ImageNet Classification with Deep Convolutional Neural Networks

Sidharth S
S7 ECE Gamma

Guided by,
Abhishek Viswakumar,
Assistant Professor
Dept. of Electronics and Communication
RSET

January 3, 2021

Rajagiri School of Engineering and Technology

Introduction

- AlexNet
- One of the most influential papers published in computer vision.
  - AlexNet paper has been cited over 70,000 times according to Google Scholar.
- The network achieved a top-5 error of 15.3% in ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) .
- It is the first Convolutional Neural Network (CNN) where multiple convolution operations were used.

Abstract

- Deep convolutional neural network to classify the 1.2 million high-resolution image.
- 60 million parameters and 650,000 neurons.
- 1000 class image classification.
- Convolutional Layers, Maxpooling layers, Fully connected layers and Softmax layer.
- ReLU activation function is used.
- GPU implementation of the convolution operation.
- Dropout regularization.

A Short Introduction to Convolutional Neural Networks

Fully Connected Neural Networks

Fully connected neural networks (FCNNs) are a type of neural network where the architecture is such that all the nodes, or neurones, in one layer are connected to the neurones in the next layer.

For a $64x64x3$ image,
No of parameters in input layer = $12,288$

For a $225x225x3$ image,
No of parameters in input layer = $151,875$

- Networks having large number of parameter face several problems, for e.g. slower training time, chances of overfitting e.t.c.

Convolution

In mathematics, convolution is a mathematical operation on two functions f and g that produces a third function $f * g$ that expresses how the shape of one is modified by the other.

- The main image matrix is reduced to a matrix of lower dimension in the first layer itself
- The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction.

How Convolution is performed

We can use an input image and a filter to produce an output image by convolving the filter with the input image.

Steps:

1. Overlaying the filter on top of the image at some location.
2. Performing **element-wise multiplication** between the values in the filter and their corresponding values in the image.
3. Summing up all the element-wise products. This sum is the output value for the destination pixel in the output image.
4. Repeating for all locations.

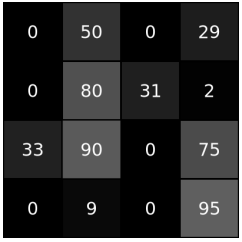

Figure 1: A 4x4 image (left) and a 3x3 Sobel filter (right)

Each filter actually happens to be a collection of kernels, with there being one kernel for every single input channel to the layer, and each kernel being unique.

Convolution Example

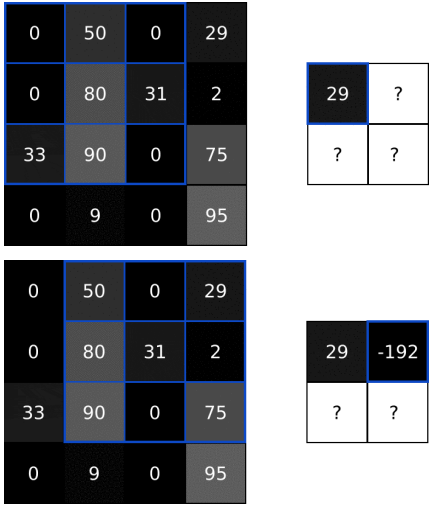

Figure 2: Example of Convolution

Convolution Example

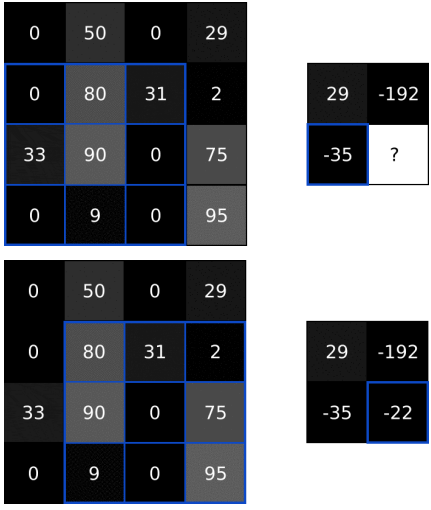

Figure 3: Example of Convolution

Kernals/Filters

- *Sobel filters* are edge-detectors.
- Kernals can detect features on a global scale, anywhere in the image.
- Kernals to find certain features can be learned using machine learning.
- Convolution helps us look for specific localized image features (like edges) that we can use later in the network.
- **Learn new filters** to classify certain features

Figure 4: An image convolved with the vertical Sobel filter

MaxPool Layers

- Neighboring pixels in images tend to have similar values, redundant information.
- Reduce the spatial dimension of the input volume for next layers.

