Understanding structure of parameters in neural networks

Emily Denton DENTON@CS.NYU.EDU

Wojciech Zaremba ZAREMBA@CS.NYU.EDU

Joan Bruna BRUNA@CS.NYU.EDU

Rob Fergus FERGUS@CS.NYU.EDU

Abstract

1. Introduction

Main criticism of neural networks is in the lack of interpretability of their results. This is serious issue, as it hinders process of knowledge injection, and makes difficult to predict their behaviour in the new conditions (not presented during training). This issue is the consequence of superficial understanding of computation underlaying neural networks. We do attempt in direction of understanding computation behind neural networks.

We show that learnt parameters of neural networks have very rich low-dimensional linear structure. We are not able to explain reason for its presents. However, such a structure has a far-reach consequences in (i) understanding computation of neural networks, (ii) regularization of neural networks, (iii) optimization, (iv) accelerating testing, (v) re-parametrization. We describe how such structure can be extracted and visualized. We show it for state-of-the-art object recognition models (~ 60 mln. learnable parameters). Moreover, we explore consequences of this low-rank linear structure for simpler models.

2. Related Work

Our low-rank approximations get inspired by work of (Denil et al., 2013) on redundancies in neural networks. They shown that based on small fraction of weights (e.g. $\sim 5\%$), one can accurately predict rest of weights. This indicates that neural networks are heavily over-parametrized. All the methods presented here focus on exploiting linear structure of such over-parametrization.

3. Low Rank Approximations

In this section, we give theoretical background on low-rank approximations. First, we discuss simplest setting, which is for matrices (two dimensional tensors). Further, we move to approximation of 4-dimensional tensors with 2 convolutional (spacial) dimensions.

3.1. Matrix Low Rank Approximation

Let's consider input $X \in \mathbb{R}^{n \times m}$, and matrix of weights $W \in \mathbb{R}^{m \times k}$. W might have a low-rank (many eigenvalues close to zero), and matrix multiplication of X with W might be very close to the matrix multiplication with approximation of W.

Every matrix can be expressed using singular value decomposition:

$W = USV^T$, where $U \in \mathbb{R}^{m \times m}, S \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{k \times k}$

S is has eigenvalues on the diagonal, and apart from it has zeros. W can be approximated by using t most significant eigenvalues from S. We can write approximation as

$$\hat{W} = \hat{U}\hat{S}\hat{V}^T, \text{ where } \hat{U} \in \mathbb{R}^{m \times t}, \hat{S} \in \mathbb{R}^{t \times t}, \hat{V} \in \mathbb{R}^{t \times k}$$

3.2. Tensor Low Rank Approximations

In typical object recognition architectures, the convolutional tensors resulting from the training exhibit strong redundancy and regularity across all its dimensions. A particularly simple way to exploit such regularity is to linearly compress the tensors, which amounts to finding low-rank approximations.

Convolution kernels can be described as a 4-dimensional tensors. Let $W \in \mathbb{R}^{C \times X \times Y \times F}$ be convolutional kernel. C is input number of feature maps (or colors), X, Y are special dimensions over which we compute convolution, and F is the target number of feature maps. Let $I \in \mathbb{R}^{C \times N \times M}$ denote an input signal. A generic convolutional layer is defined as

$$I * W(f,x,y) = \sum_{c=1}^{C} \sum_{x'=-X/2}^{X/2} \sum_{y'=-Y/2}^{Y/2} I(c, x-x', y-y')W(c,x',y',f)$$

Low-rank approximation of tensors is very similar to low-rank approximation of matrices. However, first we have to choose 2 dimensions with respect to which we are approximating, and treat rest of tensor as it would be a matrix.

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

Degree	Grammar size	Time (s)
2	5	17
3	15	188
4	48	2535
5	139	31320
6	437	434681

Table 1.

3.2.1. Monochromatic filters

For generic convolution there are connections between all output feature maps, and all input feature maps. However, one could imagine that this connections in trained network in some basis could become sparse.

4. Consequences

4.1. Understanding computation of neural networks

Low-rank analysis can reveal underlaying computation. Moreover, it can be a corner stone in understanding why neural networks works. In this section, we treat this topic briefly (it is focus of a future work), and we show evaluation for a simple network trained to classify MNIST.

