Transfer Learning Architectures

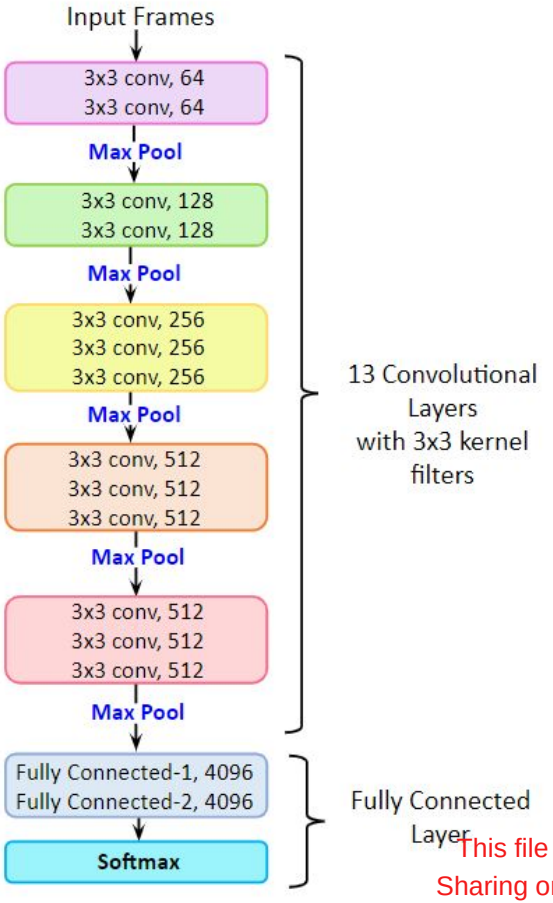
Week 2: Computer Vision

Agenda

- VGGNet
- The Inception Network
- Residual Networks (ResNets)

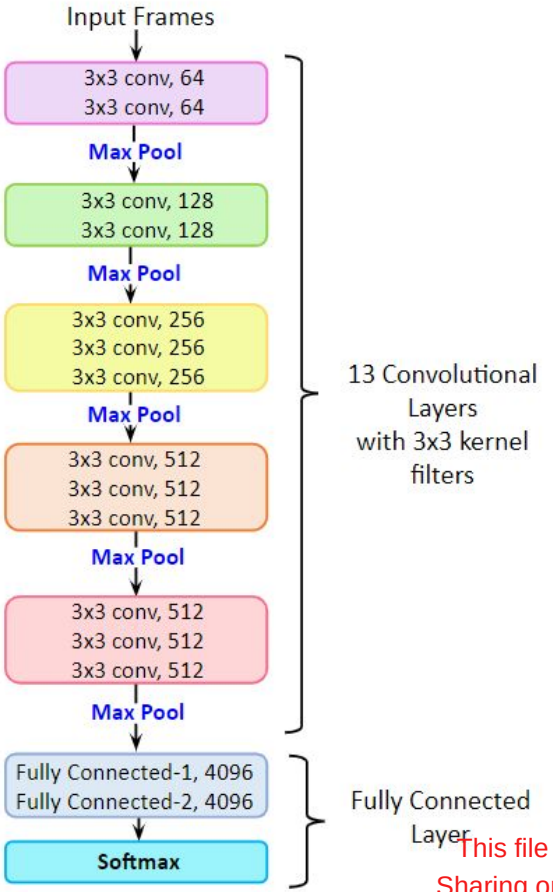
VGGNet

VGGNet



- The VGGNet architecture was proposed by Karen Simonyan and Andrew Zisserman, from the Visual Geometry Group (VGG) at the University of Oxford, in 2014. It even finished first runner-up in the ImageNet annual competition (ILSVRC) in 2014.
- VGGNet has two variants: **VGG16** and **VGG19**. Here, 16 and 19 refer to the total number of convolution and fully connected layers present in each variant of the architecture.
- In comparison to previous Deep Learning models for Computer Vision, VGGNet stood out for its **simplicity** and the **standard, repeatable nature** of its blocks. Its main innovation over standard CNNs was simply its **increased depth** (number of layers) - otherwise it utilized the same building blocks - convolution and pooling layers, for feature extraction.

VGGNet



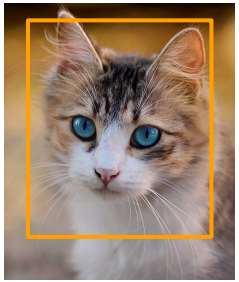
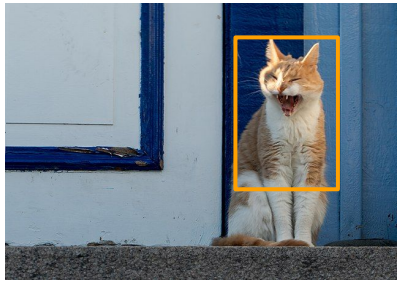
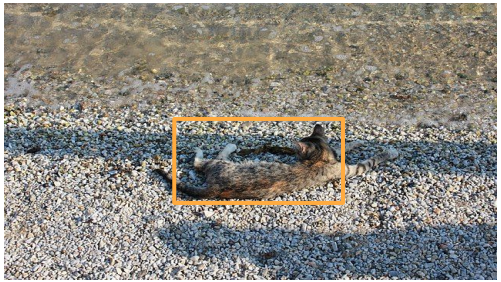
- VGGNet was the first architecture to standardize **smaller filters**, (a small 3x3 convolutional kernel is used in every layer) but **deeper networks**. The architecture has around 138 million trainable parameters.

