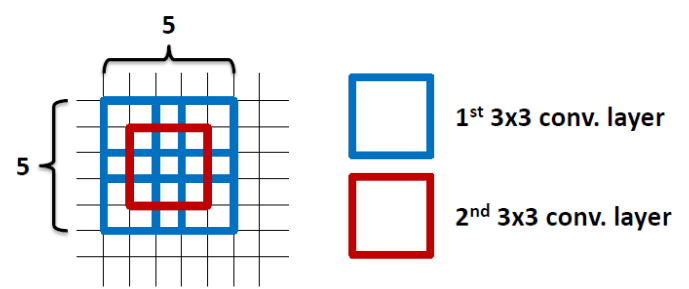
**VGGNet (2014)**

1. **The Use of 3×3 Filters** instead of large-size filters (such as 11×11, 7×7)
2. **VGG-16** and **VGG-19** based on ablation study   
   (VGG-11, VGG-11 (LRN), VGG-13, VGG-16 (Conv1) are also included.)
3. **Multi-Scale Training**
4. **Multi-Scale Testing**
5. **Dense Testing**
6. **Model Fusion**
7. **Comparison Between VGGNet and GoogLeNet**
8. **Localization Task**

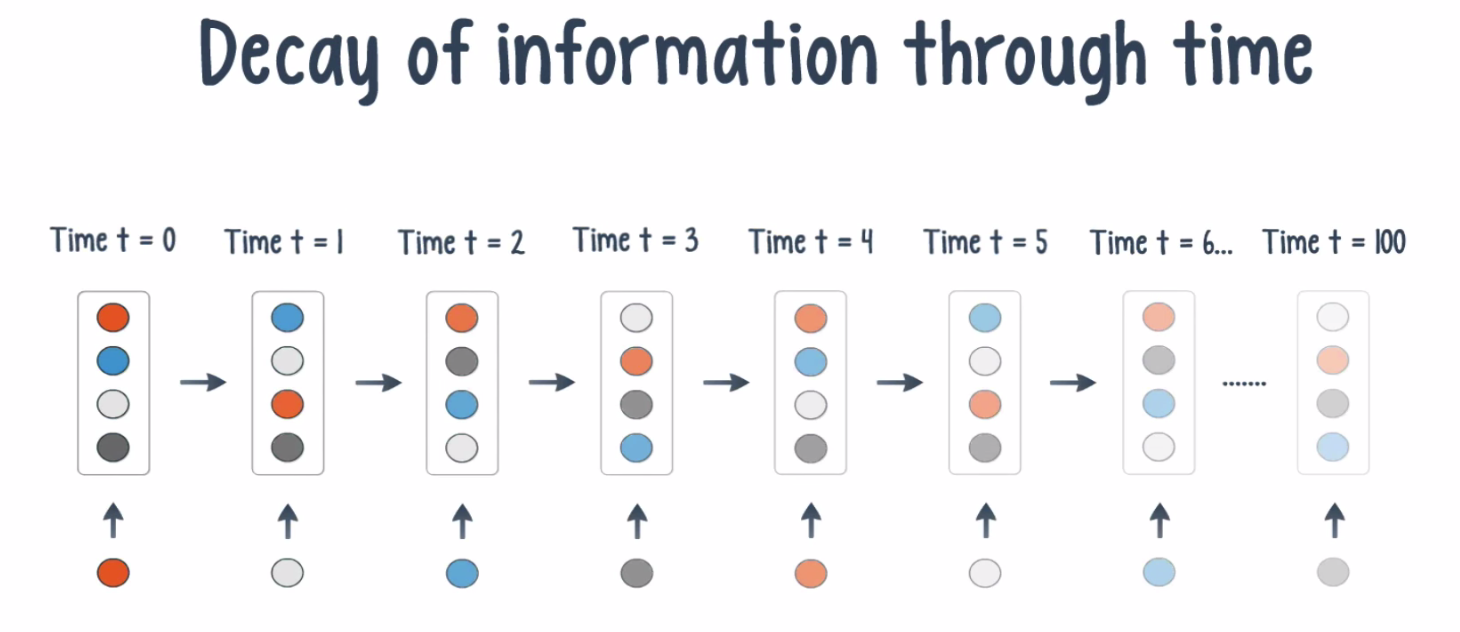
**The Use of 3×3 Filters**



Another reason is that **the number of parameters are fewer.** Suppose there is only 1 filter per layer, 1 layer at input, and exclude the bias:

* 1 layer of 11×11 filter, number of parameters = 11×11=121
* 5 layer of 3×3 filter, number of parameters = 3×3×5=45
  + Number of parameters is reduced by 63%
* 1 layer of 7×7 filter, number of parameters = 7×7=49
* 3 layers of 3×3 filters, number of parameters = 3×3×3=27
  + Number of parameters is reduced by 45%
* By using 1 layer of 5×5 filter, number of parameters = 5×5=25
* By using 2 layers of 3×3 filters, number of parameters = 3×3+3×3=18
  + Number of parameters is reduced by 28%
* Larger network, hungrier the network for the training images. There are also vanishing gradient problem.
* With fewer parameters to be learnt, it is better for faster convergence, and reduced overfitting problem.

