# Types of Artificial Neural Networks Currently Being Used in Machine Learning

1. Feedforward Neural Network -Artificial neural network ANN
2. Multilayer perceptron (MLP)
3. Convolutional neural network (CNN)
4. Recursive neural network (RNN)
5. Recurrent neural network (RNN)
6. Long short-term memory (LSTM) : Improved RNN
7. Radial basis function Neural Network (RBF)
8. Hopfield Networks : Type of Recurrent neural network (RNN)
9. Boltzmann Machine Network: Type of Recurrent neural network (RNN)

**10 Advanced Deep Learning Architectures Data Scientists Should Know!**

1. Alexnet
2. VGG (Visual Graphics Group at oxford)
3. Google net or Inception net: GoogleNet was the winner of ImageNet 2014
4. Res Net (Residual Network)
5. ResNeXt : It builds upon the concepts of inception
6. RCNN(Region based CNN)
7. YOLO( You only look once)
8. SquuezeNet
9. SegNet
10. GAN

## Introduction

It is becoming very hard to stay up to date with recent advancements happening in deep learning. Hardly a day goes by without a new innovation or a new application of deep learning coming by. However, most of these advancements are hidden inside the large amount of research papers that are published on mediums like ArXiv / Springer

To keep ourselves updated, we have created a small reading group to share our learnings internally at Analytics Vidhya. One such learning I would like to share with the community is a a survey of advanced architectures which have been developed by the research community.

This article contains some of the **recent advancements in Deep Learning along with codes for implementation in keras library**. I have also provided links to the original papers, in case you are interested in reading them or want to refer them.

To keep the article concise, I have only considered the architectures which have been successful in Computer Vision domain.

If you are interested, read on!

**P.S.:** This article assumes the knowledge of neural networks and familiarity with keras. If you need to catch up on these topics, I would strongly recommend you read the following articles first:

* [Fundamentals of Deep Learning – Starting with Artificial Neural Network](https://www.analyticsvidhya.com/blog/2016/03/introduction-deep-learning-fundamentals-neural-networks/)
* [Tutorial: Optimizing Neural Networks using Keras (with Image recognition case study)](https://www.analyticsvidhya.com/blog/2016/10/tutorial-optimizing-neural-networks-using-keras-with-image-recognition-case-study/)

## Table of Contents

* What do we mean by an advanced architecture?
* Types of Computer Vision Tasks
* List of Deep Learning Architectures

## What do we mean by an Advanced Architecture?

Deep Learning algorithms consists of such a diverse set of models in comparison to a single traditional machine learning algorithm. This is because of the flexibility that neural network provides when building a full fledged end-to-end model.

Neural network can sometimes be compared with lego blocks, where you can build almost any simple to complex structure your imagination helps you to build.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131633/temp1.png

We can define an advanced architecture as one that has a proven track record of being a successful model. This is mainly seen in challenges like ImageNet, where your task is to solve a problem, say image recognition, using the data given. Those who don’t know what ImageNet is, it is the dataset which is provided in ILSVR (ImageNet Large Scale Visual Recognition) challenge.

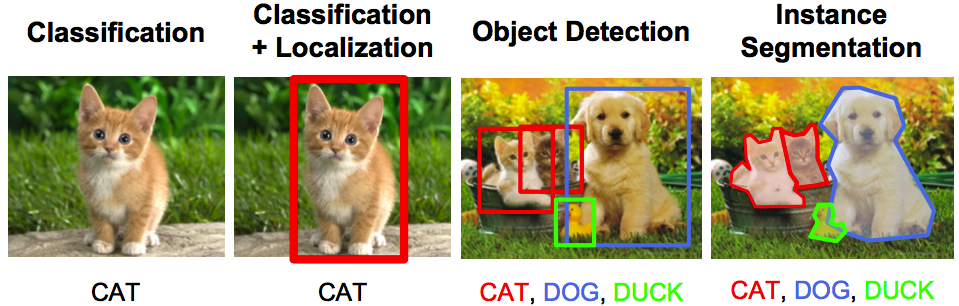
Also as described in the below mentioned architectures, each of them has a nuance which sets them apart from the usual models; giving them an edge when they are used to solve a problem. These architectures also fall in the category of “deep” models, so they are likely to perform better than their shallow counterparts.

