**FAMOUS CONVOLUTIONAL NEURAL NETWORK**

**ASSIGNMENT:** 02 **NAME:** SRIRANJANI.S **REG\_NUMBER :**122011173026

**COURSE\_NAME:** APPLIED DEEP LEARNING **COURSE\_CODE:**XAI602C

**ALEX CNN ARCHITECTURE**

**INTRODUCTION**

* AlexNet is an Image Classification model that transformed deep learning. It was introduced by Geoffrey Hinton and his team in 2012, and marked a key event in the history of deep learning, showcasing the strengths of CNN architectures and its vast applications.
* Before AlexNet, people were skeptical about whether deep learning could be applied successfully to very large datasets. However, a team of researchers were driven to prove that Deep Neural Architectures were the future, and succeeded in it; AlexNet exploded the interest in deep learning post-201REFERENCE 2.

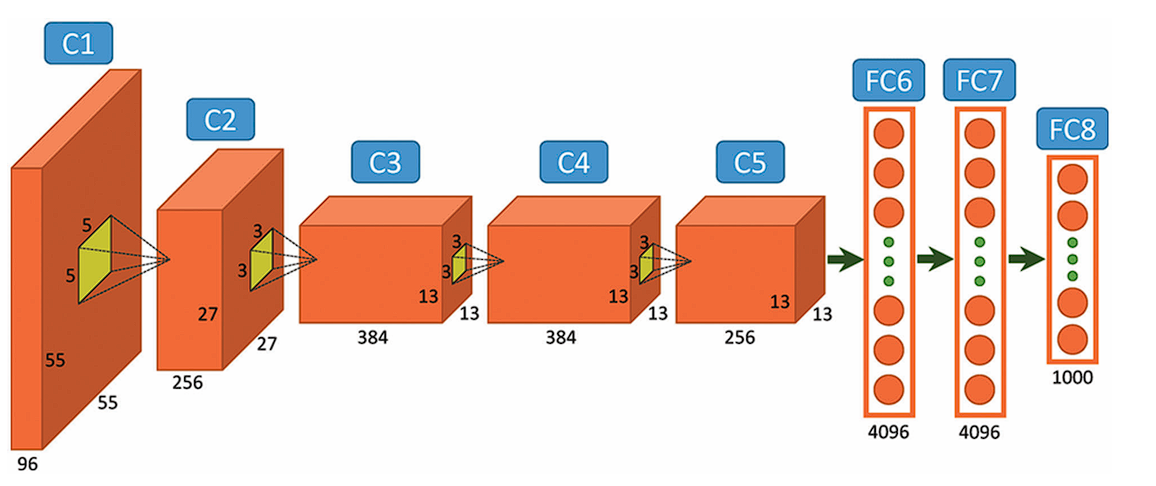
**CONTRIBUTION OF ALEXNET**

The AlexNet paper titled “ImageNet Classification with Deep Convolutional Neural Networks” solved the above-discussed problems.

This paper’s release deeply influenced the trajectory of deep learning. The methods and innovation introduced became a standard for training Deep Neural Networks. Here are the key innovations introduced:

* Deep Architecture: This model utilized deep architecture compared to any NN model released previously. It consisted of five convolutional layers followed by three fully connected layers.
* ReLU Nonlinearity: [CNNs](https://viso.ai/deep-learning/convolutional-neural-networks/) at that time used functions such as Tanh or Sigmoid to process information between layers. These functions slowed down the training. In contrast, ReLU (Rectified Linear Unit) made the entire process simpler and many times faster. It outputs only if the input is given to it as positive, otherwise it outputs a zero.
* Overlapping Pooling: Overlapping pooling is just like regular max pooling layers, but, in overlap pooling, as the window moves across, it overlaps with the previous window. This improved the error percentage in AlexNet.
* Use of GPU: Before AlexNet, NNs were trained on the CPU, which made the process slow. However, the researcher of AlexNet incorporated GPUs, which accelerated computation time significantly. This proved that Deep NNs can be trained feasibly on GPUs.
* Local Response Normalization (LRN): This is a process of [normalizing](https://viso.ai/deep-learning/batch-normalization/) adjacent channels in the network, which normalizes the activity of neurons within a local neighborhood.