**Vanishing Gradient Problem.**



Vanishing Gradient Problem occurs when we try to train a Neural Network model using Gradient based optimization techniques.

Vanishing Gradient Problem was actually a major problem 10 years back to train a Deep neural Network Model due to the long training process and the degraded accuracy of the Model.

What happens is that as we keep on adding more and more Hidden layers in The model , the learning speed of the next hidden layers in the model keep on getting faster and faster.

Generally, adding more hidden layers tends to make the network able to learn more complex arbitrary functions, and thus do a better job in predicting future outcomes.

This is where **Deep Learning** is making a big difference due to the *thousands and millions of* ***hidden layers*** it has , we can now make sense of highly complicated data such as images , speeches , videos etc and do Speech Recognition and Image Classification , Image Captioning etc.

What is the Vanishing Gradient Problem?

Now when we do Back-propagation i.e moving backward in the Network and calculating gradients of loss(Error) with respect to the weights , the gradients tends to get smaller and smaller as we keep on moving backward in the Network.

This means that the neurons in the Earlier layers learn very slowly as compared to the neurons in the later layers in the Hierarchy.

The Earlier layers in the network are slowest to train.

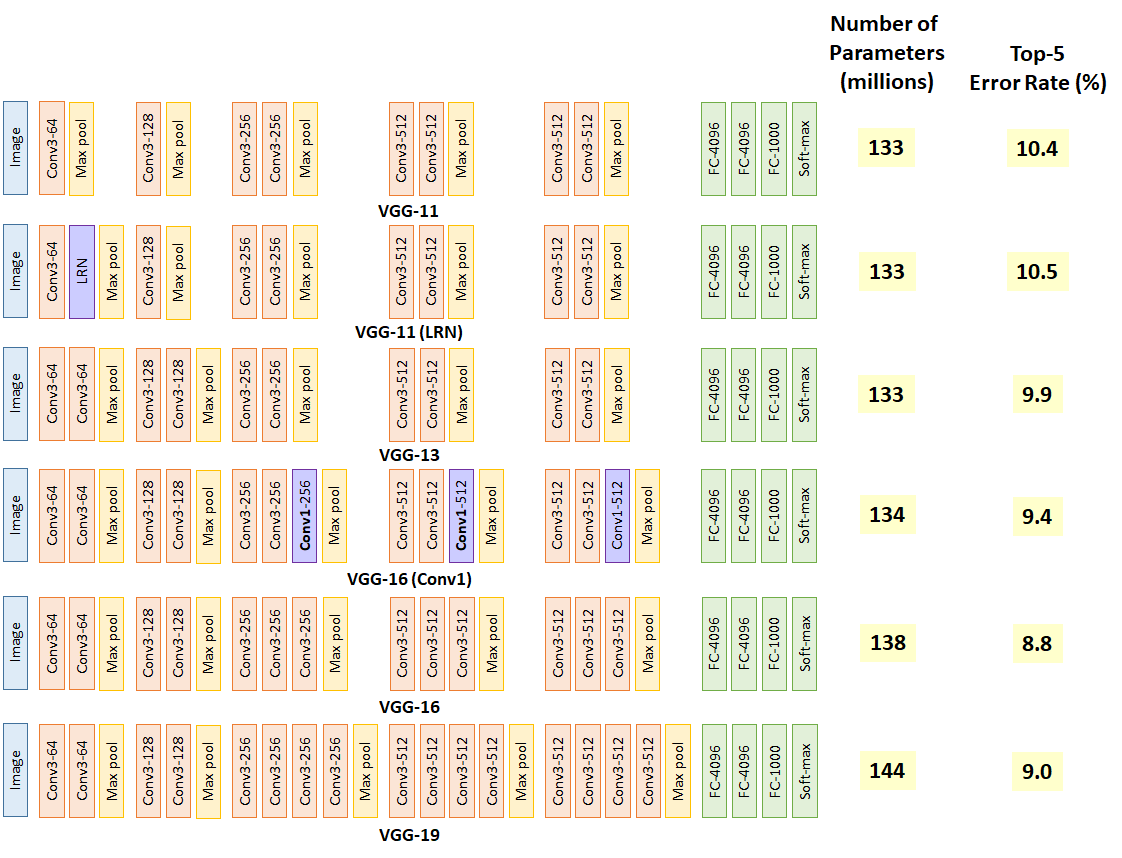
Why Earlier layers of the Network are so important to us?