## Types of Computer Vision Tasks

This article is mainly focused on Computer Vision, so it is natural to describe the horizon of computer vision tasks. Computer Vision; as the name suggests is simply creating artificial models which can replicate the visual tasks performed by a human. This essentially means what we can see and what we perceive is a process which can be understood and implemented in an artificial system.

The main types of tasks that computer vision can be categorised in are as follows:

* **Object Recognition / classification** – In object recognition, you are given a raw image and your task is to identify which class does the image belong to.
* **Classification + Localisation** – If there is only one object in the image, and your task is to find the location of that object, a more specific term given to this problem is localisation problem.
* **Object Detection** – In object detection, you task is to identify where in the image does the objects lies in. These objects might be of the same class or different class altogether.
* **Image Segmentation** – Image Segmentation is a bit sophisticated task, where the objective is to map each pixel to its rightful class.



## List of Deep Learning Architectures

Now that we have understood what an advanced architecture is and explored the tasks of computer vision, let us list down the most important architectures and their descriptions:

### 1. AlexNet

AlexNet is the first deep architecture which was introduced by one of the pioneers in deep learning – Geoffrey Hinton and his colleagues. It is a simple yet powerful network architecture, which helped pave the way for groundbreaking research in Deep Learning as it is now. Here is a representation of the architecture as proposed by the authors.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131757/temp6.png

When broken down, AlexNet seems like a simple architecture with **convolutional** and pooling layers one on top of the other, followed by fully connected layers at the top. This is a very simple architecture, which was conceptualised way back in 1980s. The things which set apart this model is the scale at which it performs the task and the use of GPU for training. In 1980s, CPU was used for training a neural network. Whereas AlexNet speeds up the training by 10 times just by the use of GPU.

Although a bit outdated at the moment, AlexNet is still used as a starting point for applying deep neural networks for all the tasks, whether it be computer vision or speech recognition.

***Convolutional Neural Networks***

CNNs have wide applications in image and video recognition, recommender systems and natural language processing. In this article, the example that I will take is related to Computer Vision. However, the basic concept remains the same and can be applied to any other use-case!

### ****CNNs operate over Volumes****

Unlike neural networks, where the input is a vector, here the input is a multi-channeled image

### 2. VGG Net

The VGG Network was introduced by the researchers at Visual Graphics Group at Oxford (hence the name VGG). This network is specially characterized by its pyramidal shape, where the bottom layers which are closer to the image are wide, whereas the top layers are deep.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131808/temp7.png

As the image depicts, VGG contains subsequent convolutional layers followed by pooling layers. The pooling layers are responsible for making the layers narrower. In their paper, they proposed multiple such types of networks, with change in deepness of the architecture.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131823/temp8.png

The advantages of VGG are :

* It is a very good architecture for benchmarking on a particular task.
* Also, pre-trained networks for VGG are available freely on the internet, so it is commonly used out of the box for various applications.

On the other hand, its main disadvantage is that it is very slow to train if trained from scratch. Even on a decent GPU, it would take more than a week to get it to work.

* [**Original Paper link**](https://arxiv.org/abs/1409.1556)
* [**Link for code implementation**](https://github.com/fchollet/keras/blob/master/keras/applications/vgg16.py)

### 3. GoogleNet

GoogleNet (or Inception Network) is a class of architecture designed by researchers at Google. GoogleNet was the winner of ImageNet 2014, where it proved to be a powerful model.

In this architecture, along with going deeper (it contains 22 layers in comparison to VGG which had 19 layers), the researchers also made a novel approach called the Inception module.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131905/temp10.png

As seen above, it is a drastic change from the sequential architectures which we saw previously. In a single layer, multiple types of “feature extractors” are present. This indirectly helps the network perform better, as the network at training itself has many options to choose from when solving the task. It can either choose to convolve the input, or to pool it directly.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131838/temp9.png

The final architecture contains multiple of these inception modules stacked one over the other. Even the training is slightly different in GoogleNet, as most of the topmost layers have their own output layer. This nuance helps the model converge faster, as there is a joint training as well as parallel training for the layers itself.

The advantages of GoogleNet are :

* GoogleNet trains faster than VGG.
* Size of a pre-trained GoogleNet is comparatively smaller than VGG. A VGG model can have >500 MBs, whereas GoogleNet has a size of only 96 MB

GoogleNet does not have an immediate disadvantage per se, but further changes in the architecture  are proposed, which make the model perform better. One such change is termed as an Xception Network, in which the limit of divergence of inception module (4 in GoogleNet as we saw in the image above) are increased. It can now theoretically be infinite (hence called extreme inception!)

