PRETRAINED MODELS

problem with own models?

In order to get good performance or accuracy more amount of the data is required if the data is more then model can be trained well and predict the accurate results.

As we know that the deep learning models are **data hungry,** they will perform well when they trained on huge amount of data.

Let us take an image data as example to understand scenario.

Our model wanted to classify the image whether it is a dog or cat

In order to identify we need to train the model with different images of cats and dogs

Assume that in train data we have to label the output or classify the image whether it is a cat or dog.

So, we need human resources in order to label the data and so much time is required in order to classify the more images to get good predictions.

The companies are not ready to do develop their own models as time and money and computation cost is increases.

Pretrained models

Usually, pretrained models which are developed by others

This pretrained models where already trained with so many images and we are using them

Evolution of pretrained models

1. Image net dataset

* It is the visual database of image
* The **ImageNet** dataset contains 14,197,122 annotated images according to the WordNet hierarchy.
* It has 20,000 categories (class names)
* It has 1 million images having the bounding boxes
* Image net was implemented by the crowd sourcing they asked by the people about the image.

1. ILSVRC (image net large set visualize recognition challenge)

The image net is used in this challenge where many people developed their models by taking the subsets of images of image net in order to get good accuracy

Many developers are participated in this challenge

But they made model by using the ml

2010---------------------28% error rate ML

2011---------------------25% error rate ML

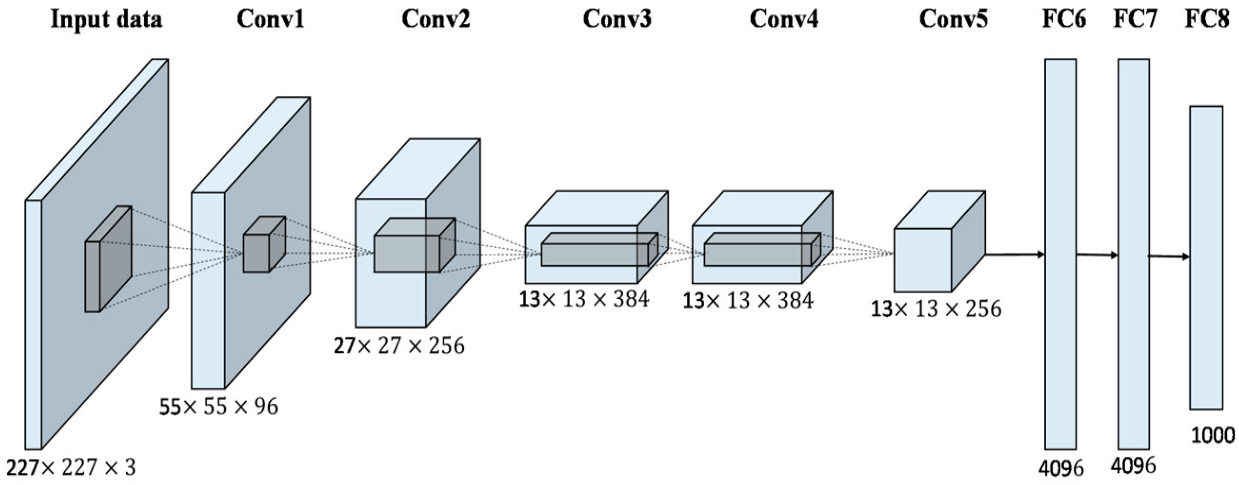
2012 --------------------- 16% error rate -----(Alex net) CNN **benchmark in DL**