Earlier layers in the Network are important because they are responsible to learn and detecting the simple patterns and are actually the building blocks of our Network.

IF Training process takes too long and the Prediction Accuracy of the Model will decrease.

Hence This is all Vanishing Gradient problem does to our Neural Network Model. Just think of a Deep Neural Network Model which is highly complicated and has millions of layers in it, how problematic it can be to train such a deep Network and produce good and accurate results.

This is the reason why we do not use Sigmoid and Tanh as Activation functions which causes vanishing Gradient Problems . Hence mostly nowadays we use RELU based activation functions in training a Deep Neural Network Model to avoid such complications and improve the accuracy .



To obtain the optimum deep learning layer structure, ablation study has been done as shown in the above figure.

* VGG-11 already obtains 10.4% error rate, which is similar to that of ZFNet in ILSVRC 2013.
* VGG-11 is set as benchmark.
* VGG-11 (LRN) obtains 10.5% error rate, is the one with additional local response normalization (LRN) operation suggested by AlexNet.
* By comparing VGG-11 and VGG-11 (LRN), the error rate doesn’t improve which means LRN is not useful.
* In fact, LRN is not used any more in later on deep learning network, instead, batch normalization (BN) is used.
* VGG-13 obtains 9.9% error rate, which means the additional conv helps the classification accuracy.
* VGG-16 (Conv1) obtains 9.4% error rate, which means the additional three 1×1 conv layers help the classification accuracy. 1×1 conv actually helps to increase non-linearlity of the decision function. Without changing the dimensions of input and output, 1×1 conv is doing the projection mapping in the same high dimensionality.
* VGG-16 obtains 8.8% error rate which means the deep learning network is still improving by adding number of layers.
* VGG-19 obtains 9.0% error rate which means the deep learning network is NOT improving by adding number of layers. Thus, authors stop adding layers.
* By observing the addition of layers one by one, we can observe that VGG-16 and VGG-19 start converging and the accuracy improvement is slowing down. When people are talking about VGGNet, they usually mention VGG-16 and VGG-19.

**Multi-Scale Training**

If we only train the network at the same scale, we might miss the detection or have the wrong classification for the objects with other scales.

For multi-scale training, an image is scaled with smaller-size equal to a range from 256 to 512, i.e. S=[256;512], then cropped to 224×224. Therefore, with a range of S, we are inputting different scaled objects into the network for training.

**VGG-13** reduced the error rate from 9.4%/9.3% to **8.8%.**

**VGG-16** reduced the error rate from 8.8%/8.7% to **8.1%.**

**VGG-19** reduced the error rate from 9.0%/8.7% to **8.0%.**

**Multi-Scale Testing**

Similar to multi-scale training, multi-scale testing can also reduce the error rate since we do not know the size of object in the test image. If we scale the test image to different sizes, we can increase the chance of correct classification.

VGG-13 reduced the error rate from 9.4%/9.3% to 9.2%.

VGG-16 reduced the error rate from 8.8%/8.7% to 8.6%.

VGG-19 reduced the error rate from 9.0%/8.7% to 8.7/8.6%.

By using both multi-scale training and testing, error rate is reduced.

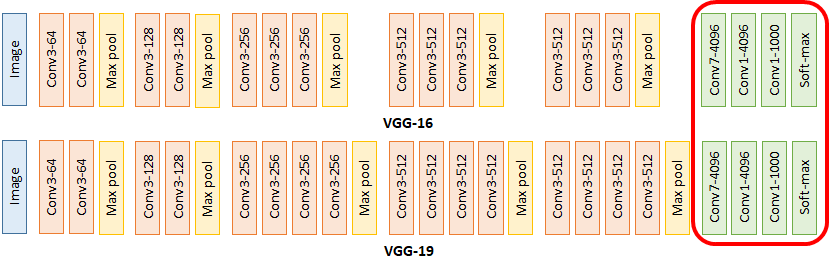
Compared with only multi-scale testing,

VGG-13 reduced the error rate from 9.2%/9.2% to 8.2%,

VGG-16 reduced the error rate from 8.6%/8.6% to 7.5%,

VGG-19 reduced the error rate from 8.7%/8.6% to 7.5%

**Dense (Convolutionalized) Testing**



**VGGNet During Testing**

During testing, in **VGGNet**, the test image is directly go through the VGGNet and obtain a class score map. This class score map is spatially averaged to be a fixed-size vector.