* [**Original Paper link**](http://arxiv.org/abs/1512.00567)
* [**Link for code implementation**](https://github.com/fchollet/keras/blob/master/keras/applications/inception_v3.py)

### 4. ResNet

ResNet is one of the monster architectures which truly define how deep a deep learning architecture can be. Residual Networks (ResNet in short) consists of multiple subsequent residual modules, which are the basic building block of ResNet architecture. A representation of residual module is as follows

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131914/temp11.png

In simple words, a residual module has two options, either it can perform a set of functions on the input, or it can skip this step altogether.

Now similar to GoogleNet, these residual modules are stacked one over the other to form a complete end-to-end network.

## https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/08/08131926/temp12.png

A few more novel techniques which ResNet introduced are:

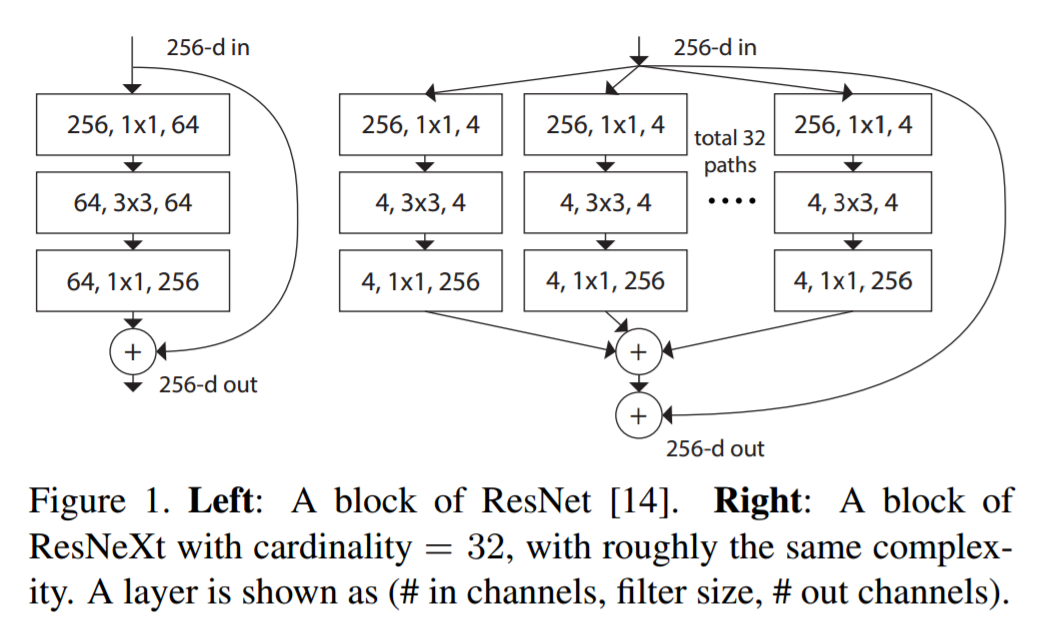
* Use of standard SGD instead of a fancy adaptive learning technique. This is done along with a reasonable initialization function which keeps the training intact
* Changes in preprocessing the input, where the input is first divided into patches and then feeded into the network

The main advantage of ResNet is that hundreds, even thousands of these residual layers can be used to create a network and then trained. This is a bit different from usual sequential networks, where you see that there is reduced performance upgrades as you increase the number of layers.

* [**Original Paper link**](https://arxiv.org/abs/1512.03385)
* [**Link for code implementation**](https://github.com/fchollet/keras/blob/master/keras/applications/resnet50.py)

### 5. ResNeXt

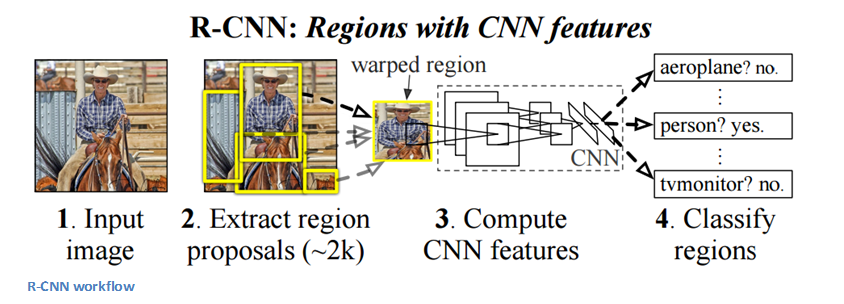
ResNeXt is said to be the current state-of-the-art technique for object recognition. It builds upon the concepts of inception and resnet to bring about a new and improved architecture. Below image is a summarization of how a residual module of ResNeXt module looks like.