- **Why use smaller filters?** A stack of three 3x3 convolutional (stride 1) layers has the same effective region as one 7x7 convolutional layer.
  Hence, the use of smaller convolution layers depicts more information about the image across each channel, while ensuring that the size of the output is consistent with the size of the input image (with the help of padding). In addition, multiple 3x3 convolutions will have a fewer number of trainable parameters in comparison to a 7x7 convolution, making this idea more computationally efficient as well

The Inception Network

The Inception Network

- While **VGGNet chooses a standard 3x3 size** for the convolutional kernels to simplify the architecture, an important question is, **is that necessarily the best kernel size** for every dataset?
- Choosing the perfect kernel size for the convolution operation can be a difficult endeavor, due to the huge variance in the location of critical information from image to image.

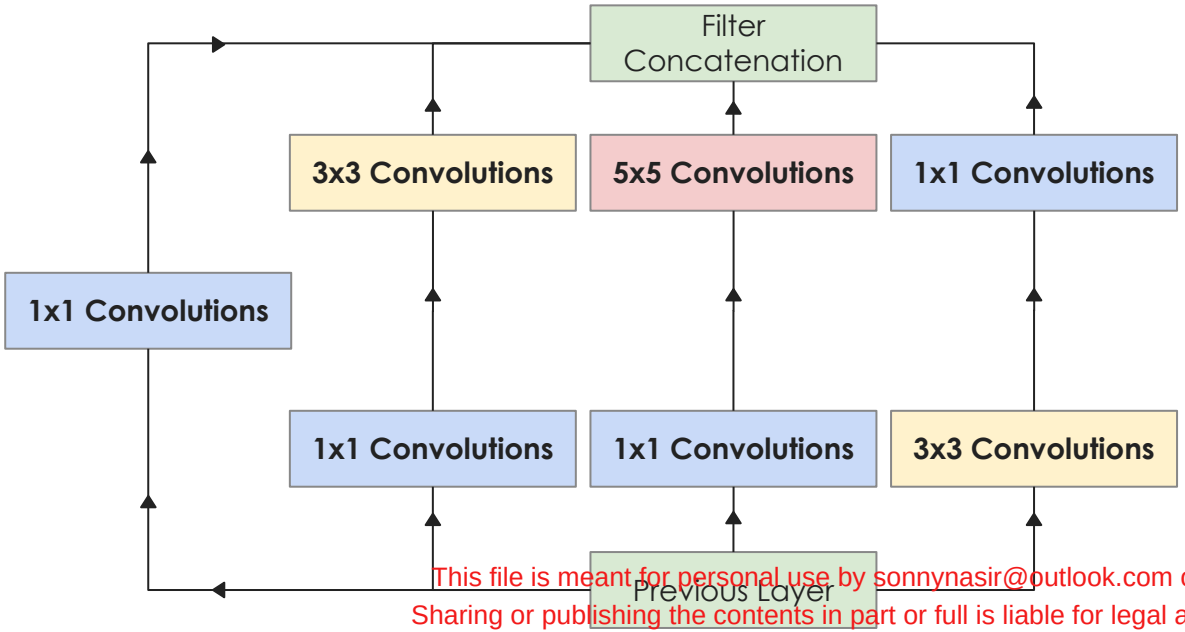
- Choosing the right filter size to extract the information needed from an image is not straightforward, as we see from the three images of cats.
- So in that case, why not have **filters with multiple sizes** operate on the **same input**, and then **aggregate their results** somehow, like an ensemble model?

This is the core idea behind the Inception Network.

The Inception Network - Inception Module

- The Inception Network (the winner of ILSVRC 2014), consists of certain specialized repeating blocks, each of which concatenates the outputs from multiple convolutional pathways.
- These blocks are also known as **Inception Modules**, and they represent a key point of difference between the Inception Network and other architectures.

The Inception Module

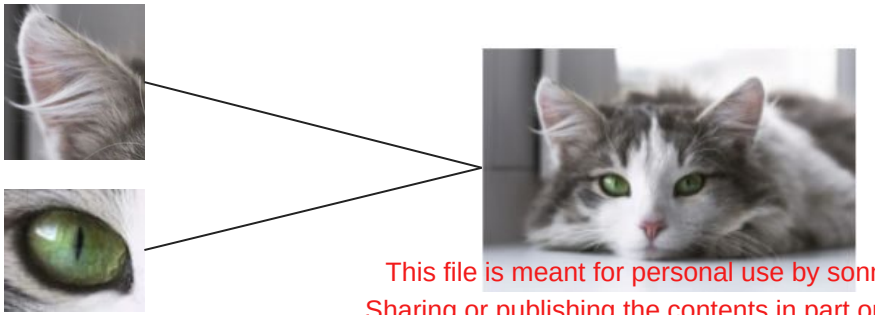


The Inception Network - Inception Module

- **Hence rather than choosing between** 1x1, 3x3 or 5x5 convolutional filter sizes, **the Inception Module concatenates the feature maps** that are produced from these filters.