**THE ALEXNET ARCHITECTURE**



AlexNet comprises a rather simple architecture compared to the latest Deep Learning Models. It consists of 8 layers: 5 convolutional layers and 3 fully connected layers.

However, it integrates several key innovations of its time, including the ReLU [Activation Functions](https://viso.ai/deep-learning/neuron-activation/), Local Response Normalization (LRN), and Overlapping Max Pooling. We will look at each of them below.

**INPUT LAYER**

AlexNet takes images of the Input size of 227x227x3 RGB Pixels.

**CONVOLUTIONAL LAYERS**

* First Layer: The first layer uses 96 kernels of size 11×11 with a stride of 4, activates them with the ReLU activation function, and then performs a Max Pooling operation.
* Second Layer: The second layer takes the output of the first layer as the input, with 256 kernels of size 5x5x48.
* Third Layer: 384 kernels of size 3x3x256. No pooling or normalization operations are performed on the third, fourth, and fifth layers.
* Fourth Layer: 384 kernels of size 3x3x192.
* Fifth Layer: 256 kernels of size 3x3x192.

**FULLY CONNECTED LAYERS**

The fully connected layers have 4096 neurons each.

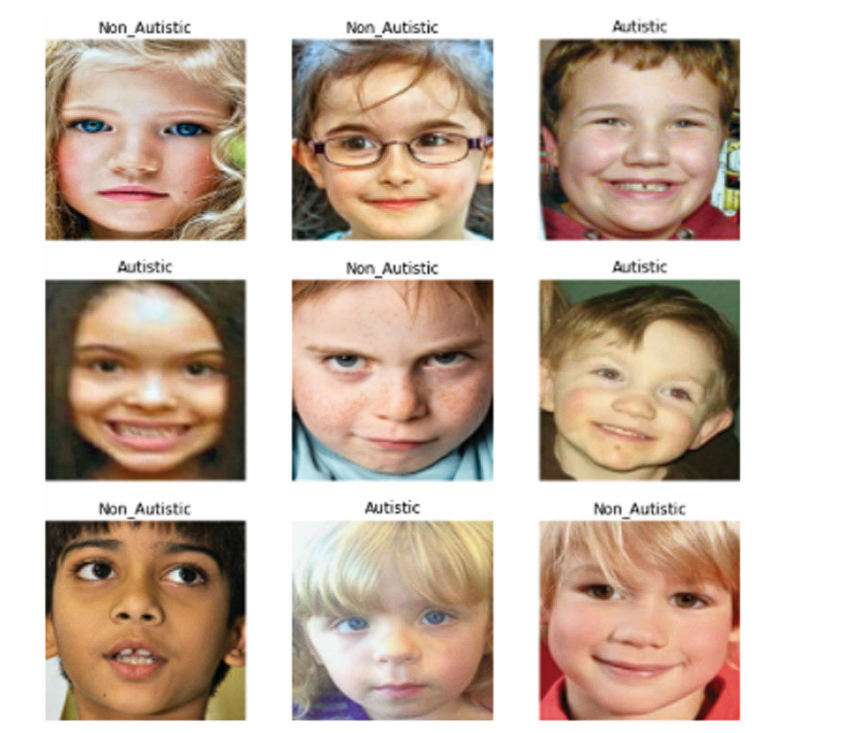
**OUTPUT LAYER**

The output layer is a SoftMax layer that outputs probabilities of the 1000 class labels.