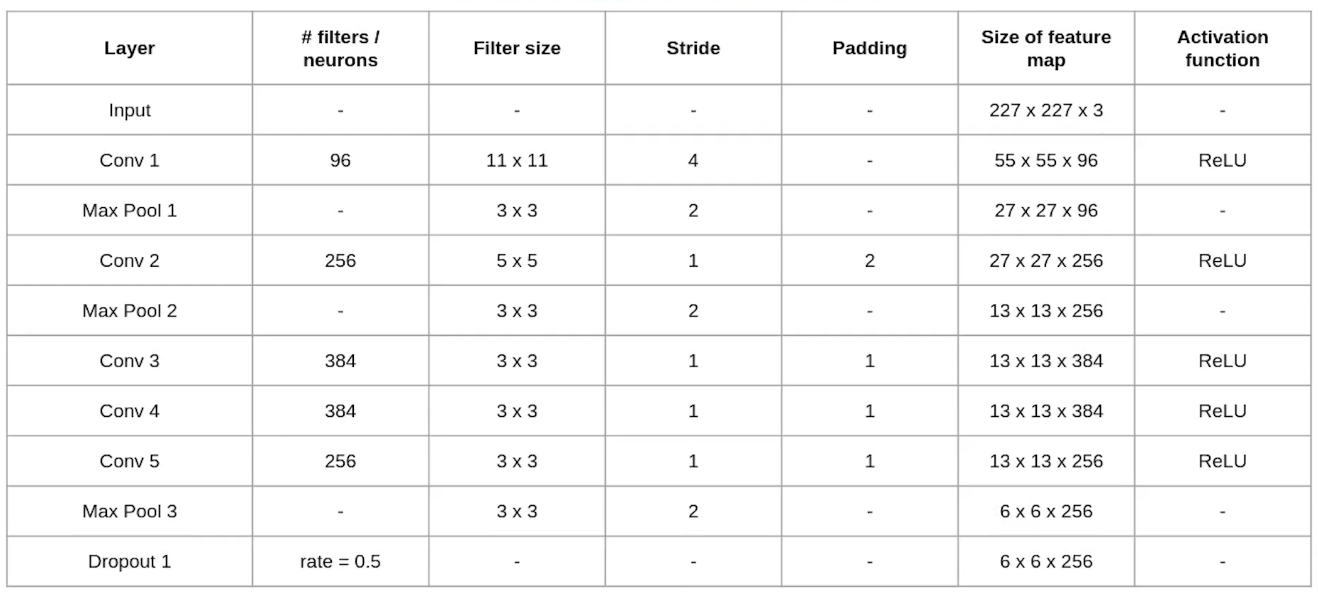
ALEXNET

Alexnet won the Imagenet large-scale visual recognition challenge in 2012

The Alexnet has eight layers with learnable parameters. The model consists of five layers with a combination of max pooling followed by 3 fully connected layers and they use Relu activation in each of these layers except the output layer.

They found out that using the relu as an activation function accelerated the speed of the training process by almost six times. They also used the dropout layers, that prevented their model from overfitting. Further, the model is trained on the Imagenet dataset. The Imagenet dataset has almost 14 million images across a thousand classes.





2012----------alexnet---------**16.4%** error rate

2013---------ZFnet----------**11.7%** error rate

2014---------VGG--------**7.3%** error rate

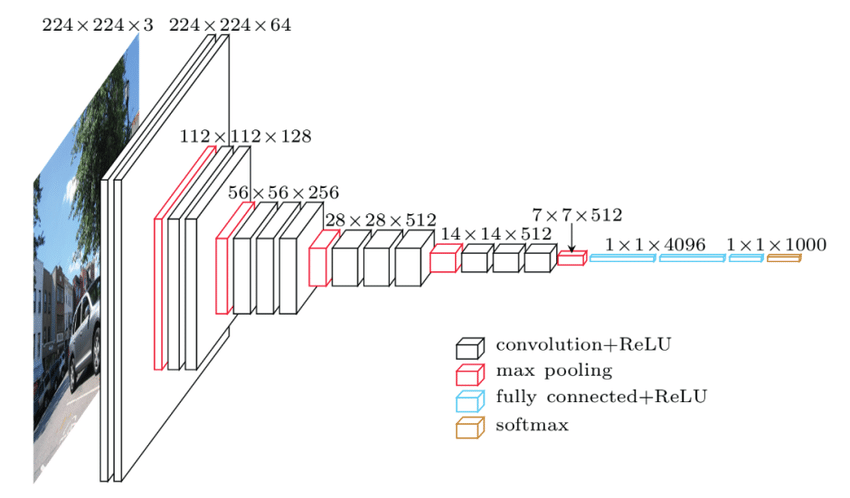
2015------GoogleNet------**6.7%** error rate