- The importance of the Inception module can be recalled by the **Hebbian Principle of Human Learning,** which states that **"Neurons that fire together wire together."** This means that when creating a new layer in a Deep Learning model, one should always pay attention to the learnings of the previous layer.

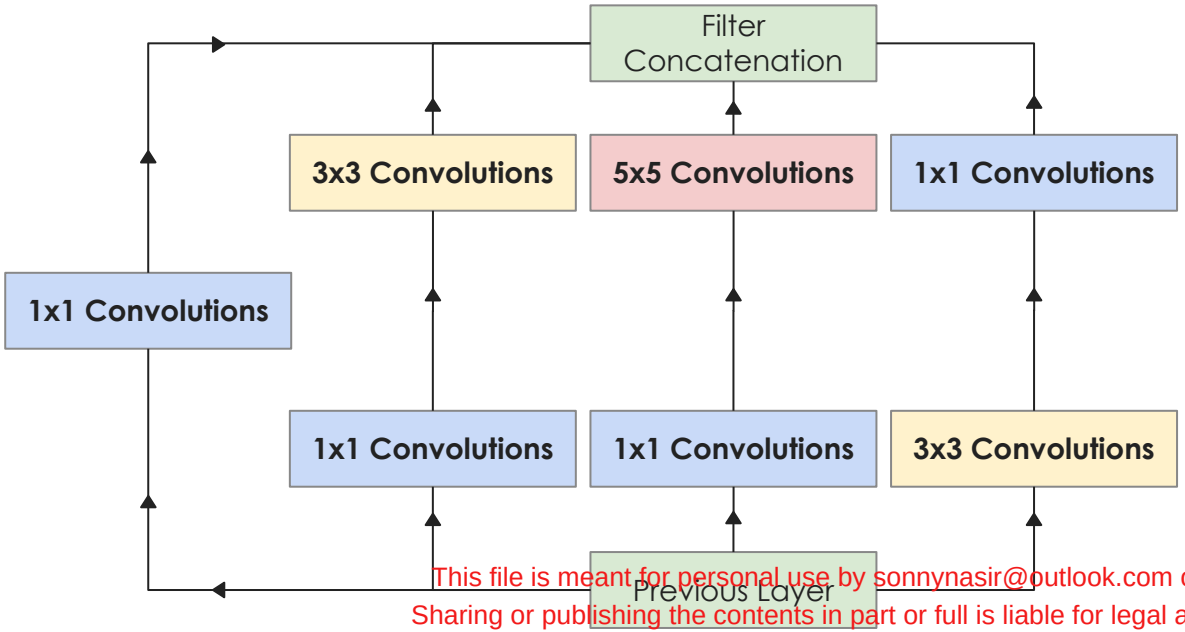
- Suppose a layer in our CNN has learned to focus on individual parts of the face. The next layer should then focus on the overall face in the image, and recognize it. **To perform this recognition, the layers should have the appropriate filter sizes for the recognition task of that layer.**
  The Inception Network, unlike other architectures of the time, **provides scope for this idea.**



The Inception Network - Inception Module

- What happens in the Inception Module is that by providing choices of different filter sizes, **the network itself learns to rely more on larger filter sizes (like 5x5 Convolutions) the deeper we go into the network**, as those larger filter sizes are more appropriate for higher-level feature detection such as recognizing a whole face, as opposed to smaller parts like eyes or ears.

