Local Unsupervised Learning for Image Analysis

Anonymous Author(s)
Affiliation
Address
email

Abstract

Local Hebbian learning is believed to be inferior in performance to end-to-end training using a backpropagation algorithm. We question this popular belief by designing a local algorithm that can learn convolutional filters on image datasets. These filters combined with patch normalization and very steep non-linearities result in a good classification accuracy for shallow networks trained locally, as opposed to end-to-end. The filters learned by our algorithm contain both orientation selective units and unoriented color units, resembling the responses of pyramidal neurons located in the cytochrome oxidase "interblob" and "blob" regions in the primary visual cortex of primates. It is shown that convolutional networks with patch normalization significantly outperform standard convolutional networks on the task of recovering the original classes when shadows are superimposed on top of standard CIFAR-10 images. Patch normalization approximates the retinal adaptation to the mean light intensity, important for human vision. All these results taken together suggest a possibility that local unsupervised training might be a powerful tool for learning general representations (without specifying the task) directly from unlabeled data.

1 Introduction

Local learning, motivated by Hebbian plasticity, is usually believed to be inferior in performance to gradient-based optimization, for example a backpropagation algorithm. Common arguments include the following ideas. Feature detectors in the early layers of neural networks should be specifically crafted to solve the task that the neural network is designed for. Thus some information about the performance of the network or the loss functions must be available during learning of the early layer weights. Making random changes in weights and accepting those changes that improve performance, as evolutionary algorithms do, is very inefficient in large networks. Gradient-based optimization seems to converge to the desired solution faster than methods which do not rely on the gradient.

At the same time, modern neural networks are heavily overparametrized, which means that there are many combinations of weights that lead to a good generalization performance. Thus, there is an appealing idea that this manifold of "good" weights might also be found by a purely local learning algorithm that operates directly on the input data without the information about the output of the network or the task to be performed.

A variety of local learning algorithms relying only on bottom-up propagation of the information in neural networks have been recently discussed in the literature. A recent paper [1], for example, proposed a learning algorithm that is local and unsupervised in the first layer. It manages to learn useful early features necessary to achieve a good generalization performance, in line with networks trained end-to-end on simple machine learning benchmarks. The limitation of this study is that the proposed algorithms were tested only in fully connected networks, only on pixel permutation invariant tasks, and only on very simple datasets: pixel permutation invariant MNIST and CIFAR-10. An additional limitation of study [1], which is not addressed in the present work, is that they studied neural networks with only one hidden layer.

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

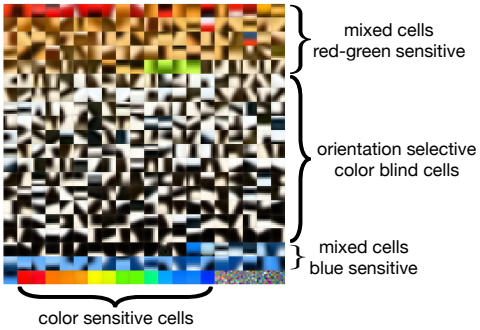

Figure 1: Filters of size $W = 8$ pixels, trained using the proposed algorithm. The filters are ordered to emphasize the differences between several groups of cells. The last five units in the last row did not learn any useful representation. They can be deleted from the network.

Our main contributions are the following. Based on the open source implementation of the algorithm proposed in [1], we designed an unsupervised learning algorithm for networks with *local* connectivity. We wrote a fast CUDA library that allows us to quickly learn weights of the convolutional filters at scale. We propose a modification to the standard convolutional layers, which includes patch normalization and very steep non-linearities in the activation functions. This allows us to match the performance of networks of similar size and architecture trained using the backpropagation algorithm end-to-end on CIFAR-10. The usefulness of patch normalization is illustrated by designing an artificial test set from CIFAR-10 images that are dimmed by shadows. The network with patch normalization outperforms the standard convolutional network by a large margin on this task. We also note that the filters learned by our algorithm show a well pronounced separation between color-sensitive cells and orientation-selective cells. This computational aspect of the algorithm matches nicely with a similar separation between the stimulus specificity of the responses of neurons in blob and interblob pathways, known to exist in the V1 area of the visual cortex.

2 Learning Algorithm and Network Architecture

During training each input image is cut into small patches, which are presented to the learning algorithm proposed in [1]. The output of this algorithm is a matrix of weights $M_{\mu i}$, which is used as weights of the convolutional filters with two important modifications. First, each patch v_i of the image is normalized to be a unit vector before taking the dot product with the weight matrix. Given that the rows of the weight matrix themselves are unit vectors, as explained in [1], the dot product between the weight and the patch is a cosine of the similarity between the two. Second, the result of the dot product is passed through a very steep non-linearity - rectified power function

$$f(x) = \Big[\text{ReLU}(x)\Big]^n \tag{1}$$

where the power n is a hyperparameter of that layer. Since the maximum input to our units is 1.0, the main effect of using a large power in the input-output relation is to suppress the outputs from the many weakly driven units, making the overall representation more sparse. The hyperparameter n tunes this sparseness. We call these slightly unusual convolutional layers NNL-CONV layers (normalized non-linear convolutional layers), in order to distinguish them from the standard ones, denoted CONV in this paper. The standard CONV layers do not use per-patch normalization and use ReLU as an activation function.

