Sparse Deep Learning

Merging double sparsity with Deep NN

Elad Hoffer

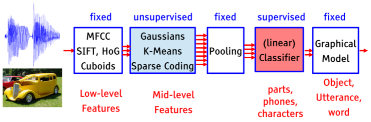
Final Project - Sparse and redundant representations
March 2015

Outline

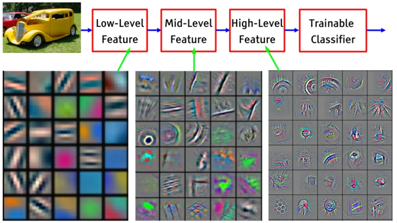
Outline

Deep Learning

Deep learning is a set of machine learning algorithms based on neural networks. These algorithms mostly concern multi-layered hierarchies (hence "Deep") that aim at finding abstract features representations from vast information.



Traditional ML



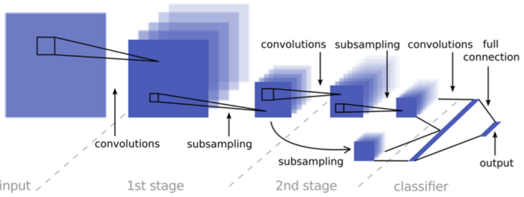
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

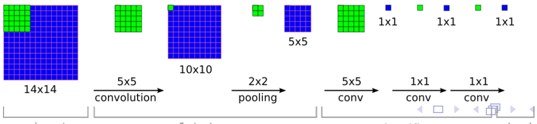
Deep Learning

Convolutional Neural Networks

Deep convolutional neural network represent the main approach of deep learning in computer vision tasks [3].

- Trainable weight kernels produce their output value by cross-correlating with input data.
- A non-linearity (ReLU, Sigmoid) is applied between layers
- Pooiling layers alow diminesion-reduction+invariance





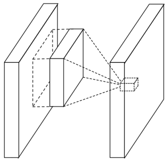
Network-In-Network Model

Min Lin et al. 2013 [1]

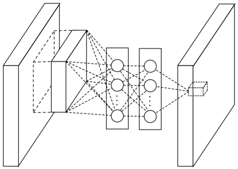
One recent model of convolutional network is the *Network-In-Network* model. This model replaces the usual convolutional layer building-block, with the *conv-mlp layer* , which is essentially:

- A $k \times k$ convolution layer
- Followed by two successive 1×1 convolutional layers.

This creates feature maps that are linear combinations of the matching pixels in previous layer with a non-linear *ReLU* activation layer.



Convolution layer



NiN layer

Outline

Dictionary Learning

For many computer-vision tasks, a dictionary of visual atoms is used to represent the data.

$$x = D\gamma$$

where x is the represented data, D is the underlying dictionary, and γ are the dictionary coefficient.

Two main approaches for choosing the dictionary D:

- Predefined structured dictionaries (DCT, Wavelets, Curvelets). No learning needed, low compute time due to structure.
- Using machine learning techniques to train the needed dictionary D. Requires training and longer inference time, but usually achieves better results on the final objective.

Double Sparsity Model

Rubinstein, Zibulevsky, Elad. 2010

The double sparsity model suggestes a tradeoff between the two. By using a pre-defined structured dictionary Φ , and an additional set of sparse coefficients A, the needed dictionary D is formed such that

$$D = \Phi A$$

where A is constrained to be sparse - $\|a_i\|_0^0 < p$ for some p.

The double sparsity model allows us to learn useful, adaptive, dictionaries with good results, but with a lower computational cost that accompanies structured dictionaries.

Outline

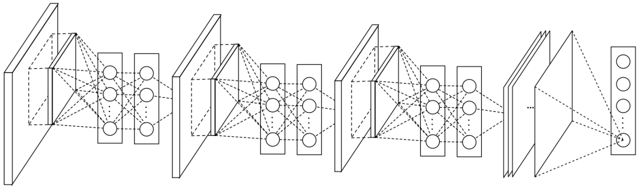
Putting it all together

We can now see how the NiN architecture can fit nicely into the double-sparsity model to produce partially structured kernels for deep convolutional networks. By using

- Φ_c - Predefined and fixed DCT dictionary atoms of $k \times k$
- W_l - trainable weights

$$D_l = \Phi_c W_l$$

Thus we are effectively creating a double-sparsity induced deep network, where we can control the sparsity of the coefficient by regularizing the weights to have low L_0 norm.



Training the network

- The network is trained on classification task using the negative-log-likelihood criterion. Regularization is introduced into the loss function to produce:

  $$Loss(x, y_t) = E_{NLL}\left(Net(x; w), y_t\right) + \frac{\mu}{2} \cdot \|w\|_2 + \zeta \cdot \|w\|_1$$

  where μ and ζ are the L_2 and L_1 regularization coefficient respectively.
  The update rule at each step is then

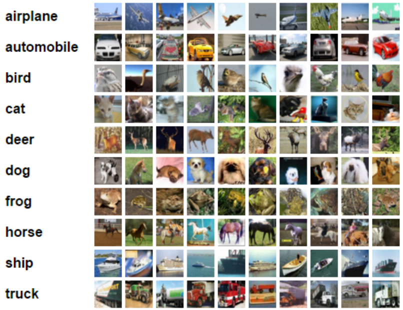
  $$w_{t+1} = w_t - \epsilon \cdot [\nabla_w E_{NLL} + \mu \cdot w + \zeta \cdot sign(w)]$$

Outline

Experiments

We experimented with 2 datasets

- *Cifar10* - 60,000 32x32 color images of 10 classes.
- *Street-View-House-Numbers (SVHN)* - 600,000 32x32 color images of house-number digits 0-9.



