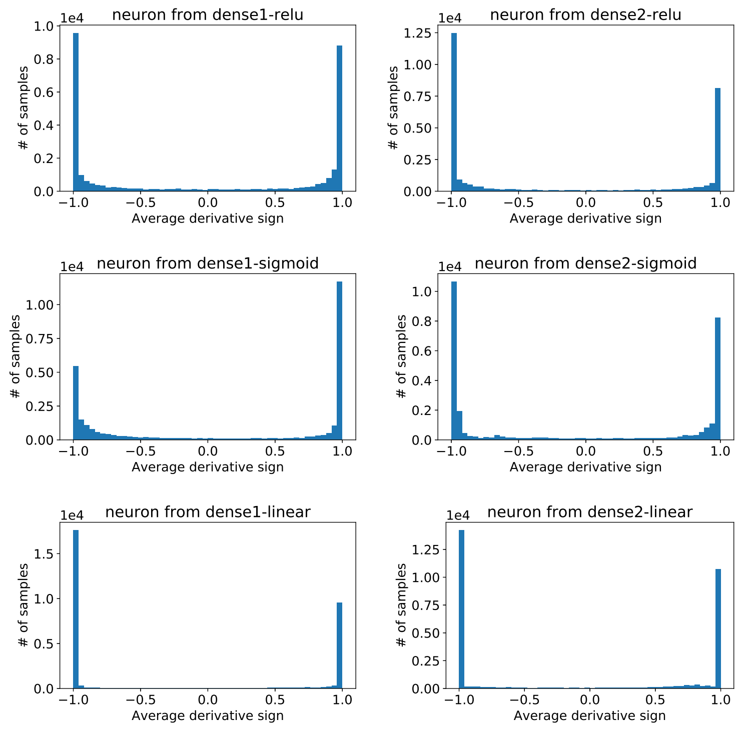

Fig. 2: The figures show the histograms of the average sign of partial derivatives of the loss with respect to activation of samples, as collected over training for a random neuron in six different layers (cfr. $(\frac{\partial \mathcal{L}}{\partial a_{i,j}})_t$ in Figure 1). An average derivative sign of 1 means that the derivative with respect to the neuron's activation for this sample was positive in all the recordings performed during training. We clearly observe two distinct categories: some sample activations should always be increased, others always decreased. This reveals that the neuron receives consistent information about how to affect the activation of a sample, allowing it to act as a binary classifier. As detailed in Section 2, the experiment was performed on a network trained on MNIST (with ReLU, sigmoid or linear activation functions).

which we call *subtask creation*. As one can see in the last column of Figure 3, the neurons we studied are not able to fully separate the two categories that were assigned to them by the backpropagated derivatives. Instead of failing at trying to separate them completely, another pertinent strategy for a neuron would be to focus on the classification of a subset of the samples. The subtask so created would be easier for the neuron to realise with high precision, and the classification of the neglected samples would hopefully be managed by other neurons of

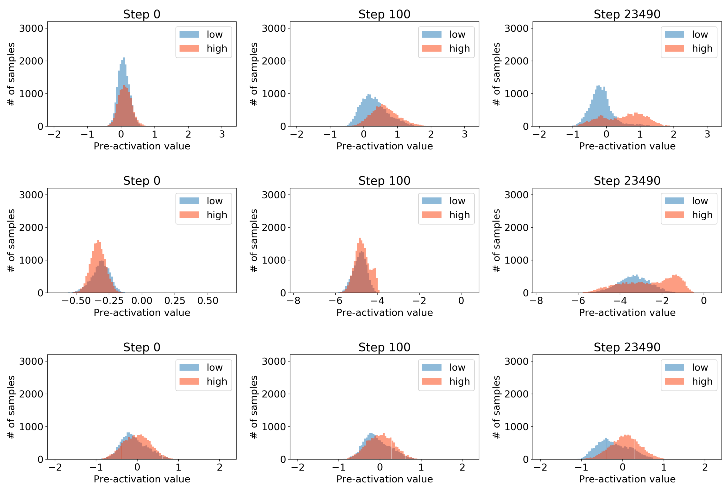

Fig. 3: Evolution of the pre-activation distributions across training. Plots correspond to one neuron from dense2-relu (*first row*), dense2-sigmoid (*second row*) and dense2-linear (*third row*). Pre-activations are separated into two categories, high and low, based on the average partial derivative sign over training of their corresponding activation (cfr. Figure 2). We can see that both categories are being separated during training, providing supplementary evidence that hidden neurons behave like binary classifiers during backpropagation-based training.

the same layer, leading to a divide-and-conquer strategy.

The ReLU, which is currently the most successful activation function, seems particularly adapted for the emergence of such strategy. Indeed, a ReLU neuron cancels the derivatives of all samples with a negative pre-activation, and thus ignores them during training of all its incoming connections/weights. Figure 4 (left) shows this mechanism in action in a neuron of the MLP network we study. Indeed, we observe that at the end of training, samples with positive pre-activations are classified (in the low and high categories associated to the neuron) with much higher accuracy than samples with negative pre-activations. In our experiments, we observed a similar behaviour in sigmoid neurons when all pre-activations are negative (cfr. Figure 4 (right)). In such case, the more the pre-activation of a sample is negative, the more the sigmoid neuron shrinks its derivative, which enables the neuron to focus on samples with higher pre-activations.

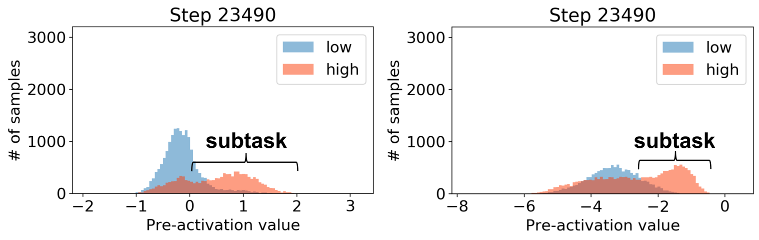

Fig. 4: A ReLU (*left*) and a sigmoid (*right*) neuron are able to focus on the classification of a subset of the samples with high precision by canceling/shrinking derivatives associated to samples in the saturated regions of the activation function.

6 Conclusion

Intrigued by the gap between the basic principles of the backpropagation algorithm and the intuitions that have successfully guided neural network research since their origin, our paper investigates experimentally if neuron-level mechanisms emerge during backpropagation-based training. Our first results, based on a simple MLP network, are conclusive: we observe that hidden neurons behave like binary classifiers, dividing the inputs (or a subset of them, depending on the activation function) into two categories of quasi equal size. Many questions remain, and further validation of our claim is needed. Our hope is that this paper will foster interest around this original approach to apprehend the inner workings of deep learning.

References

[1] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

[2] Donald Olding Hebb. *The Organization of Behavior: a neuropsychological theory.* Wiley & Sons, New York, 1949.

[3] R Quian Quiroga, Leila Reddy, Gabriel Kreiman, Christof Koch, and Itzhak Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102, 2005.

[4] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[5] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network Dissection: Quantifying Interpretability of Deep Visual Representations. In *CVPR*, 2017.

[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998.

[7] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[8] Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *ICML*, pages 807—-814, 2010.

[9] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.