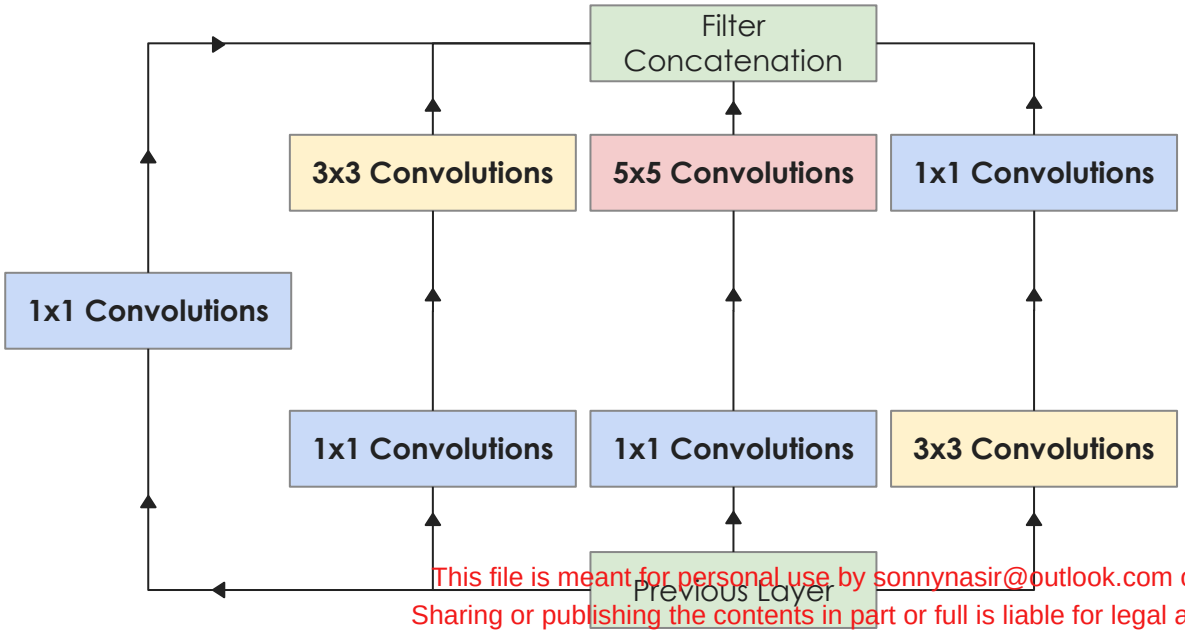
The Inception Module



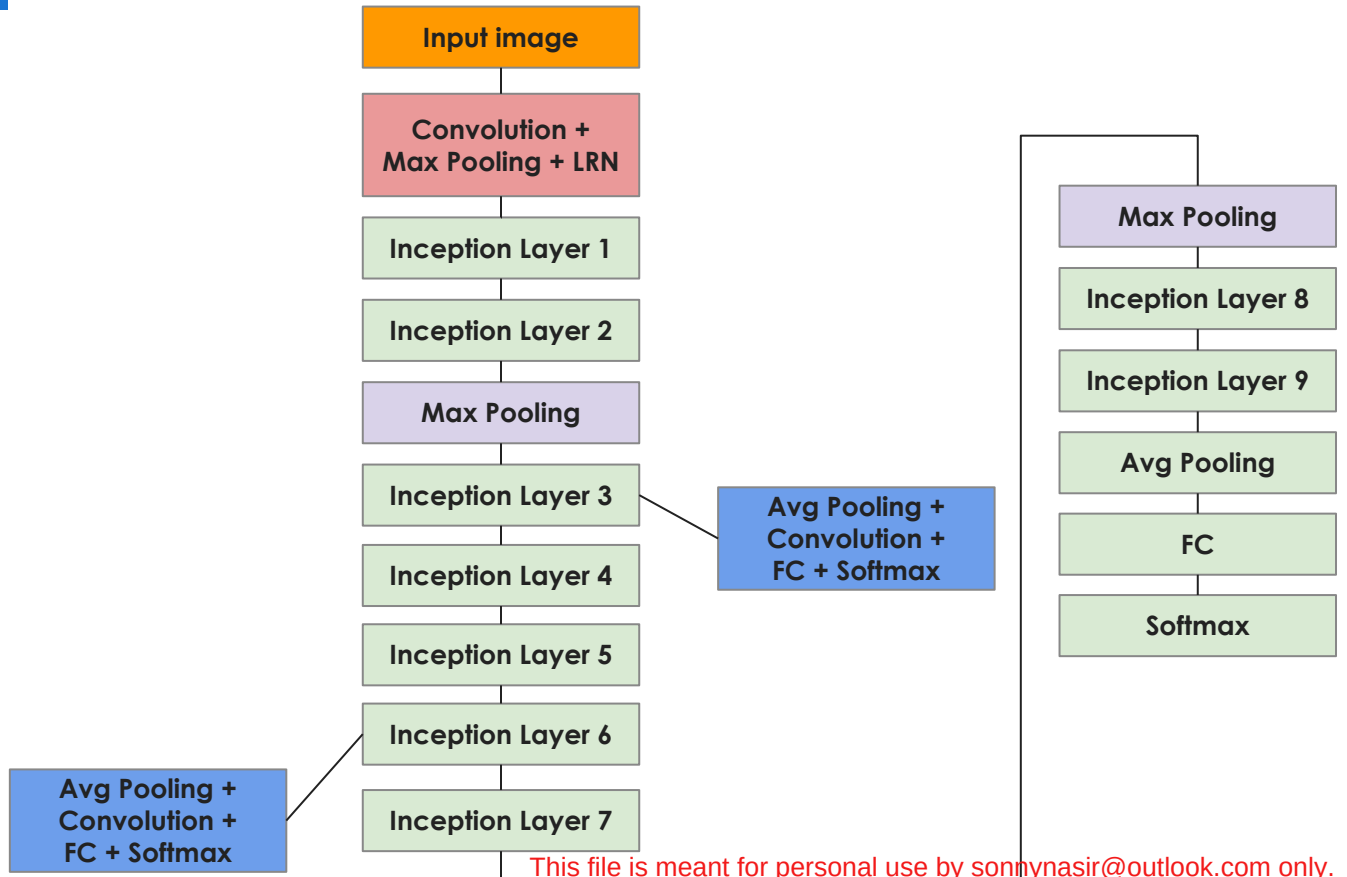
The Inception Network - Inception Module