3 Evaluation of the model on CIFAR-10 dataset

The proposed algorithm was applied to images from CIFAR-10 dataset to learn the weights of the NNL-CONV filters. They are shown in Fig.1. Each small square corresponds to a different channel (hidden unit) and is shown by projecting its corresponding weights into the image plane. The connection strength to each pixel has 3 components corresponding to the RGB color image. Thus, black color corresponds to synaptic weights that are either equal to zero, or negative; white color corresponds to weights that are large and have approximately equal values in all three RGB channels; blue color corresponds to weights that have large weights connected to blue neurons, and small or zero weights connected to red and green neurons, etc. As is clear from this figure, the resulting weights show a diversity of features of the images, including line detectors, color detectors, and detectors of more complicated shapes.

Guided by these examples, one can make the following qualitative observations. First, the majority of the hidden units are black and white, having no preference for R,G, or B color. These units respond strongly to oriented edges, corners, bar ends, spots of light, more complicated shapes, etc. Second, there is a significant presence of hidden units detecting color. Those neurons tend to have a smaller preference for orientation selectivity, compared to black and white units.

In order to test the quality of these learned filters with respect to the generalization performance, they were used as the weights of a simple network with one NNL-CONV layer. Consider for example

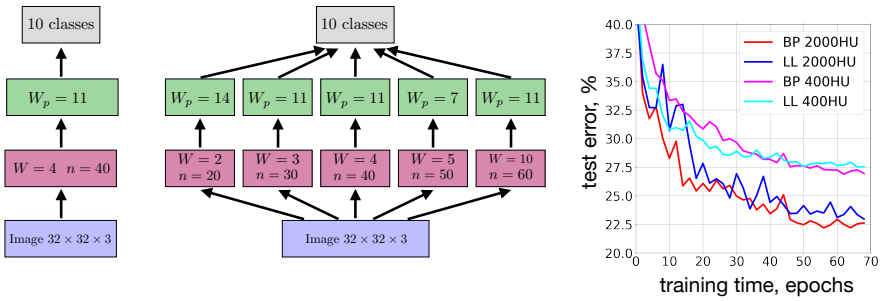

Figure 2: (Left) A simple architecture with an image projected to the NNL-CONV layer (shown in red), max pooling layer (shown in green), and a softmax classifier (gray); Parameters: convolutional window size $W = 4$ pixels; the power of the activation function is $n = 40$; number of channels $K = 400$, pooling window size $W_p = 11$ pixels (all the hyperparameters were determined on the validation set). (Middle) An architecture with blocks of different W and n. (Right) The test set error of the two networks shown in the left and middle panels together with their end-to-end counterparts as a function of supervised training time.

the architecture shown in Fig. 2 (left). In this architecture a NNL-CONV layer is followed by a max pooling layer and then a fully connected softmax classifier. The weights of the NNL-CONV layer were fixed, and given by the output of the proposed algorithm. The max-pooling layer does not have any trainable weights. The weights of the top layer were trained using the gradient decent based optimization (Adam optimizer with the cross-entropy loss function). The accuracy of this network was compared with the accuracy of the network of the same capacity, with NNL-CONV layers replaced by the standard CONV layers, trained end-to-end using Adam optimizer. The results are shown in Fig. 2 (right). Here one can see how the errors on the held-out test set decrease as training progresses. Training time here refers to the training of the top layer classifier only in the case of the NNL-CONV network, and training of all the layers (convolutional layer and the classifier) in the case of the standard CONV network trained end-to-end. For the simple network on the left the error of the locally trained network is 27.80%, the error of the network trained end-to-end is 27.11%.

In order to achieve a better test accuracy it also helps to organize the NNL-CONV layer as a sequence of blocks with various sizes of the window W, like in Fig. 2 (middle). This helps to detect features of different scales. As above, the performance of this network trained using the proposed local learning algorithm is compared with the performance of a similar size network trained end-to-end. The results are shown in Fig. 2 (right). Both networks show errors $\approx 23\%$.

The main conclusion here is that the networks with filters obtained using the local unsupervised algorithm achieve almost the same accuracy as the networks trained end-to-end. This result is at odds with the common belief that the first layer feature detectors should be crafted specifically to solve the narrow task specified by the top layer classifier. Instead, this suggests that a general set of weights of the first layer can be learned using the local bottom-up unsupervised training, and that it is sufficient to communicate the task only to the top layer classifier, without adjusting the first layer weights.

