Rahul Thapa

Summary:

In this ImageNet Classification paper, the author uses a large and deep convolutional neural network to classify 1.2 million high-resolution images in the ImageNet LSVRC-2010 content. They classified 1000 different classes with their neural network. The paper claims that the capacity of a convolutional neural network can be controlled by varying their depth and breadth and that they make strong and mostly correct assumptions about the nature of the image. The authors assert that CNN has fewer connections and parameters and hence, they are easier to train compared to standard feedforward neural networks.

The authors made few modifications to the image from the ImageNet dataset. First, they down sampled the images to a fixed resolution. They also subtracted the mean activity over the training set from each pixel. The architecture itself consisted of eight layers, five of which were convolutional layers and three were fully connected layers. For the non-linearity, they used the ReLU activation function because they found out that neural networks with ReLU train several times faster than their equivalent tanh units. The faster learning rate has a great influence on the performance of large models trained on large datasets. One hardware modification they made in their experiment was that they trained the network on two different GPUs because the 1.2 million training examples are too big to fit in one GPU.

Even though ReLUs do not require input normalization to prevent them from saturating, the authors still did local response normalization which reduced their error significantly. They also had a pooling layer in their network which summarize the outputs of the neighboring groups of neurons in the same kernel map. Instead of using traditional local pooling, which is commonly employed in CNNs, the authors used overlapping pooling. They found out that the models had less chance of overfitting when they used overlapping pooling. Response normalization layers follow the first and second convolutional layers and max-pooling follow both response-normalization layers and fifth convolutional layer. The ReLU nonlinearity is applied to the output of all the layers in the network.

The authors mention two primary ways they tackled overfitting. First, they talk about data augmentation which is basically artificially enlarging the dataset using label-preservation transformations. They used two distinct forms of data augmentation. The first form of data augmentation consists of generating image translations and horizontal reflections. The second form of data augmentation consists of altering the intensity of the RGB image channels in the images. Finally, they also mention another methodology called dropout in which they basically set the output of each hidden neurons to zero with the probability of 0.5. These dropped out neurons do not contribute to the forward pass and back-propagation. This technique reduces the complexity of the model and helps the model to generalize much better.

Strengths:

1. The figure showing the convolutional neural network is clear and descriptive.

2. The organization of the paper is good. There are multiple sections each explaining a crucial part of the network.
3. They do a good job to explain each part of their network. They also included a lot of their system specification and parameters they used which makes it easy to envision how they did the experiment.
4. They explain the rationale behind why they used the technique they used very clearly. They even include the result of their competitor to give an idea of how good the performance of their model is as compared to others.

Weaknesses:

1. Few mathematical equations they include in the paper are confusing. They have not done a good job of explaining those equations.
2. I felt like the introduction section was too long and repetitive.

Confusions:

1. How does splitting the training into two different GPUs work? How is it different than training in a single powerful GPU? Do we see any difference in the performance of the model?
2. What is local response normalization? How do we understand its math intuitively?

Discussions:

1. What is ReLU and why does it perform better than other activation functions for this specific purpose?
2. What is the difference between local pooling and overlapping pooling?
3. How is multiplying the output of the neurons in the test data set by 0.5 analogous to dropping out the neurons in the training set?