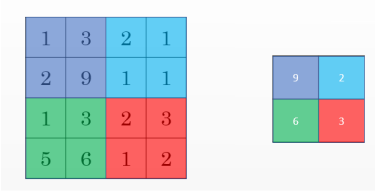

Figure 5: Maxpooling Example

- The max pooling is saying, if the feature is detected anywhere in this filter then keep a high number.
- No parameters to learn.

Padding

- If a matrix *nxn* is convolved with *fxf* filter/kernel give us $n - f + 1 x n - f + 1$ matrix.
- Shrinks output.
- Throws away a lot of information that are in the edges.

To solve these problems we can pad the input image before convolution by adding some rows and columns to it. *p* rows and columns are padded to the input image.

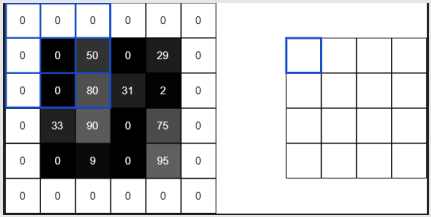

Figure 6: Padding the input

Stride

Stride s tell us the number of pixels we will jump when we are convolving filter/kernel.

For a layer l of a ConvNet,
$f[l]$ = filter size
$p[l]$ = padding
$s[l]$ = stride
$n_c[l]$ = number of filters, Then,

Size of output layer

When a layer is convolved with filter of size $f[l]$ x $f[l]$ x $n_c[l-1]$
Output size is $n[l]$ x $n[l]$ x $n_c[l]$ where,

$$n[l] = \frac{n[l-1] + 2p[l] - f[l]}{s[l]} + 1$$

$n_c[l]$ = Number of filters used.

AlexNet

Architecture of AlexNet

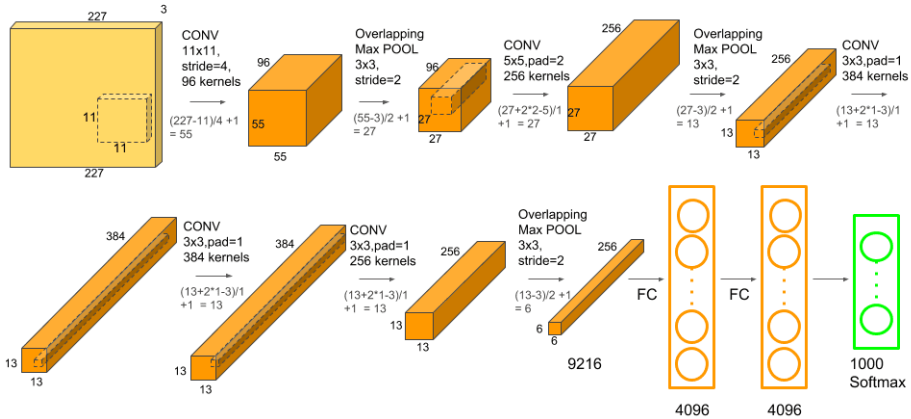

Figure 7: Architecture of AlexNet

Arhitecture of AlexNet

- 8 learned layers
  - Five convolutional and three fully-connected.
- Final layer is softmax layer which can classify 1000 classes
- $\sim$ 60 million parameters.
- Trained on ImageNet dataset.
- Uses ReLU nonlinearity for activation.

ReLU Nonlinearity

What is an activation function?

- Decides whether a neuron should fire or not.
- Helps the network learn complex patterns in data.
- Takes output of previous layer and converts it to meaningful form to transfer to next layer as input.

Why do we need activation functions?

- Output value of a layer is restricted to a limit.
- Ability to add **nonlinearity** into a neural network.
- Allows back propagation and gradient descent to update the weights in the network.

Some examples of activation functions are: *Sigmoid, tanh, ReLU, Linear*

ReLU Activation

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

- Deep convolutional neural networks with ReLUs train several times faster than their equivalents with tanh units.
- Easier to train.
- Better for learning complex relationships.