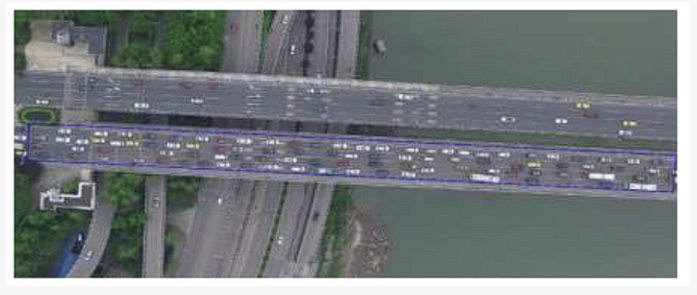
### **APPLICATIONS OF ALEXNET**

Developers created AlexNet for [image classification](https://viso.ai/computer-vision/image-classification/). However, advances in its architecture and transfer learning (a technique where a model trained on one task is repurposed for a novel related task) opened up a new set of possibilities for AlexNet. Moreover, its [convolutional](https://viso.ai/deep-learning/convolution-operations/) layers form the foundation for object detection models such as Fast [R-CNN](https://viso.ai/deep-learning/mask-r-cnn/) and Faster R-CNN, and professionals have utilized them in fields like autonomous driving and [surveillance](https://viso.ai/applications/computer-vision-applications-in-surveillance-and-security/).

* **Autism Detection:** Gazal and their team developed a model using transfer learning, for early detection of autism in children. The model was first trained on ImageNet and then the pre-trained model was further trained on their dataset related to autism.



* **Video Classification:** For video classification, researchers have used AlexNet to extract critical [features](https://viso.ai/deep-learning/feature-extraction-in-python/) in videos for action recognition and event classification.



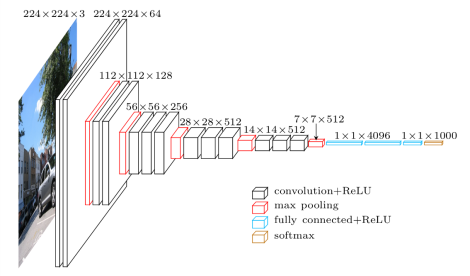
UAV Vehicle Target – [source](https://www.mdpi.com/2071-1050/14/13/7912)

### REFERENCE [AlexNet: A Revolutionary Deep Learning Architecture - viso.ai](https://viso.ai/deep-learning/alexnet/#elementor-toc__heading-anchor-19)

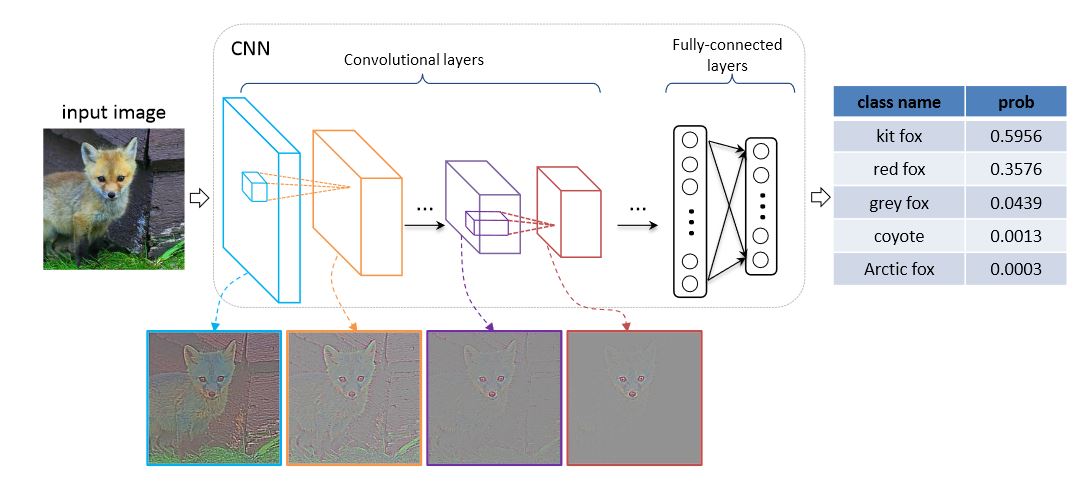
**VGG CNN ARCHITECTURE**

**INTRODUCTION**

VGG stands for Visual Geometry Group; it is a standard deep [Convolutional Neural Network (CNN)](https://viso.ai/deep-learning/convolutional-neural-networks/) architecture with multiple layers. The “deep” refers to the number of layers with VGG-16 or VGG-19 consisting of 16 and 19 convolutional layers.