Cifar10 Dataset

SVHN Dataset

Results

Accuracy of the double-sparse model vs original NiN

Dataset	Original NiN model	Double-Sparsity model
Cifar10	87.6%	74.2%
SVHN	97.65%	95.47%

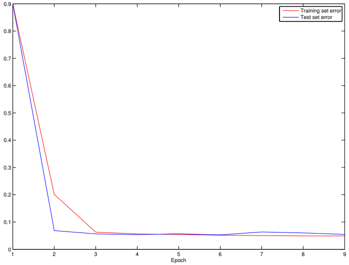
- Using the NiN with DCT kernels created a model with $\frac{1}{7}$th the number of parameters, thus reducing computation and memory footprint by an order of magnitude.
- Whereas the Cifar10 dataset suffered a significant 13% drop, the SVHN had only 2% percent drop in accuracy while using very low number of non-zero parameters.
- These results shows not only the complexity difference between the two datasets, but also the ability of this approach to achieve descent accuracy with pre-defined kernels and low number of parameters.

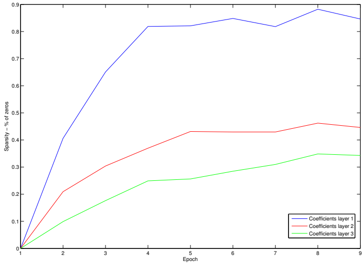
Sparsity evaluation

We can accumulate the number of zero valued parameters in the coefficient layers to judge how sparse these layers are.

Dataset	Layer #1 Sparsity	Layer #2 Sparsity	Layer #3 Sparsity
Cifar10	25.47%	10.43%	44.5%
SVHN	84.63%	44.62 %	34.28%

We can also see that we can gain sparsity (right) by continuing training after error has dropped (left).





Outline

Visualizing low-level kernels

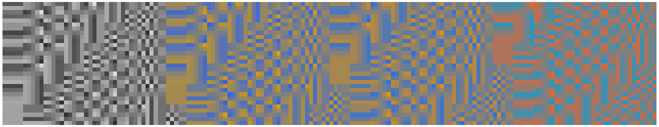
- One known way to visualize the outcome of a trained convolutional network is to view the form of kernels in its first layer.
- As this layer used to extract the most basic aspects in an image, its kernels are usually some form of gabor filters, which are known to capture low-level features of visual information such as edges, orientation, and color nuances.

We will use the first layer dictionary components, where Φ_1 is the overcomplete 5×5 DCT dictionary, and W_1 is the first set of trainable parameters.

$$D_1 = W_1^T \cdot \Phi_1, \quad W_1 \in \mathbb{R}^{F_l \times F_{l+1}}, \Phi_1 \in \mathbb{R}^{F_l \times F_l \times 5 \times 5}$$

Visualizing low-level kernels

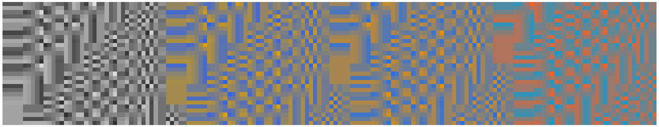
Using the Overcomplete DCT 5×5 dictionary



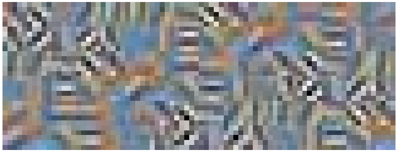
and multiplied by the learned coefficients, we get:

Visualizing low-level kernels

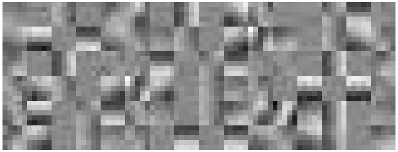
Using the Overcomplete DCT 5×5 dictionary



and multiplied by the learned coefficients, we get:



Cifar10 kernels



SVHN kernels

We can observe that the learned dictionaries reflect the complexity of each dataset - color diversity, orientation importance, richness.

Computational cost

The double sparsity layers were created by simply using a convolutional layer with pre-defined and fixed (untrainable) DCT filters. This was used to simplify implementation, but in fact a much more optimized usage can be creating by noting that

- Structured dictionaries are faster to compute - we can accelerate inference time using the known structure of the predefined dictionary [2].
- DCT kernel are separable - we can perform 1-dimensional convolution twice and allow for much better performance.
- Sparse coefficients can be translated to fewer kernels

Summary

In this work it was shown that

- The double-sparsity model can be embedded in the deep convolutional network framework, paving the way for other sparse-oriented models and techniques.
- This can potentially allow models with lower number of tunable parameters, and structured dictionaries that can be efficiently used to accelerate computations.

Further research is needed to improve upon those findings and allow better results that are competitive with state-of-the-art results of conventional deep networks.

For Further Reading

Min Lin, Qiang Chen, and Shuicheng Yan.
Network in network.
CoRR, abs/1312.4400, 2013.

Ron Rubinstein, Michael Zibulevsky, and Michael Elad.
Double sparsity: Learning sparse dictionaries for sparse signal approximation.
Signal Processing, IEEE Transactions on, 58(3):1553–1564, 2010.

Matthew D Zeiler and Rob Fergus.
Visualizing and Understanding Convolutional Networks.
arXiv preprint arXiv:1311.2901, 2013.