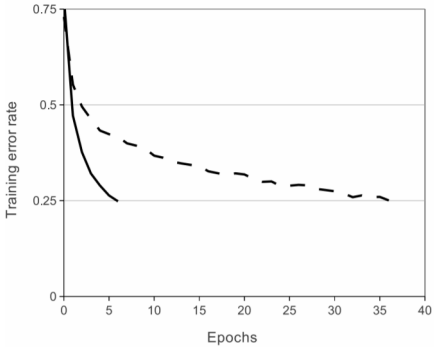

Figure 8: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**.

ImageNet Dataset

- 15 million high-resolution images labeled with 22 thousand classes.
- ImageNet Large-Scale Visual Recognition Challenge (ILSVRC).
- 1.2 million training images, 50 thousand validation images, and 150 thousand testing images.
- Authors used downsampled 227 x 227 images.

Training on GPUs

- GPUs are faster and efficient for matrix multiplication and convolution.
- CPUs are latency optimized. GPUs are bandwidth optimized.
- AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU.

Softmax Function

$$softmax(\overrightarrow{z})_i = \frac{exp(z_i)}{\sum_j^K exp(z_j))}$$

$\overrightarrow{z}$ The input vector to the softmax function, made up of (z0, ... zK)
z_i elements of the input vector to the softmax function

- Used for multiclass classification/regression.
- The Softmax regression is a form of logistic regression that normalizes an input value into a vector of values that follows a probability distribution whose total sums up to 1.

Reduce Overfitting

- Overfitting refers to a model that models training data too well.
- Happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data.

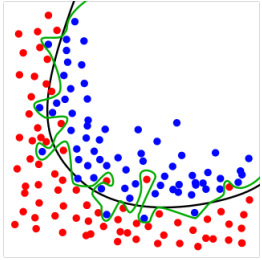

Figure 9: Black line fits the data well. Green is overfitting.

Overfitting can be reduced by: Getting more training data, Regularization.

Dropout Regularization

- Sets to zero the output of each hidden neuron with probability 0.5.
- The neurons which are “dropped out” do not contribute to the forward pass and do not participate in backpropagation.
- This technique reduces complex co-adaptations of neurons.
- Network is forced to learn more robust features.
- Employs dropout in the first two fully-connected layers of the network.
- *”Without dropout, our network exhibits substantial overfitting.”*
- Dropout increases number of iterations required to train the network.

Details of Learning

- Trained using stochastic gradient descent with batch size = 128.
- Momentum 0.9 and weight decay of 0.0005.
- Small amount of weight decay was important for the model to learn.

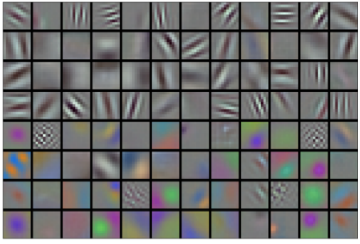

Figure 10: 96 convolutional kernels of size 11×11×3 learned by the first convolutional layer on the 224×224×3 input images.

- The network achieves top-1 and top-5 test set error rates of 37.5% and 17.0% in ILSVRC - 2010.
  - Next best result is 45.7% and 25.7% respectively.

Model	Top-1	Top-5
Sparse Coding	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Table 1: Comparison of results on ILSVRC-2010 test set. In italics are best results achieved by others.

- The CNN described in this paper achieves a top-5 error rate of 18.2% in ILSVRC - 2012.
  - The second-best contest entry achieved an error rate of 26.2%.

Figure 11: Eight ILSVRC-2010 test images and the five labels considered most probable by the model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5).

Conclusion

- AlexNet was the pioneer in CNN and open the whole new research era.
- Dropout, ReLU, and deep layers are key steps in achieving excellent performance in computer vision tasks.
- Removing any of the convolutional layers will drastically degrade AlexNet's performance.

Comparison:

- *GoogleNet*: Winner of the ILSVRC 2014 competition was GoogleNet from Google. Achieved top-5 error rate of 6.67%.
- *ResNet*: Winner, ILSVRC 2015. Introduced a novel architecture with "skip connections". Achieves a top-5 error rate of 3.57% which beats human-level performance on this dataset.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1303–1314, 2018.

THANK YOU