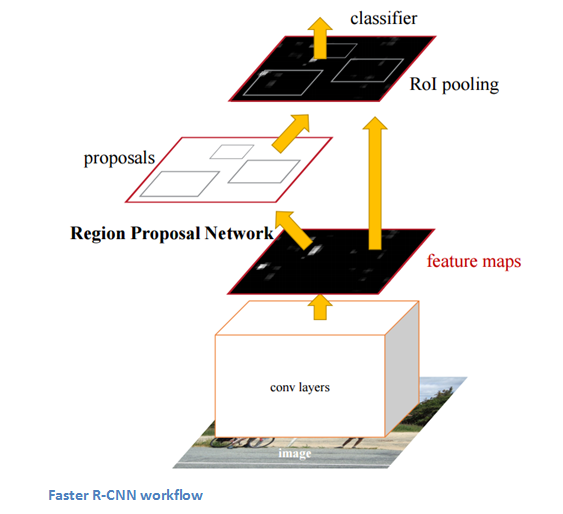
* [**Original Paper link**](https://arxiv.org/pdf/1611.05431.pdf)
* [**Link for code implementation**](https://github.com/titu1994/Keras-ResNeXt)

### 6. RCNN (Region Based CNN)

Region Based CNN architecture is said to be the most influential of all the deep learning architectures that have been applied to object detection problem. To solve detection problem, what RCNN does is to attempt to draw a bounding box over all the objects present in the image, and then recognize what object is in the image. It works as follows:



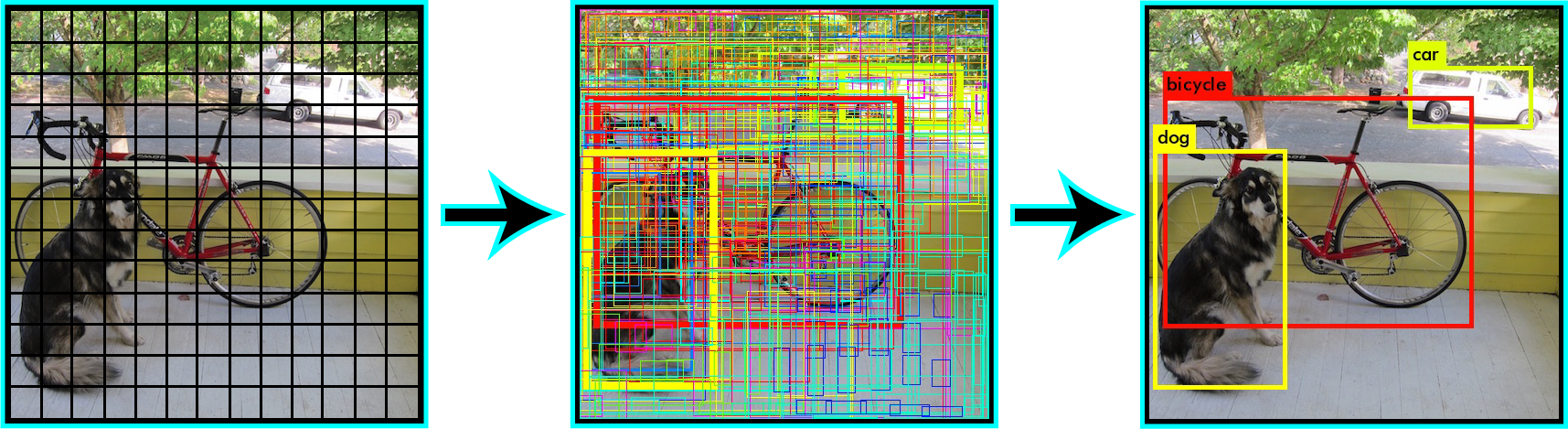
The structure of RCNN is as follows:



* [**Original Paper link**](https://arxiv.org/abs/1506.01497)
* [**Link for code implementation**](https://github.com/yhenon/keras-frcnn)