- **The flexibility to make this choice**, which was not provided by other architectures of the time, is what allowed the Inception Module, and ultimately the Inception Network, to **achieve state-of-the-art performance on image prediction tasks.**

The Inception Module

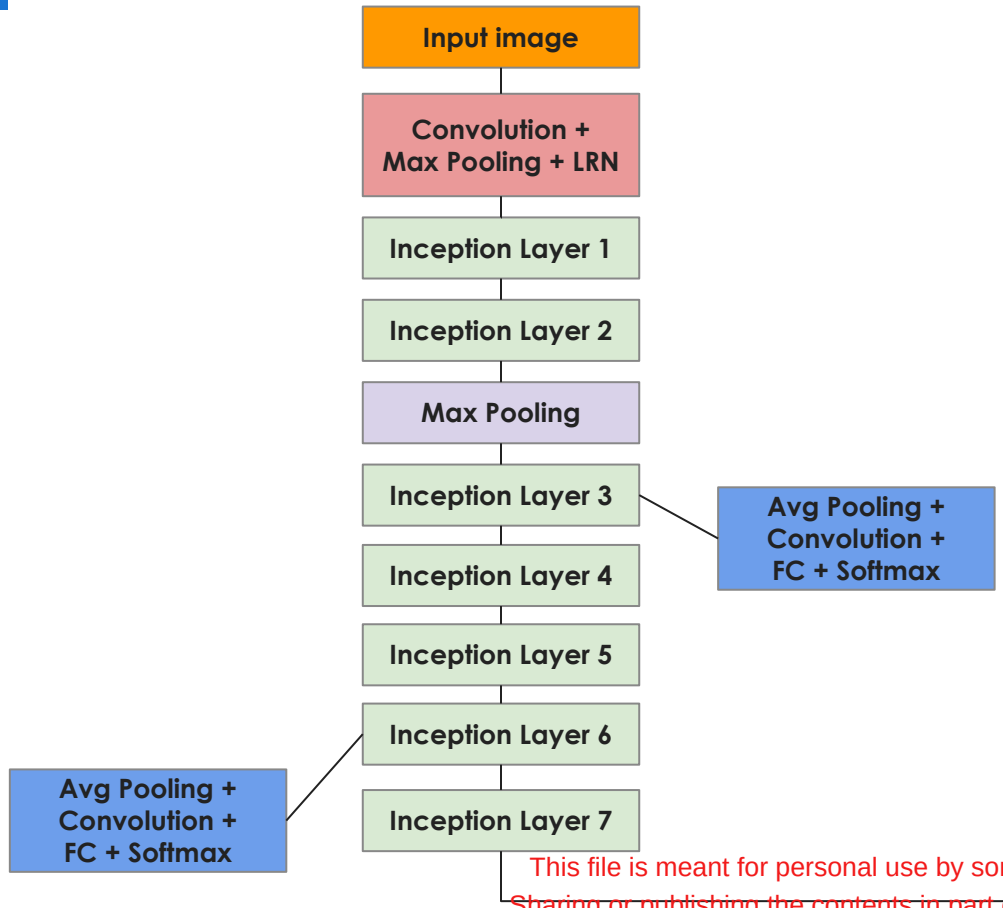


The Inception Network - Architecture



The actual Inception Network consists of **multiple Inception Layers throughout the architecture**, in addition to the other, more standard building blocks, like Convolution, Pooling and Fully Connected layers.

The Inception Network - Architecture



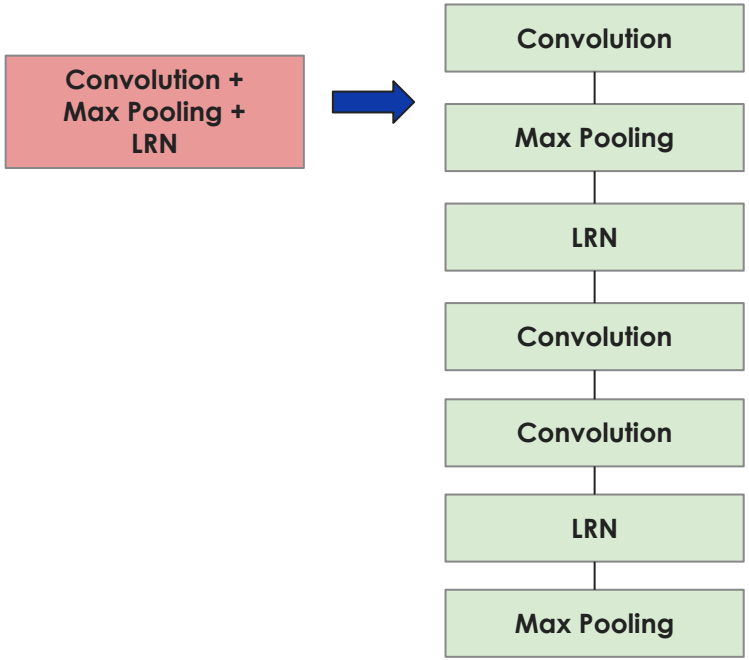
Another unique feature of InceptionNet is that at certain Inception Layers (Inception Layers 3 & 6 in this diagram), **outputs are taken at those intermediate layers as well, to create auxiliary classifiers.** The loss function combines the outputs of these auxiliary classifiers with the output of the main chain of the network, as an ensembling technique to reduce error.

