Paper Notes

Very Deep Convolutional Networks for Large-Scale Image Recognition

Authors of the Paper: Karen Simonyan, Andrew Zisserman

Compiled By: Dakshit Agrawal

Discussion held on 11th October 2018.

This paper introduces a uniform very deep convolutional network for large-scale image recognition. It evaluates several different models on the ImageNet dataset, all with the major emphasis on a deeper network with smaller convolutional filters (3X3). The authors note significant improvement than previous models, achieving 1st place in the localization task, and 2nd in the classification task.

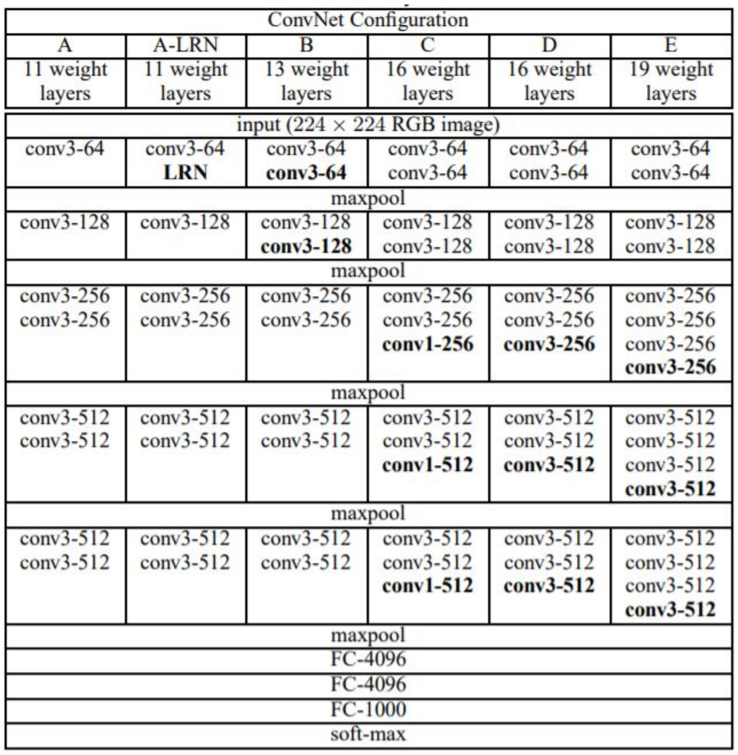
Motivation

1. Previous papers observations:
    a. **Krizhevsky et al., 2012:** Used deep convolutional networks for large-scale image recognition for first time.
    b. **Zeiler & Fergus, 2013:** Used smaller receptive window size and smaller stride of the first convolutional layer
    c. **Sermanet et al., 2014:** Training and testing the networks densely over the whole image and over multiple scales.
2. No previous major work on testing the effect of increasing depth of convolutional layers on large-scale image recognition.
3. Previous models were also very complex. This paper tries to find a uniform and more simpler model for large-scale image recognition.

Work Done in Parallel

1. **Goodfellow et al., 2014:** applied deep ConvNets to the task of street number recognition, showing increased depth led to better performance.
2. **GoogleNet (Szegedy et al., 2014):** another deep ConvNet that was a top-performing entry of the ImageNet (ILSVRC-2014) challenge.

Architecture



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
	LRN	**conv3-64**	conv3-64	conv3-64	conv3-64
maxpool					
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
		conv3-128	conv3-128	conv3-128	conv3-128
maxpool					
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
			conv1-256	**conv3-256**	conv3-256
					conv3-256
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

1. **Preprocessing:** subtracting the mean RGB value computed on the training set from each pixel.
2. Convolution filters are 3X3, i.e. very small receptive field, with stride of 1 and ‘SAME’ padding.
3. Pooling layers are 2X2 with stride 2.
4. All hidden layers are followed by ReLU non-linearity.
5. The number of weights in VGG is fewer than shallower models with larger convolutional layer widths and receptive fields (fewer than 144M weights).

Implementation Details

1. Batch size was 256, momentum was 0.9, number of epochs equal to 74.
2. The fully connected layers had L2 weight norm with the regularization constant set to 0.0005, dropout with probability equal to 0.5.
3. The learning rate started from 0.01, and decayed by 10 for three times whenever validation accuracy stopped improving.
4. Data Augmentation:
    a. Random crop
    b. Random horizontal flip
    c. Random RGB shift
    d. Random scaling:
        i. **Single Scale Training:** the images are scaled to a certain size (256 or 384).
        ii. **Multi Scale Training:** the images are individually rescaled by randomly sampling from a certain range {S_{min} , S_{max}} (S_{min} = 256, S_{max} = 512). Since objects in images can be of different size, it is beneficial to take this into account during training. This can also be seen as training set augmentation by scale jittering.
5. Implemented in Caffe and multiple GPUs for data parallelism. They saw a speedup of 3.75 on an off-the-shelf 4-GPU system compared to using a single GPU. Training a single net took 2-3 weeks depending on the architecture.
6. During testing, both multi-crop evaluation and dense evaluation used. When applying ConvNet to a crop, the convolved feature maps are padded with zeros, while in the case of dense evaluation the padding for the same crop naturally comes from the neighbouring parts of an image (due to both the convolutions and spatial pooling), which substantially increases the overall network receptive field, so more content is captured.

Observations

1. Faster convergence due to two reasons:
    a. Implicit regularization imposed by greater depth and smaller convolution filter sizes.
    b. Pre-initialization of certain layers (first network A is trained, then its weights are used in training of other networks).
2. **Using 3X3 filters because:**
    a. The receptive field of two 3X3 convolutional layers are equivalent to one 5X5 convolutional layer.
    b. The receptive field of three 3X3 convolutional layers are equivalent to one 7X7 convolutional layer.
    c. More 3X3 layers with the same receptive field have the following two advantages:
        i. There is more than one non-linear layers, which makes the decision function more discriminative.
        ii. It internally provides regularization, by reducing the number of parameters to be trained.
3. Local Response Normalisation (LRN) does not improve on the model and leads to increased memory consumption as well as computation time.
4. Increase in depth (from network A to network E) improves the performance.
5. Use of 3X3 filters rather than 1X1 filters work better in VGG.
6. Two 3X3 filters give better performance than one 5X5 filter (nearly 7% difference).
7. Scale jittering helps in improving performance, rather than training with a fixed scale.
8. Taking the average of multiple crop evaluation output and dense evaluation output gives better performance.

Take Away Message

Increasing the depth of the network is important for learning better visual representations. Also, using smaller convolutional filters like 3X3 gives significant boost in performance.