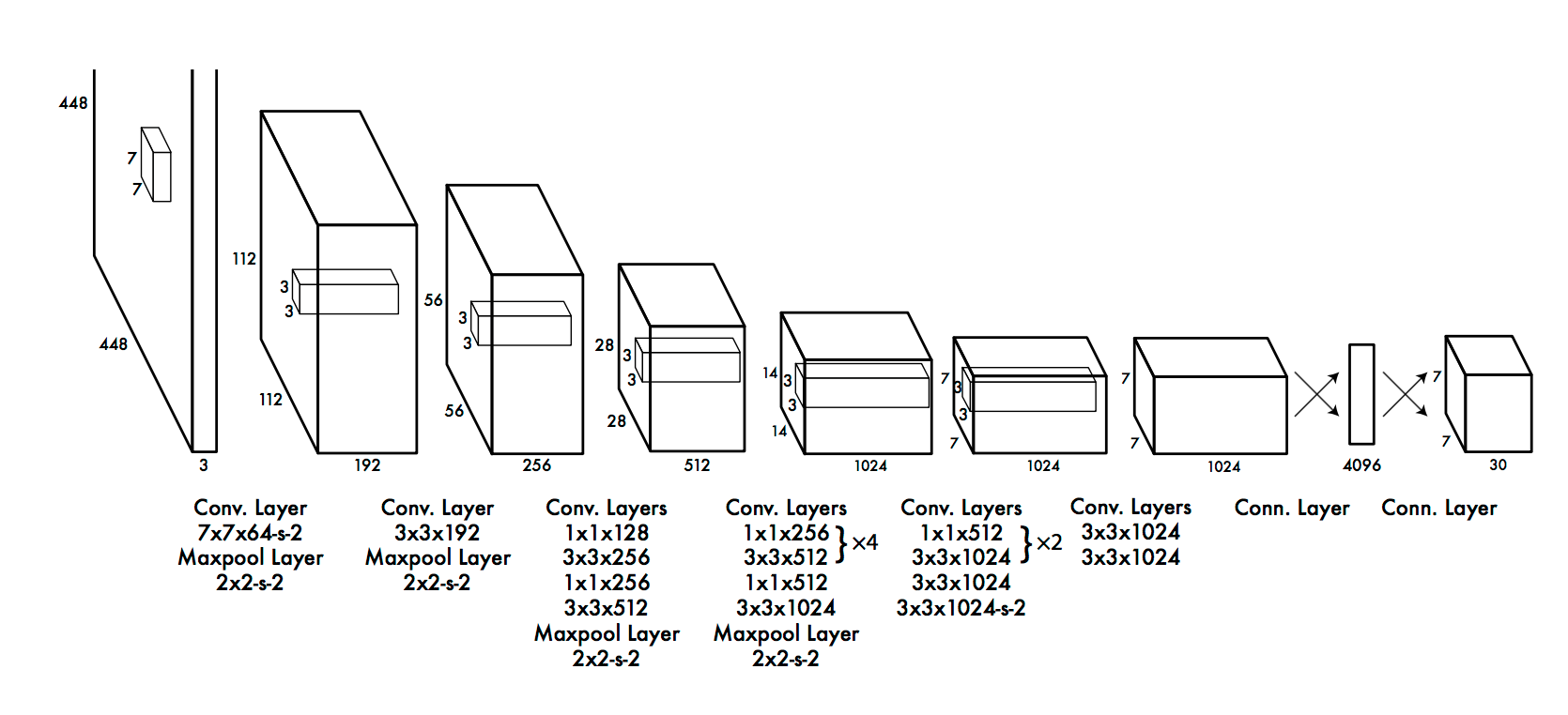
### 7. YOLO (You Only Look Once)

YOLO is the current state-of-the-art real time system built on deep learning for solving image detection problems. As seen in the below given image, it first divides the image into defined bounding boxes, and then runs a recognition algorithm in parallel for all of these boxes to identify which object class do they belong to. After identifying this classes, it goes on to merging these boxes intelligently to form an optimal bounding box around the objects.



All of this is done in parallely, so it can run in real time; processing upto 40 images in a second.

