Discussion Quiz #1

Ashutosh Tiwari (ashutiwa@iu.edu)

January 31, 2022

1 Papers

1.1 Paper 1

AlexNet: ImageNet Classification with Deep Convolutional Neural Networks

1.2 Paper 2

VGGNet: Very Deep Convolutional Networks for Large-Scale Visual Recognition

2 Questions

2.1 What are the assumptions of convolutions presented in paper 1. Describe each of these assumptions briefly.

Paper talks about following assumptions about CNNs. These assumptions are underlined by authors in order to justify their reasoning to pick these networks for this problem statement and to compensate for data they do not have to learn the required representations.

(a) Their complexity can be controlled by scale of the network This means that by upscaling and downscaling the network we can force network to increase/decrease number of trainable parameters, receptive fields, feature map size etc, which in turn controls the feature learning power of the network.

(b) They make strong and exact predictions about the stationarity of statistics and locality of pixel dependencies This can be derived just from the convolution properties. In other words this statement just says that feature maps keep statistics of local features intact.

2.2 Why CNNs are easier to train as compared with fully connected networks? In paper 1, why does the author say, "their (CNNs) theoretically-best performance is likely to be only slightly worse"? Explain.

CNNs are easier to train because as they have fewer parameters they need less memory and tend to overfit less. Similarly as they have less connections, forward and backward passes take much less time. For these very same reasons they need to update less weights and do less calculations as compared to fully connected networks.

Their theoretical best is little bit worse than dense layers, because dense layers have connections between each neurons of two consecutive layers and thus can theoretically learn all possible representations or features. CNNs are in general not that good in being able to generalize global patterns and their context is concentrated locally.

2.3 Why is ReLU a non-saturating non-linearity? What are its advantages over other activation functions like sigmoid, tanh, and softsign. How it help in training deep neural network models?

ReLU is non-saturating because in one direction its gradients don't saturate i.e. they are not asymptotic to zero. Non linearity is just another term for activation function. These functions (or non-linearities) are used to multiply output from one layer to be used as an input for another, to make that signal non-linear.

When compared to other non-linearities which saturate, it keeps providing sufficient gradient to be multiplied with learning rate so that at every iteration you get a significant value to update your already existing weights. A network will not be able to learn in case they don't get enough gradient to be multiplied with gradient of error along with learning rate.

2.4 Define briefly (in context of paper 1): (a) Local Response Normalization, (b) Overlapping pooling, (c) Data augmentation, (d) Dropout, (e) top-1 accuracy, and (f) top-5 accuracy

(a) Local Response Normalization

Unbounded nature of few activation functions (like ReLU) can make learning in a network unstable. The amount by which they change the output of a layer can be disproportional to importance of the feature map it receives. And for that reason it may try to suppress the activity of adjacent neurons.

In this paper, authors talks about a non-trainable form of local response normalization which takes into consideration the output from all activations at same spatial position.

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^\beta$$

Squares of activation from all those kernel maps is added and then is added to hyperparameter k raised to power β and then used to divide the activation value from concerned non-linearity.

Another form of LRN is batch normalization, which has trainable parameters as opposed to this method.

(b) Overlapping Pooling

Pooling layers select the most prominent/important feature in their receptive field at their center. In different cases, it can either can be maximum or average of that of receptive field. Authors point out that in most of cases people use pooling such that adjacent pooling units don't overlap.

In this work authors used pooling such that the input receptive field between every adjacent overlapped. Paper points out that it helped them tackle over-fitting in their network.

(c) Data augmentation

In general data augmentation is the practice of modifying data before it is fed to network to learn feature maps. Authors here used two different sets of augmentations.

In first case they used random patches of their images and then their horizontal reflections. In second, they altered the intensities of RGB channels in images by adding multiples of principal components found on the entire dataset with magnitudes proportional to eigenvalues multiplied with a random variable drawn from a Normal distribution with mean zero and standard deviation of 0.1.