The Inception Network - Architecture

- Prior to the Inception Network, researchers believed that increasing the depth of the network (in terms of the number of layers) was the best way to improve performance. With InceptionNet however, researchers realized that **the width of the network** (the number of feature maps concatenated together at each layer) **was also important in addition to its depth**, as a way to **improve model performance with minimal increase in the model's computational requirements.**

- As seen in the architecture diagram earlier, the Inception Network has three main structural components:
  - **The Inception Module**
  - **The [Convolution + Max Pooling + LRN] Module**
  - **The [Avg Pooling + Convolution + Fully Connected + Softmax] Module**

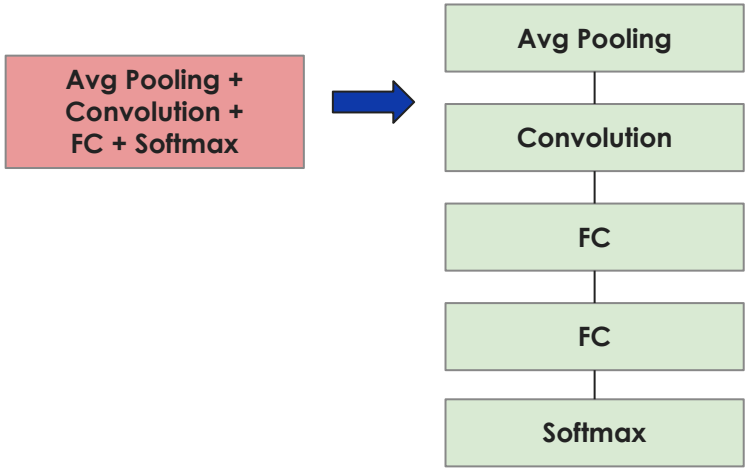
The Inception Network - Convolution + Max Pooling + LRN



- This is the first structural module of InceptionNet after the input image. This module has 3 Convolution layers, 2 Max Pooling layers and **2 LRN (Local Response Normalization) layers.**

- LRN is a non-trainable layer that **square-normalizes each pixel value along its entire depth across all feature maps** (within the local neighbourhood). The main reason for using LRN over **Batch Normalization** was to encourage **"lateral inhibition"**, or **maintaining the contrast values of the neighborhood of an image,** so that the locally maximum pixel values maintain their importance for the next layers.

The Inception Network - Avg Pooling + Conv + FC + Softmax



- In CNNs, **as the network grows deeper, the chances of the network being subject to the Vanishing Gradient problem increases.**
- To prevent the middle of the Inception Network from dying out, **two auxiliary classifiers were introduced**, consisting of **Avg Pooling, Convolution, Fully Connected and Softmax Layers**. These auxiliary classifiers apply Softmax to the outputs of two inception modules, and compute an auxiliary loss.
- The total loss function of the InceptionNet equals **the weighted sum of the real loss of the main chain and the auxiliary loss.**

The Inception Network - Higher Computational Efficiency

- The other advantage of the Inception Network is that it has only 5 million parameters.

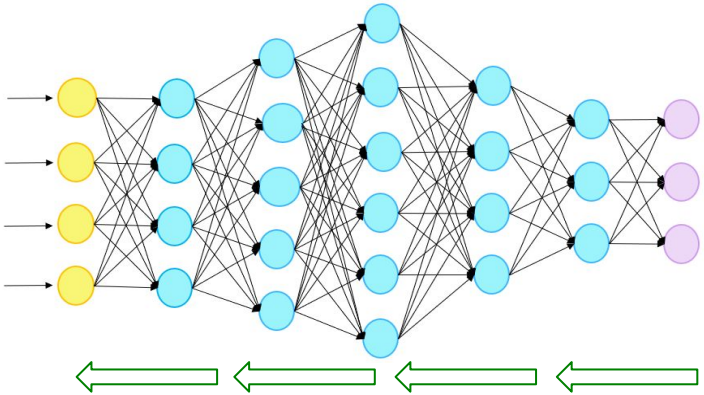
  In comparison to VGGNet, it has 27 times less parameters.

- **The increased width of each module (no. of feature maps concatenated)**, and **the use of 1x1 convolutions** in the network architecture is what allows InceptionNet to maintain top performance despite this reduced computational cost.