The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNet also surpasses baselines on many tasks and datasets beyond [ImageNet](https://viso.ai/deep-learning/imagenet/). Moreover, it is now still one of the most popular [image  recognition](https://viso.ai/computer-vision/image-recognition/) architectures.

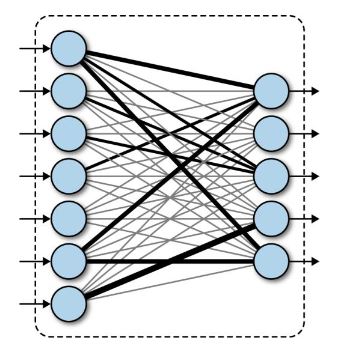
**VGG CONVOLUTIONAL NETWORK ARCHITECTURE**

 VGGNets are based on the most essential features of convolutional neural networks (CNN).The following graphic shows the basic concept of how a CNN works:

The architecture of a Convolutional Neural Network: Image data is the input of the CNN; the model output provides prediction categories for input images. – [Source](https://icmlviz.github.io/icmlviz2016/assets/papers/4.pdf)

The VGG network is constructed with very small convolutional filters. The VGG-16 consists of 13 convolutional layers and three fully connected layers.

Let’s take a brief look at the architecture of VGG:

* Input: The VGGNet takes in an image input size of 224×224. For the ImageNet competition, the creators of the model cropped out the center 224×224 patch in each image to keep the input size of the image consistent.
* Convolutional Layers: VGG’s convolutional layers leverage a minimal receptive field, i.e., 3×3, the smallest possible size that still captures up/down and left/right. Moreover, there are also 1×1 convolution filters acting as a linear transformation of the input. This is followed by a ReLU unit, which is a huge innovation from AlexNet that reduces training time. ReLU stands for rectified linear unit activation function; it is a piecewise linear function that will output the input if positive; otherwise, the output is zero. The convolution stride is fixed at 1 pixel to keep the spatial resolution preserved after convolution (stride is the number of pixel shifts over the input matrix).
* Hidden Layers: All the hidden layers in the VGG network use ReLU. VGG does not usually leverage Local Response Normalization (LRN) as it increases memory consumption and training time. Moreover, it makes no improvements to overall accuracy.
* Fully-Connected Layers: The VGGNet has three fully connected layers. Out of the three layers, the first two have 4096 channels each, and the third has 1000 channels, 1 for each class.

Fully Connected Layers

**COMPLEXITY AND CHALLENGES OF VGG**

The number of filters we can use doubles on every step or through every stack of the convolution layer. This is a major principle used to design the architecture of the VGG16 network. One of the crucial downsides of the VGG16 network is that it is a huge network, which means that it takes more time to train its parameters.

Because of its depth and number of fully connected layers, the VGG16 model is more than 533MB. This makes implementing a VGG network a time-consuming task.The VGG16 model is used in several deep learning image classification problems, but smaller network architectures such as [GoogLeNet](https://viso.ai/deep-learning/googlenet-explained-the-inception-model-that-won-imagenet/) and SqueezeNet are often preferable. In any case, the VGGNet is a great building block for learning purposes as it is straightforward to implement.

**PERFORMANCE OF VGG MODELS**

VGG16 highly surpasses the previous versions of models in the ILSVRC-2012 and ILSVRC-2013 competitions. Moreover, the VGG16 result is competing for the classification task winner (GoogLeNet with 6.7% error) and considerably outperforms the ILSVRC-2013 winning submission Clarifai. It obtained 11.2% with external training data and around 11.7% without it. In terms of the single-net performance, the VGGNet-16 model achieves the best result with about 7.0% test error, thereby surpassing a single GoogLeNet by around 0.9%.