(d) Dropout

Dropout is a very well known technique to tackle overfitting. In this paper authors use a dropout value of 0.5 which means that in every forward and backward pass, each neuron has a probability of 0.5 to be involved in forward and backward pass in that pair of forward and backward pass.

However, authors do point out that using dropout increases the iterations required to converge for the network. They used dropouts for their first two fully connected layers.

(e) Top-1 accuracy

Top-1 accuracy is number of times our top prediction was same as expected answer.

$$Top-1\ accuracy = \frac{\#\ of\ times\ our\ top\ prediction\ was\ same\ as\ expected}{total\ \#\ of\ predictions}$$

(f) Top-5 accuracy

Top-5 accuracy is number of times our top five predictions included expected answer as one of those values.

$$Top-5\ accuracy = \frac{\#\ of\ times\ our\ top\ five\ predictions\ included\ expected\ value}{total\ \#\ of\ predictions}$$

2.5 Explain the sentence, "initializing biases with 1 instead of 0 accelerates the early stages of learning in deep model". Also, how do we choose, when to initialize biases with 1 or 0?

Paper points out that they initialized biases in second, fourth and fifth convolutional layers and all fully connected layers to be 1 and for all other layers to be 0. They claim that this will facilitate learning by giving positive inputs to their activation function. This makes sense since output for values ≤ 0 is constant in case of ReLU, and in that case further layers won't get any substantial input to learn.

In order to decide what value to initialize for bias, we should keep two things in mind. First is the size of neural network. Because in first few iterations, you are basically propagating your biases. And if there are a lot of layers, upstream layers need to get enough inputs in starting phase of training. Second thing to keep in mind is the data distribution. If it is too unbalanced, setting bias of logits such that network predicts probability that of data distribution helps in initial learning.

2.6 How can you design an effective receptive field of 5x5 and 7x7 using only 3x3 convolutional layers? Why there should not be spatial pooling in between these convolutional layers?

Stack of two 3×3 convolutional layers have same receptive area as one 5×5 layer and stack of three 3×3 convolutional layers have same receptive area as one 7×7 layer. This can also be shown arithmetically.

Spatial pooling cannot be included in between these layers because then we won't be doing convolutional operation over whole receptive field but rather on selected view of it, depending on the kind of spatial pooling we choose to use.

2.7 Explain, "The incorporation of 1x1 convolutional layers is a way to increase the non-linearity of the decision function without affecting the receptive fields of the convolutional layers?"

1×1 convolutional layers can be used for many purposes, including dimensionality reduction, reduce parameter map and adding non-linearity to network.

Here authors talk about a configuration where they used same number of output channels in 1×1 convolution layer as the input layer followed by ReLU which operates on linear projection by the colvolution layer and thus produce non-linearity.

2.8 In paper 1 and paper 2, while training, the learning rate is reduced by some constant factor once validation error stops improving. Why?

In general this nature of neural networks to stop improving for validation set is called "plateauing". This happens because of two reasons. Either the network has encountered a saddle point or it has reached a point of local minima.

There are multiple ways to come out of these situations. One of them is to do learning rate decay, which is used in both of these papers. What it means is that you use a learning rate scheduler and learning rate is decreased by an amount which decreases with time in future. This just means that velocity with which you glide on decision surface decreases with times.

Another technique which is widely used is cycling learning rate, which is particularly helpful in local minima problem.

2.9 According to paper 2, how does using a small filter size (3x3) all across the network helped them in training?

Authors talk about multiple benefits of using small convolutional layers. First one is the reduced number of trainable parameters allow them to use a very deep network. Secondly by incorporating multiple 3×3 rectification layers instead of a single big one, they make decision function more discriminate i.e. separation between classes will be more wide. Because they can effectively imitate receptive field of any size, using 3×3 is not a downside in any case.

3 References

1. Andrej Karpathy's Blog