4 Color Sensitivity, Orientation Selectivity and Cytochrome Oxidase Stain

An interesting anatomical feature of the primary visual cortex is revealed when cells are stained with a cytochrome oxidase enzyme. This staining shows a pattern of blobs and interblob regions. In a famous set of experiments [2] it was discovered that the cells in the interblob regions are highly orientation selective and respond to luminance rather than to color, while the neurons inside the blobs respond to colors and are insensitive to orientation.

There are many important details and subtleties of these experiments that are not discussed in this paper, e.g. the neocortical visual areas of the primate brain have six anatomical layers many clearly divided into sublayers; neurons in different layers have different response properties; the response properties also vary smoothly within a layer. These facts limit the usefulness of describing how a "typical" pyramidal cell in V1 responds, especially to strong natural stimuli. However, the literature supports the qualitative assertion that there exist a segregation of orientation selective and color processing cells [2]. A segregation resembling the one discussed above can be seen in the learned filters shown in Fig. 1, which can be interpreted as preferred stimuli of the hidden neurons.

5 Patch Normalization, Retinal Adaptation, and Shadows

Natural scenes can have a several thousand fold range of light intensity variation [3]. At the same time, an 8-bit digital camera has only 256 possible intensity values. In order to cope with this huge variation of light intensities two separate systems exist in biological vision: a global control based on changing the size of the pupil, and a local adaptation on the retina. The latter, being the dominant one, enables the retina to have a good signal to noise ratio in both sun and shadow regions of a visual scene [3]. Patch normalization, which is essential for a good performance of our algorithm, can be

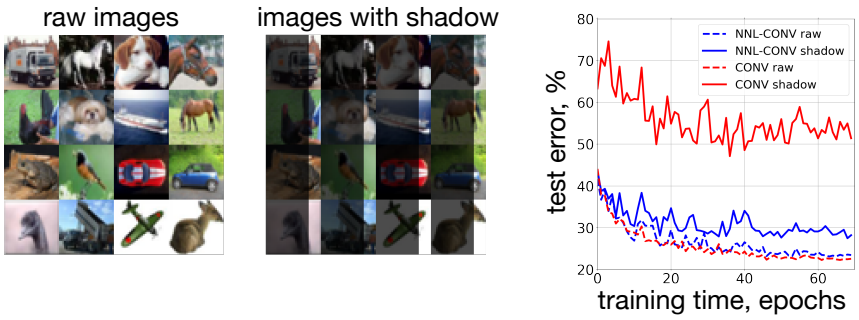

Figure 3: (Left) Randomly selected images from CIFAR-10 dataset. (Middle) Same images, part of each image is multiplied by an arbitrary parameter 0.3 pixelwise. This imitates a shadow. (Right) Errors on the test set for the model with local patch normalization (blue) and a standard convolutional network (red). Raw images - dashed lines, shadowed images - solid lines.

thought of as a mathematical formalization of the local circuitry on the retina that is responsible for this adaptation. Although we do not have a dataset with images of real scenes in various lighting conditions, it can be reasonably emulated by multiplying images from CIFAR-10 dataset pixelwise by a function $I(x, y)$, which changes between zero and one. Patch normalization discards the overall normalization constant in every patch, which is strongly dependent on the light conditions, and focuses chiefly on the shape of an object that a given unit sees.

Examples of images constructed this way are shown in Fig. 3, where $\approx 80\%$ of each image was covered by a shadow having $I(x, y) = 0.3$. Human can see and correctly classify these "shadowed" images. Two networks: standard CONV net trained end-to-end, and NNL-CONV net trained as described in section 3 were trained on raw CIFAR-10 images, but tested on the shadowed images. Both networks had exactly the same architecture, shown in Fig. 2 (middle). The results are shown in Fig. 3 (right). While the errors on the raw CIFAR-10 images are approximately the same for the two networks ($\approx 23\%$), the error on the shadowed images of the NNL-CONV net is much lower ($\approx 28\%$) compared to the error of the standard CONV net (more than $\approx 50\%$). This illustrates that patch normalization can be a useful tool for dealing with images having large differences in light intensity (for example coming from shadows), without having images with these kinds of shadows in the training set.

References

[1] Krotov, D. and Hopfield, J., 2019. Unsupervised learning by competing hidden units, Proceedings of the National Academy of Sciences, 116 (16) 7723-7731; DOI: 10.1073/pnas.1820458116

[2] Livingstone, M.S. and Hubel, D.H., 1984. Anatomy and physiology of a color system in the primate visual cortex. Journal of Neuroscience, 4(1), pp.309-356.

[3] Dunn, F.A., Lankheet, M.J. and Rieke, F., 2007. Light adaptation in cone vision involves switching between receptor and post-receptor sites. Nature, 449(7162), p.603.