GoogLeNet Brief Notes

Going deeper with convolutions : https://arxiv.org/abs/1409.4842

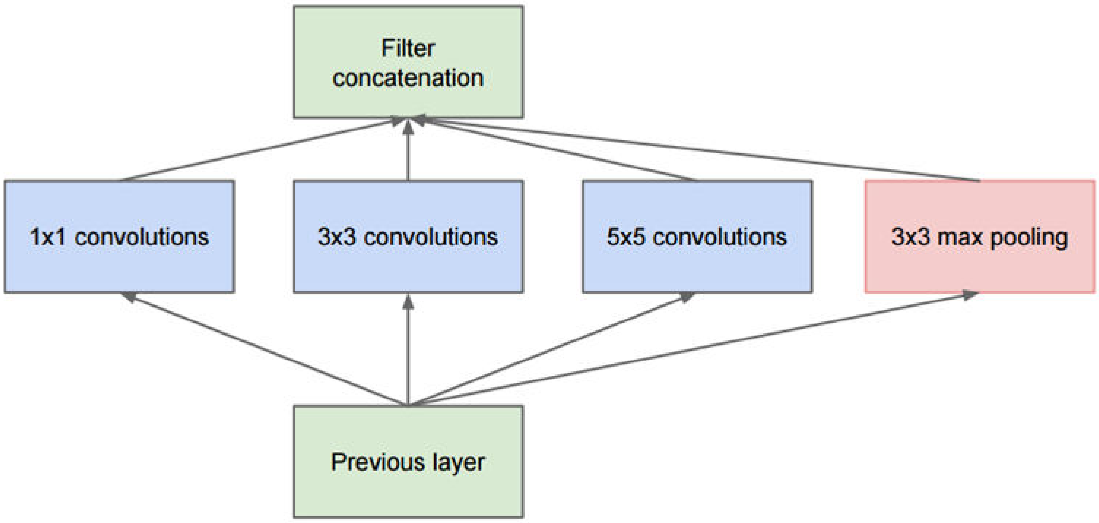
Main Objective Of GoogLeNet :

To improve speed and performance and memory usage compared to previous convnets

Architecture

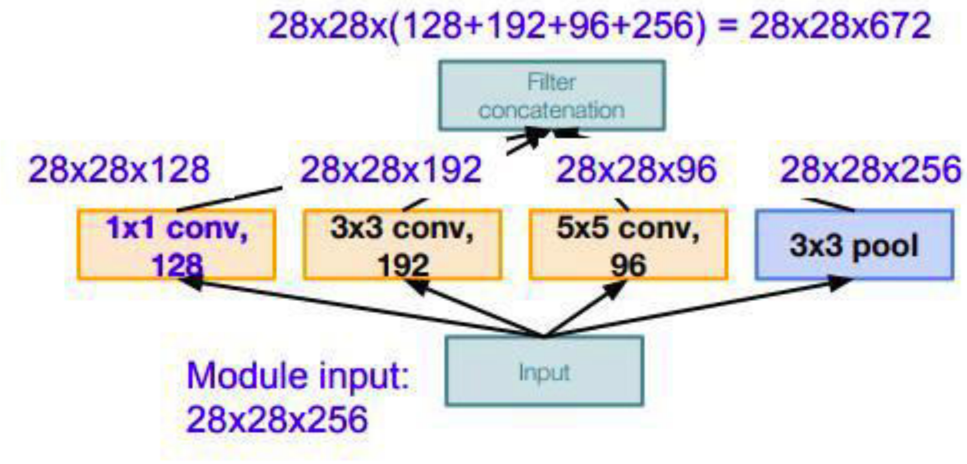
- 22 layers
- Fundamentally different from conventional convnets like alexnet.
- **No Fully Connected layers**.As result it uses just 5M parameters. 12 times less than AlexNet(60M parameters.)
- Contains inception modules instead of convolution and pooling and FC layers.
- In an inception module we apply parallel convolution and pooling operations and then concatenate.

A **Naive** Inception Module



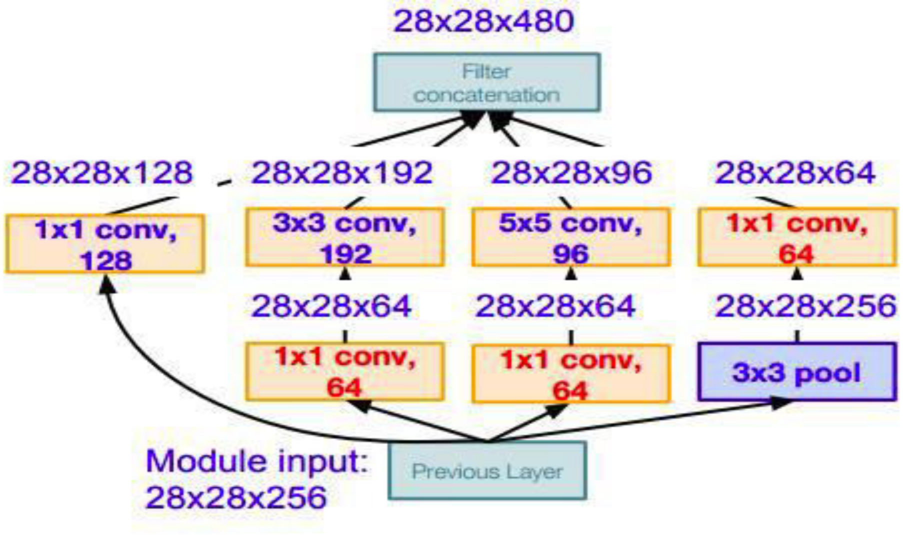
Naïve idea of an Inception module

The problem that we face in the above module is computational complexity. We can see that depth in the final output of the module is almost **tripled**.And this increase is shown just for one inception module.We **stack 9 inception module** in the **final convnet**.So we need to somehow reduce the computational complexity in the module .
NOTE : The spatial dimensions of intermediate outputs are same after various convolutional operations due to **padding**.