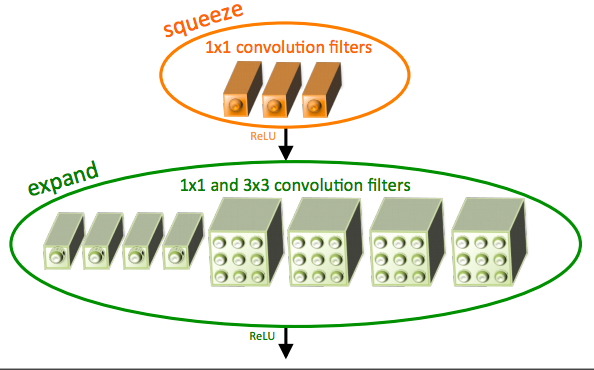
Although it gives reduced performance than its RCNN counterpart, it still has an advantage of being real time to be viable for use in day-to-day problems. Here is a representation of architecture of YOLO



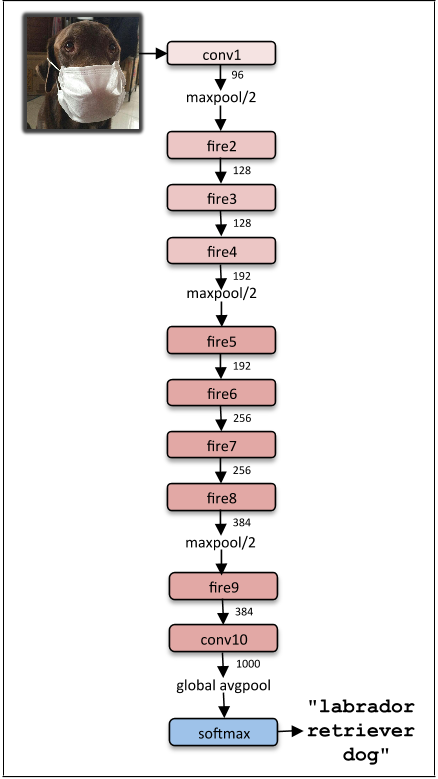
* [**Original Paper link**](https://pjreddie.com/media/files/papers/yolo.pdf)
* [**Link for code implementation**](https://github.com/allanzelener/YAD2K)

### 8. SqueezeNet

The squeezeNet architecture is one more powerful architecture which is extremely useful in low bandwidth scenarios like mobile platforms. This architecture has occupies only 4.9MB of space, on the other hand, inception occupies ~100MB! This drastic change is brought up by a specialized structure called the fire module. Below image is a representation of fire module.



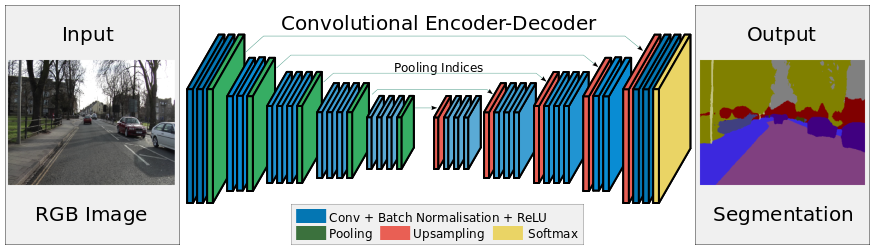
The final architecture of squeezeNet is as follows:



* [**Original Paper link**](https://arxiv.org/abs/1602.07360)
* [**Link for code implementation**](https://github.com/rcmalli/keras-squeezenet)

### 9. SegNet

SegNet is a deep learning architecture applied to solve image segmentation problem. It consists of sequence of processing layers (encoders) followed by a corresponding set of decoders for a pixelwise classification . Below image summarizes the working of SegNet.

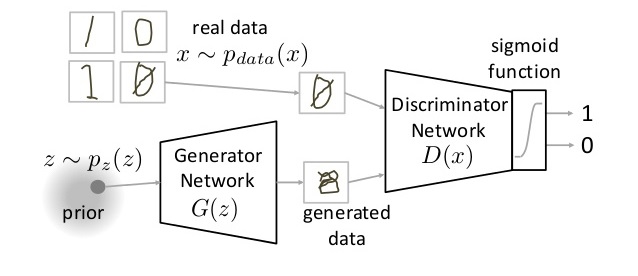


One key feature of SegNet is that it retains high frequency details in segmented image as the pooling indices of encoder network is connected to pooling indices of decoder networks. In short, the information transfer is direct instead of convolving them. SegNet is one the the best model to use when dealing with image segmentation problems

* [**Original Paper link**](https://arxiv.org/abs/1511.00561)
* [**Link for code implementation**](https://github.com/imlab-uiip/keras-segnet)

### 10. GAN (Generative Adversarial Network)

GAN is an entirely different breed of neural network architectures, in which a neural network is used to generate an entirely new image which is not present is the training dataset, but is realistic enough to be in the dataset. For example, below image is a breakdown of GANs are made of. I have covered [how GANs work in this article](https://www.analyticsvidhya.com/blog/2017/06/introductory-generative-adversarial-networks-gans/). Go through it if you are curious.