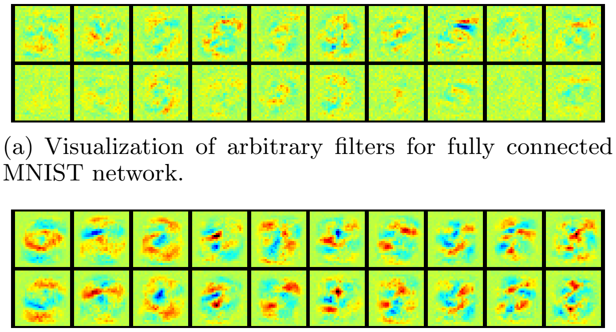
We took networks considered previously during development of dropout (Hinton et al., 2012). We look on trained filters from fully-connected layers from perspective of low-rank analysis. First, we trained fully connected 768-800-800-10 with, and without 50% dropout until convergence. Figure 1 compares arbitrary filters of such networks versus left-singular values. Moreover, it can be quantified that left-singular values for dropout network are better than for non dropout network.

Edges of MNIST dataset are almost empty, and pixels over there have very small discriminative power. Classifier, which base its decision on values there is likely to overfit. Robust classifier should match strokes, rather than small details. We wanted to measure how much dropout based, and non-dropout based classifier relies on values from edges of images. We took the same 768-800-800-10 networks, and computed average norm over strips from the top, bottom, left side, and right side of the filters, and respectively of the left-singular values of filters. We took 10 left-singular values, and all the filters. Table 1 presents numerical results.

4.2. Regularization

4.3. Speeding up testing

Training large neural networks (NN) takes weeks, or even months. This can hinder research, and there have been extensive effort devoted to speed up training procedure. However, resource-wise from perspective of companies executing neural networks on internet-scale data (e.g. annotating images), this is not the main cost. Major cost is in the final stage, where network is evaluated on the target data, which is present in quantities of billions. We focus here on speeding up evaluation of *trained* NN, which directly maps to the cost of executing NN on internet-scale data.



(a) Visualization of arbitrary filters for fully connected MNIST network.

(b) Visualization of left singular vectors for fully connected MNIST network.

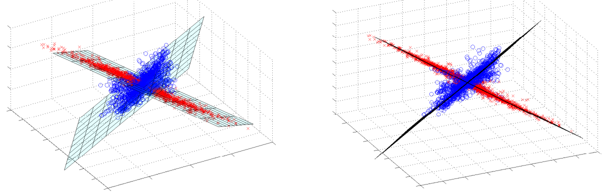
Figure 1. Visualization of filters for fully connected network trained on MNIST dataset. For every subfigure, first row is a network trained with dropout, and the second is trained without.

5. Numerical Experiments

5.1. MNIST

5.2. Imagenet

5.2.1. First layer analysis



(a) aaa

(b) aaa

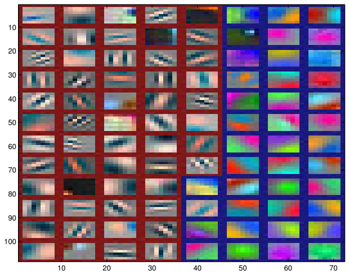
Figure 2. Experiment results

5.2.2. Further layers analysis

6. Discussion

References

Denil, Misha, Shakibi, Babak, Dinh, Laurent, Ranzato, Marc'Aurelio, and de Freitas, Nando. Pre-



(a) aaa

Figure 3. First layer filters grouped by subspace cluster

dicting parameters in deep learning. *arXiv preprint arXiv:1306.0543*, 2013.

Hinton, Geoffrey E, Srivastava, Nitish, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Le, Q. V., Ngiam, J., Chen, Z., Chia, D., Koh, P. W., and Ng, A. Y. Tiled convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2010.

Le, Quoc V, Ranzato, Marc'Aurelio, Monga, Rajat, Devin, Matthieu, Chen, Kai, Corrado, Greg S, Dean, Jeff, and Ng, Andrew Y. Building high-level features using large scale unsupervised learning. *arXiv preprint arXiv:1112.6209*, 2011.

Lowe, David G. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pp. 1150–1157. Ieee, 1999.

Mathieu, Michael, Henaff, Mikael, and LeCun, Yann. Fast training of convolutional networks through ffts. *arXiv preprint arXiv:1312.5851*, 2013.

Vanhoucke, Vincent, Senior, Andrew, and Mao, Mark Z. Improving the speed of neural networks on cpus. In *Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop*, 2011.

Zeiler, Matthew D, Taylor, Graham W, and Fergus, Rob. Adaptive deconvolutional networks for mid and high level feature learning. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2018–2025. IEEE, 2011.