Naive Inception module

To decrease the computational complexity we introduce 1*1 convolutional layers in each parallel operations.Final inception module becomes the following.



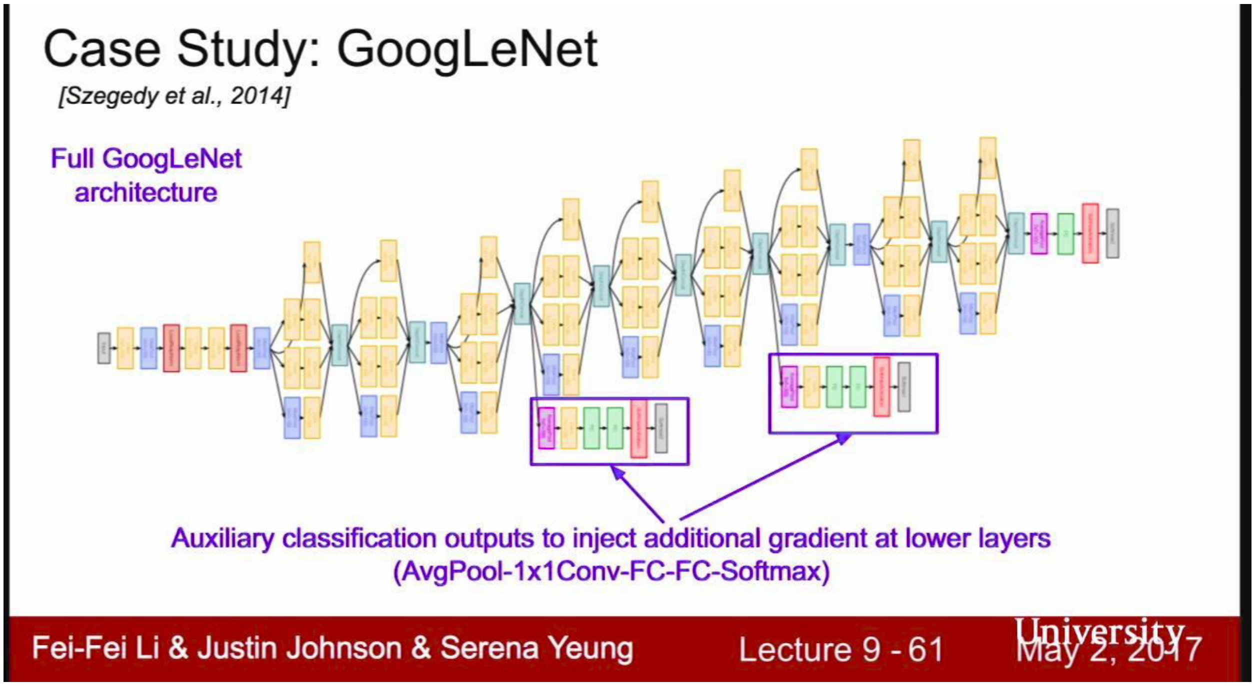
Inception module with dimension reduction

Advantages of doing this :

- By applying 1*1 convolution we actually **reduce** the computational complexity much better than naive inception module.
- Generally for every convolution operation we apply a **ReLu** operation immediately after it As a result we are introducing more non-linearity into the network
- Also you might think that by apply 1*1 convolutional operations we actually lost some features.No we didn't.We stored the feature info in the form of **linear combination** of actual features.We just reduced the depth to improve the computational complexity.
- We also reduce the number of **operations** during convolution from **854M to 358M**

In the final convnet we stack **9 such inception modules**

Along with final output there are **auxiliary outputs** in GoogLeNet.



Case Study: GoogLeNet

[Szegedy et al., 2014]

Full GoogLeNet architecture

Auxiliary classification outputs to inject additional gradient at lower layers
(AvgPool-1x1Conv-FC-FC-Softmax)

Fei-Fei Li & Justin Johnson & Serena Yeung   Lecture 9 - 61   University May 2, 2017

The reason why we add these auxiliary outputs to increase gradient training i.e
For every backward propagation the initial modules get more trained because of 3 outputs (2 auxiliary and 1 final output) and middle modules are trained by one aux and final output
And final modules are trained only by final output.
So we can visualise that more training is done in the network due to the introduction of auxiliary outputs compared to just one final output.
Suppose we used only final output (No aux outputs).During back propagation the magnitude of gradient gets reduced significantly as we go from right to the left.So the initial modules can't learn better than the final modules.So to compensate this and make sure that initial and middle modules also learn faster we introduce these auxiliary outputs