Residual Networks (ResNets)

Residual Networks (ResNets)

- ResNet, the winner of ILSVRC 2015, was developed by Microsoft researchers and achieved a significant improvement on the Inception Network's performance on ImageNet.
- While increasing the depth of the network (the number of layers) has generally shown to improve the performance of Deep Learning models on image prediction tasks, researchers found that after a threshold, the performance actually starts to degrade with an increased number of layers.
- This could be due to the optimization function, the network setup, and most critically, because of the Vanishing Gradient problem. Hence, **some alternate mechanism was required to get around the Vanishing Gradient problem** that deep neural networks face a challenge with, and **this was the problem that ResNets were able to solve.**



The Vanishing Gradient problem

Residual Networks (ResNets)

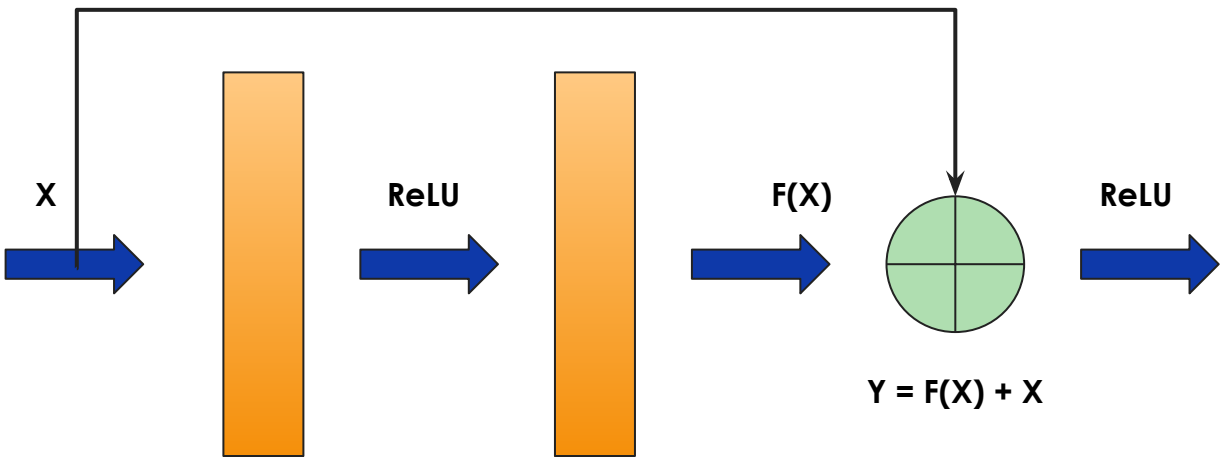
- **How do ResNets solve the Vanishing Gradient problem?**

  ResNets employ a new mechanism called **Skip Connections.** The Skip Connections of ResNets are able to solve the Vanishing Gradient problem of deep neural networks by allowing for an **alternate shortcut path** for the gradient to flow through, and have the effect of preventing the gradients from reaching a very small value or vanishing.

  The Skip Connections of ResNets are a repeatable block of operations, called a **Residual Block**, and this is what gives Residual Networks their name.

Residual Networks (ResNets)

- **What does a Residual Block look like?**

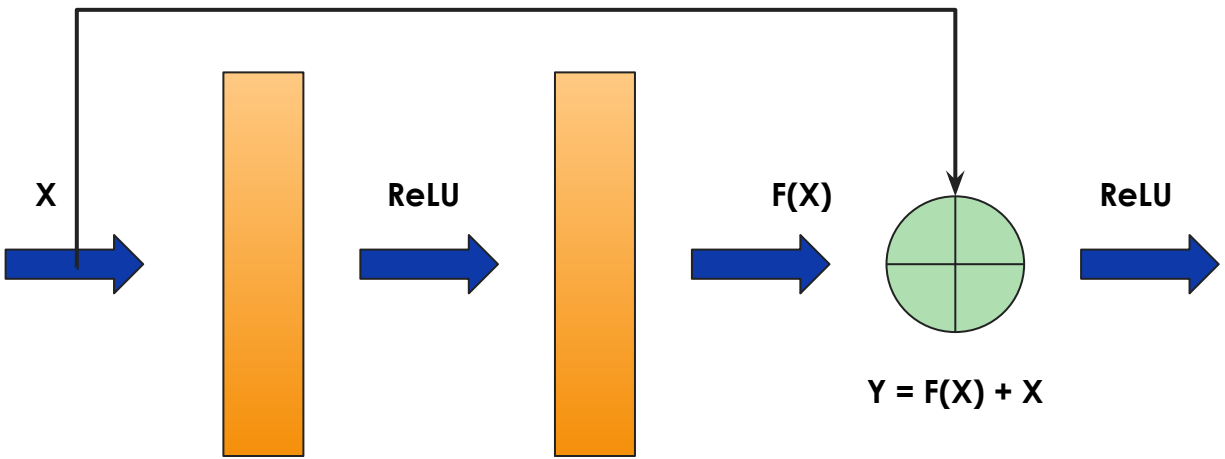


Residual Block

The Residual Block essentially **adds the original input (X) of the previous layer**, **to the linear portion of the output of the next layer F(X)**, before a non-linear activation function such as ReLU is applied. So this subsequent **ReLU would now be applied on F(X) + X**, instead of being applied on just F(X).