**VGG APPLICATIONS OF COMPUTER VISION AND IMAGE RECOGNITION:**

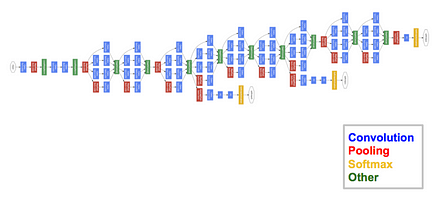
* [TensorFlow Lite](https://viso.ai/edge-ai/tensorflow-lite/): Real-Time Computer Vision on Edge Devices
* [Edge Intelligence](https://viso.ai/edge-ai/edge-intelligence-deep-learning-with-edge-computing/): Deep Learning And Edge Computing
* [AGI Meaning AI](https://viso.ai/deep-learning/artificial-intelligence-types/): What are the Different Types of AI?
* [Automatic Number Plate Recognition](https://viso.ai/computer-vision/automatic-number-plate-recognition-anpr/): Building Real-World Computer Vision Apps
* [OpenPose](https://viso.ai/deep-learning/openpose/): An Open-source Model for [Pose Estimation](https://viso.ai/deep-learning/pose-estimation-ultimate-overview/)

REFERENCE [Very Deep Convolutional Networks (VGG) Essential Guide - viso.ai](https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/)

**GOOGLE NET CNN ARCHITECTURE**

**INTRODUCTION**

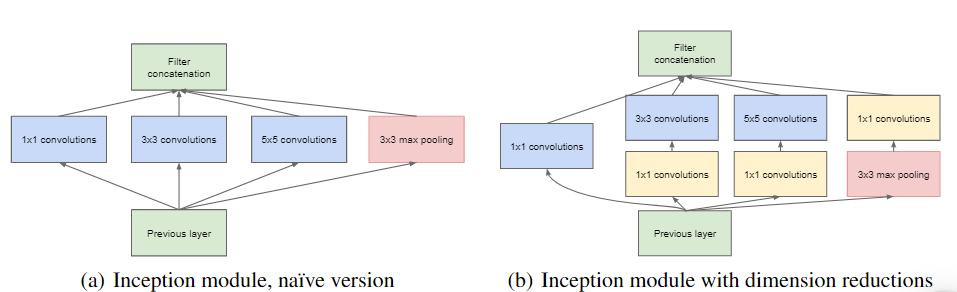
The winner of the ILSVRC 2014 competition was GoogLeNet(a.k.a. Inception V1) from Google. It achieved a top-5 error rate of 6.67%! This was very close to human level performance which the organisers of the challenge were now forced to evaluate. As it turns out, this was actually rather hard to do and required some human training in order to beat GoogLeNets accuracy. After a few days of training, the human expert (Andrej Karpathy) was able to achieve a top-5 error rate of 5.1%(single model) and 3.6%(ensemble). The network used a CNN inspired by LeNet but implemented a novel element which is dubbed an inception module. It used batch normalization, image distortions and RMSprop. This module is based on several very small convolutions in order to drastically reduce the number of parameters. Their architecture consisted of a 22 layer deep CNN but reduced the number of parameters from 60 million (AlexNet) to 4 million.



**BRINGING INCEPTION MODULES INTO THE PICTURE**

**GoogLeNet addressed the challenges of previous CNN architectures by introducing the concept of inception modules**. Inception modules are a type of building block that allows for the parallel processing of data at multiple scales. This allows the network to capture features at different scales more efficiently than previous architectures.

An inception module typically consists of several convolutional layers with different filter sizes. These layers are arranged in parallel, so that the network can process the input data at multiple resolutions simultaneously. The output of the convolutional layers is then concatenated and passed through a pooling layer. However, later there were various versions of the inception module which was integrated accordingly in the architecture which consisted of different layers and filter size patterns.

[](https://media.geeksforgeeks.org/wp-content/uploads/20200429201304/Incepption-module.PNG)

This parallel processing approach has several advantages. First, it allows the network to capture features at different scales more efficiently. This is because the network can process the input data at multiple resolutions simultaneously, which allows it to capture both large-scale and small-scale features. Second, it helps to alleviate the problem of vanishing gradients. This is because the parallel processing approach allows the network to learn features at multiple scales, which can help to stabilise the training process.