2016------RESNET---------**3.5%** error rate

Some most used models

1.VGG16

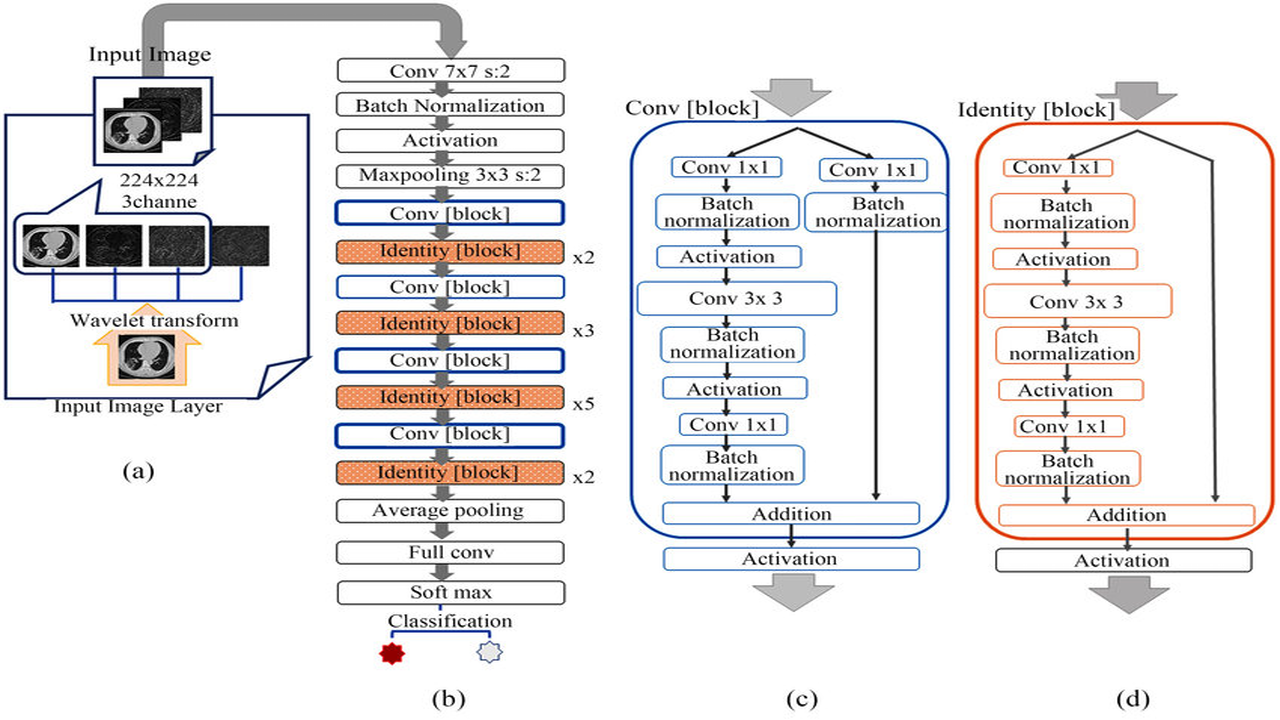
VGG16, as its name suggests, is a 16-layer deep neural network. VGG16 is thus a relatively extensive network with a total of 138 million parameters—it’s huge even by today’s standards. However, the simplicity of the VGGNet16 architecture is its main attraction.



2.RESNET

The original ResNet architecture was ResNet-34, which comprised 34 weighted layers. It provided a novel way to add more convolutional layers to a CNN, without running into the vanishing gradient problem, using the concept of shortcut connections. A shortcut connection “skips over” some layers, converting a regular network to a residual network.

The regular network was based on the VGG neural networks (VGG-16 and VGG-19)—each convolutional network had a 3×3 filter. However, a ResNet has fewer filters and is less complex than a VGGNet. A 34-layer ResNet can achieve a performance of 3.6 billion FLOPs, and a smaller 18-layer ResNet can achieve 1.8 billion FLOPs, which is significantly faster than a VGG-19 Network with 19.6 billion FLOPs



**VGG VS RESNET**

VGG introduced the concept of increasing the number of layers to improve accuracy. However, increasing the number of layers above 20 could prevent the model from converging. The main reason is the vanishing gradient problem—after too many folds, the learning rate is so low that the model’s weights cannot change.

another issue is gradient explosion. A solution is gradient clipping, which involves “clipping” the error derivative to a certain threshold during backward propagation, and using these clipped gradients to update the weights. When the error derivative is rescaled, weights are also rescaled, and this reduces the chance of an overflow or underflow that can lead to gradient explosion.

The Residual Network (ResNet) architecture uses the concept of skip connections, allowing inputs to “skip” some convolutional layers. The result is a significant reduction in training time and improved accuracy. After the model learns a given feature, it won’t attempt to learn it again—instead, it will focus on learning the new features. It’s a clever approach that can significantly improve model training.