* [**Original Paper link**](https://arxiv.org/abs/1406.2661)
* [**Link for code implementation**](https://github.com/bstriner/keras-adversarial)

## End Notes

In this article, I have covered an overview of major deep learning architectures that you should get familiar with. If you have any questions on deep learning architectures, please feel free to share them with me through comments.

## VGGNet, ResNet, Inception, and Xception with Keras

In the first half of this blog post I’ll briefly discuss the VGG, ResNet, Inception, and Xception network architectures included in the Keras library.

We’ll then create a custom Python script using Keras that can load these pre-trained network architectures from disk and classify your own input images.

Finally, we’ll review the results of these classifications on a few sample images.

### State-of-the-art deep learning image classifiers in Keras

Keras ships out-of-the-box with five Convolutional Neural Networks that have been pre-trained on the ImageNet dataset:

1. VGG16
2. VGG19
3. ResNet50
4. Inception V3
5. Xception

Let’s start with a overview of the ImageNet dataset and then move into a brief discussion of each network architecture.

#### What is ImageNet?

[ImageNet](http://image-net.org/) is formally a project aimed at (manually) labeling and categorizing images into almost 22,000 separate object categories for the purpose of computer vision research.

However, when we hear the term “ImageNet” in the context of deep learning and Convolutional Neural Networks, we are likely referring to the [ImageNet Large Scale Visual Recognition Challenge](http://www.image-net.org/challenges/LSVRC/), or ILSVRC for short.

The goal of this image classification challenge is to train a model that can correctly classify an input image into 1,000 separate object categories.

Models are trained on ~1.2 million training images with another 50,000 images for validation and 100,000 images for testing.

These 1,000 image categories represent object classes that we encounter in our day-to-day lives, such as species of dogs, cats, various household objects, vehicle types, and much more. You can find the full list of object categories in the ILSVRC challenge [here](http://image-net.org/challenges/LSVRC/2014/browse-synsets).

When it comes to image classification, the ImageNet challenge is the de facto benchmark for computer vision classification algorithms — and the leaderboard for this challenge has been **dominated** by Convolutional Neural Networks and deep learning techniques since 2012.

The state-of-the-art pre-trained networks included in the Keras core library represent some of the highest performing Convolutional Neural Networks on the ImageNet challenge over the past few years. These networks also demonstrate a strong ability to generalize to images outside the ImageNet dataset via transfer learning, such as feature extraction and fine-tuning.

#### VGG16 and VGG19

**Figure 1:** A visualization of the VGG architecture ([source](https://www.cs.toronto.edu/~frossard/post/vgg16/)).

The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper, [Very Deep Convolutional Networks for Large Scale Image Recognition](https://arxiv.org/abs/1409.1556).

This network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes are then followed by a softmax classifier (above).

The “16” and “19” stand for the number of weight layers in the network (columns D and E in **Figure 2** below):

**Figure 2:** Table 1 of [Very Deep Convolutional Networks for Large Scale Image Recognition](https://arxiv.org/abs/1409.1556), Simonyan and Zisserman (2014).

In 2014, 16 and 19 layer networks were considered very deep (although we now have the ResNet architecture which can be successfully trained at depths of 50-200 for ImageNet and over 1,000 for CIFAR-10).

Simonyan and Zisserman found training VGG16 and VGG19 challenging (specifically regarding convergence on the deeper networks), so in order to make training easier, they first trained smaller versions of VGG with less weight layers (columns A and C) first.

The smaller networks converged and were then used as initializations for the larger, deeper networks — this process is called ***pre-training***.

While making logical sense, pre-training is a very time consuming, tedious task, requiring an entire network to be trained **before** it can serve as an initialization for a deeper network.