The inception module allows the network to capture information at different scales. The inception module is made up of four paths:

* **1x1 convolution:** This path applies a 1x1 convolution to the input. This reduces the number of channels in the input, which helps to reduce the computational complexity of the network.
* **3x3 convolution:**This path applies a 3x3 convolution to the input. This is a standard convolutional operation that is used to extract features from the input image.
* **5x5 convolution:**This path applies a 5x5 convolution to the input. This path is used to capture larger-scale features from the input image.
* **Max pooling:** This path applies a max pooling operation to the input. This operation reduces the size of the input by keeping the maximum value in each 2x2 window.

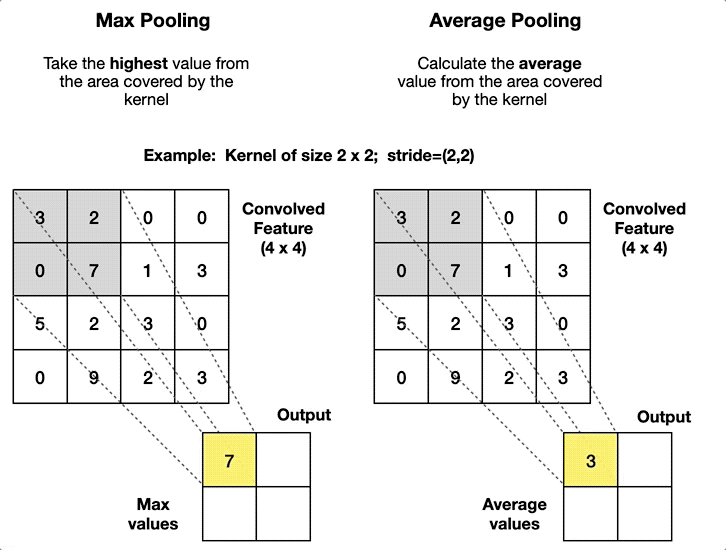
The outputs of the four paths are then concatenated together, and a 1x1 convolution is applied to the result. This final 1x1 convolution helps to reduce the number of channels in the output, and also helps to improve the accuracy of the network.

**Global Average Pooling**

This technique is commonly used in most of the Convolutional Neural Networks to reduce the total number of parameters and to minimise overfitting. Generally it is placed at the end of the CNN architecture.

Global Average Pooling performs an average operation across the Width and Height for each filter channel separately. This reduces the feature map to a vector that is equal to the size of the number of channels. The output vector captures the most prominent features by summarizing the activation of each channel across the entire feature map.

* **Inception Module:**  
  The inception module is different from previous architectures such as AlexNet, ZF-Net. In this architecture, there is a fixed convolution size for each layer.  
  In the Inception module *1×1, 3×3, 5×5* convolution and *3×3* max pooling performed in a parallel way at the input and the output of these are stacked together to generated final output. The idea behind that convolution filters of different sizes will handle objects at multiple scale better.
* **Auxiliary Classifier for Training:**  
  Inception architecture used some intermediate classifier branches in the middle of the architecture, these branches are used during training only. These branches consist of a 5×5 average pooling layer with a stride of 3, a *1×1* convolutions with *128* filters, two fully connected layers of 1024 outputs and 1000 outputs and a softmax classification layer. The generated loss of these layers added to total loss with a weight of 0.3. These layers help in combating gradient vanishing problem and also provide regularization.



**KEY POINTS**

* GoogLeNet introduced inception modules, revolutionising CNN design by enabling parallel processing at multiple scales.
* Its architecture addressed challenges like vanishing gradients and computational efficiency, boosting both accuracy and speed.
* Global average pooling replaced fully connected layers, enhancing accuracy and reducing overfitting.
* Auxiliary classifiers aided gradient flow and regularisation, making training more efficient and preventing over-reliance on specific features.

REFERENCE

[GoogLeNet: A Deep Dive into Google’s Neural Network Technology | by Siddhesh Bangar | Medium](https://medium.com/@siddheshb008/googlenet-a-deep-dive-into-googles-neural-network-technology-f588d1b49e55)