Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation

Abstract

1. 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). We then consider the approximation of 4-dimensional tensors of convolution weights.

1.1. Matrix Low Rank Approximation

Let $X \in \mathbb{R}^{n \times m}$ denote the input to a fully connected layer of a neural network and let $W \in \mathbb{R}^{m \times k}$ denote the weight matrix for the layer. Matrix multiplication, the main operation for fully connected layers, costs $O(nmk)$. However, W is likely to have a low-rank structure and thus have several eigenvalues close to zero. These dimensions can be interpreted as noise, and thus can be eliminated without harming the accuracy of the network. We now show how to exploit this low-rank structure and to XW much faster than $O(nmk)$.

Every matrix $W \in \mathbb{R}^{m \times k}$ can be expressed using singular value decomposition:

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

S is has eigenvalues on the diagonal, and zeros elsewhere. W can be approximated by choosing the t largest eigenvalues from S. We can write the approximation as

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

Now the computation $X\tilde{W}$ can be done in $O(nmt + nt^2 + ntk)$, which, for sufficiently small t can be significantly smaller than $O(nmk)$.

1.2. Tensor Low Rank Approximations

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

Convolution weights can be described as a 4-dimensional tensor. Let $W \in \mathbb{R}^{C \times X \times Y \times F}$ denote such a weight tensor. C is the number of number of input channels, X and Y are the special dimensions of the kernel, and F is the target number of feature maps. Let $I \in \mathbb{R}^{C \times N \times M}$ denote an input signal where C is the number of input maps, and N and M are the spatial dimensions of the maps. The computation performed by a generic convolutional layer is defined as

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

We would like to approximate W with a low rank tensor that has a particular structure that allows for a more efficient computation of the convolution. The approximations will be more efficient in two senses: both the number of floating point operations required to compute the convolution output and the number of parameters that need to be stored will be dramatically reduced.

A standard first convolutional layer will receives three color channels, typically in RGB or YUV space, as input whereas later hidden layers typically receive a much larger number of feature maps that have resulted from computations performed in previous layers. As a result, the first layer weights often have a markedly different structure than the weights in later convolutional layers. We have found that different approximation techniques are well suited to the different layers which we now describe. The first approach, which we call the monochromatic filter approximation can be applied to the weights in the first convolutional layer. The second approach, which we call the bi-clustering approximation, can be applied to later convolutional layers where the number of input and output maps is large.

1.3. Monochromatic filters

Let $W \in \mathbb{R}^{C \times X \times Y \times F}$ denote the first convolutional layer weights of a trained network. The number of input channels, C, is 3 and each channel corresponds to a different color component (either RGB or YUV).

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

We have found that the color components of weights from a trained convolutional neural network have low dimensional structure. In particular, the weights can be well approximated by projecting the color dimension down into a 1D subspace. Figure ?? shows the original first layer convolutional weights of a trained network and the weights after the color dimension has been projected into 1D lines.

The approximation is computed as follows. First, for every output feature, f, we consider consider the matrix $W_f \in \mathbb{R}^{C \times XY}$, where the spatial dimensions have been combined, and find the singular value decomposition,

$$W_f = U_f S_f V_f^\top$$

where $U_f \in \mathbb{R}^{C \times C}, S_f \in \mathbb{R}^{C \times XY}, V_f \in \mathbb{R}^{XY \times XY}$. We then take the rank-1 approximation to W_f

$$\tilde{W}_f = \tilde{U}_f \tilde{S}_f \tilde{V}_f^\top$$

where $\tilde{U}_f \in \mathbb{R}^{C \times 1}, \tilde{S}_f \in \mathbb{R}, \tilde{V}_f \in \mathbb{R}^{1 \times XY}$.

This approximation corresponds to shifting from C color channels to 1 color channel for each output feature. We can further exploit the regularity in the weights by sharing the color component basis between different output features. We do this by clustering the F left singular vectors, $\tilde{U}_f$, of each output feature f into C' different clusters, where C' is much smaller than F. Then, for each output feature, f, that is assigned to cluster c, we can approximate W_f with

$$\tilde{W}_f = U_c \tilde{S}_f \tilde{V}_f^\top$$

where $U_c \in \mathbb{R}^{C \times 1}$ is the cluster center for cluster c and $\tilde{S}_f$ and $\tilde{V}_f$ as as before.

This low-rank approximation allows for a more efficient computation of the convolutional layer output. By decomposing the approximated weights into two tensors. Let $W_C \in \mathbb{R}^{C' \times C}$ denote the color transform matrix where the rows of W_c are the cluster centers $U_c^\top$. Let $W_{mono} \in \mathbb{R}^{X \times Y \times F}$ denote the monochromatic weight tensor containing $\tilde{S}_f \tilde{V}_f^\top$ for each of the F output features. First, we transform the input signal, $I \in \mathbb{R}^{C \times N \times M}$

1.4. Bi-clustering of hidden layer weights