We no longer use pre-training (in most cases) and instead prefer Xaiver/Glorot initialization or MSRA initialization (sometimes called He et al. initialization from the paper, [Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification](https://arxiv.org/abs/1502.01852)). You can read more about the importance of weight initialization and the convergence of deep neural networks inside [All you need is a good init](https://arxiv.org/abs/1511.06422), Mishkin and Matas (2015).

Unfortunately, there are two major drawbacks with VGGNet:

1. It is painfully slow to train.
2. The network architecture weights themselves are quite large (in terms of disk/bandwidth).

Due to its depth and number of fully-connected nodes, VGG is over 533MB for VGG16 and 574MB for VGG19. This makes deploying VGG a tiresome task.

We still use VGG in many deep learning image classification problems; however, smaller network architectures are often more desirable (such as SqueezeNet, GoogLeNet, etc.).

#### ResNet

Unlike traditional sequential network architectures such as AlexNet, OverFeat, and VGG, ResNet is instead a form of “exotic architecture” that relies on micro-architecture modules (also called “network-in-network architectures”).

The term micro-architecture refers to the set of “building blocks” used to construct the network. A collection of micro-architecture building blocks (along with your standard CONV, POOL, etc. layers) leads to the macro-architecture (i.e,. the end network itself).

First introduced by He et al. in their 2015 paper, [Deep Residual Learning for Image Recognition](https://arxiv.org/abs/1512.03385), the ResNet architecture has become a seminal work, demonstrating that extremely deep networks can be trained using standard SGD (and a reasonable initialization function) through the use of residual modules:

**Figure 3:** The residual module in ResNet as originally proposed by He et al. in 2015.

Further accuracy can be obtained by updating the residual module to use identity mappings, as demonstrated in their 2016 followup publication, [Identity Mappings in Deep Residual Networks](https://arxiv.org/abs/1603.05027):

**Figure 4:** (Left) The original residual module. (Right) The updated residual module using pre-activation.

That said, keep in mind that the ResNet50 (as in 50 weight layers) implementation in the Keras core is based on the former 2015 paper.

Even though ResNet is much deeper than VGG16 and VGG19, the model size is actually substantially smaller due to the usage of global average pooling rather than fully-connected layers — this reduces the model size down to 102MB for ResNet50.

#### Inception V3

The “Inception” micro-architecture was first introduced by Szegedy et al. in their 2014 paper, [Going Deeper with Convolutions](https://arxiv.org/abs/1409.4842):

**Figure 5:** The original Inception module used in GoogLeNet.

The goal of the inception module is to act as a “multi-level feature extractor” by computing 1×1, 3×3, and 5×5 convolutions within the same module of the network — the output of these filters are then stacked along the channel dimension and before being fed into the next layer in the network.

The original incarnation of this architecture was called GoogLeNet, but subsequent manifestations have simply been called Inception vN where N refers to the version number put out by Google.

The Inception V3 architecture included in the Keras core comes from the later publication by Szegedy et al., [*Rethinking the Inception Architecture for Computer Vision*](https://arxiv.org/abs/1512.00567) (2015) which proposes updates to the inception module to further boost ImageNet classification accuracy.

The weights for Inception V3 are smaller than both VGG and ResNet, coming in at 96MB.

#### Xception

**Figure 6:** The Xception architecture.

Xception was proposed by none other than [François Chollet](https://twitter.com/fchollet) himself, the creator and chief maintainer of the Keras library.

Xception is an extension of the Inception architecture which replaces the standard Inception modules with depthwise separable convolutions.

The original publication, Xception: Deep Learning with Depthwise Separable Convolutions can be found [here](https://arxiv.org/abs/1610.02357).

Xception sports the smallest weight serialization at only 91MB.

#### What about SqueezeNet?

**Figure 7:** The “fire” module in SqueezeNet, consisting of a “squeeze” and an “expand”. ([Iandola et al., 2016](https://arxiv.org/abs/1602.07360)).

For what it’s worth, the [SqueezeNet architecture](https://arxiv.org/abs/1602.07360) can obtain AlexNet-level accuracy (~57% rank-1 and ~80% rank-5) at only 4.9MB through the usage of “fire” modules that “squeeze” and “expand”.

While leaving a small footprint, SqueezeNet can also be very tricky to train.

That said, I demonstrate how to train SqueezeNet from scratch on the ImageNet dataset inside my upcoming book, [*Deep Learning for Computer Vision with Python*](https://www.pyimagesearch.com/deep-learning-computer-vision-python-book/).