Residual Networks (ResNets)

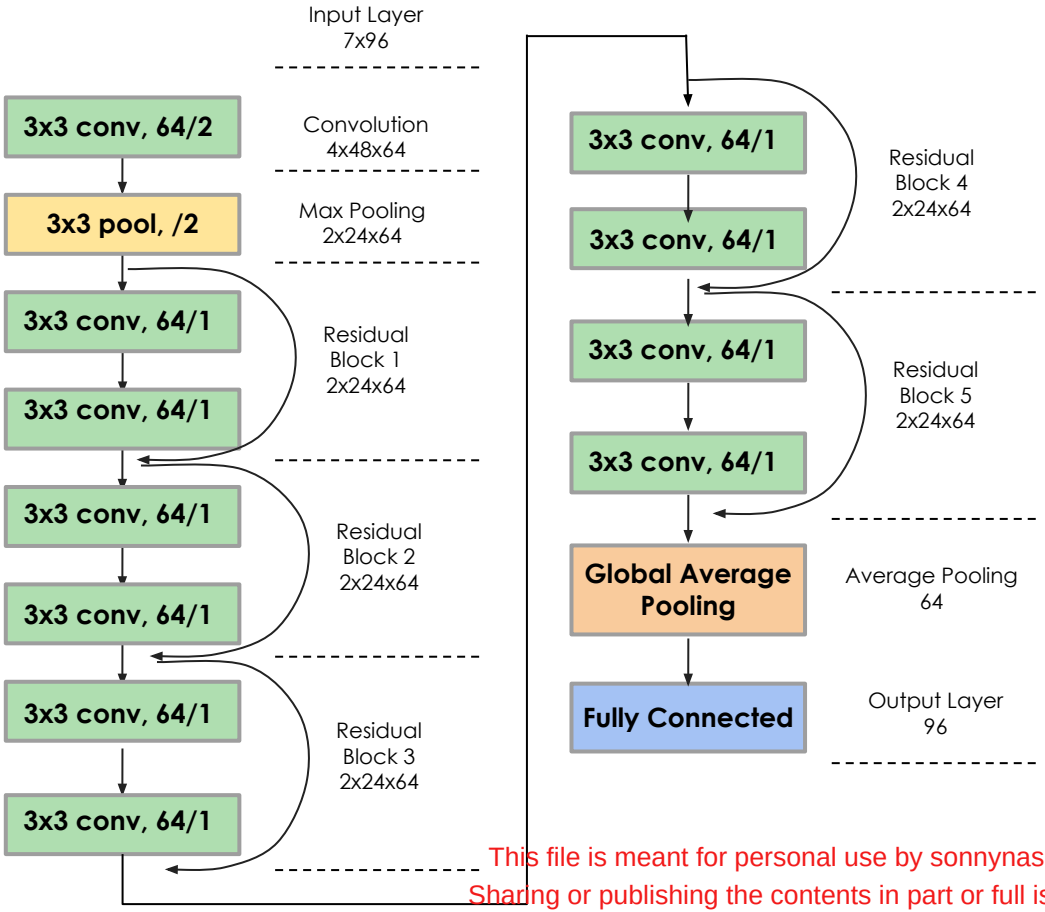
- **What does a Residual Block look like?**



Residual Block

- Hence, if F(X) were to become zero due to the Vanishing Gradient problem, then the ReLU would still be applied on F(X) + X which is still X, since F(X) is now zero. This would prevent the input to the ReLU from becoming zero, and **this is what enables the model to sidestep the issue of Vanishing Gradients.**

ResNet Architecture - ResNet-12



- This is an example of the **ResNet-12 Architecture** (which only has 12 layers).
- The Residual Blocks show the **Skip Connections** highlighted earlier, which **utilize the input of the previous layer in computing the activation output of a subsequent layer.**
- The same principle has been utilized to build much deeper Residual Network architectures, such as ResNet-34, ResNet-50 and ResNet-100, which show state-of-the-art performance on image prediction tasks.

Summary

So in order to summarize:

- We've gone over the intuition behind some of the most well-known deep learning architectures used in computer vision, and the unique characteristics of each model.
- In computer vision, as in all of machine learning, there is no hard-and-fast rule to determine which architecture works best for a specific use case. Ideally, you would start with a simple model and work your way up in terms of complexity. Within your constraints (training resource availability, time/memory limit during inference), you may choose the model that performs best.
- With respect to Transfer Learning models, it is quite simple to import the weights and architecture of the pre-trained model using TensorFlow/Keras and fine-tune the model with data specific to the prediction problem at hand. This is normally a faster way to reach a high-performance predictive model than by creating a custom CNN architecture from scratch, due to the complexity of the image prediction problems found in computer vision.

Thank You