International Journal of Scientific and Technical Research in Engineering (IJSTRE)
www.ijstre.com Volume 10 Issue 1 ǁ Jan-Feb 2025
ISSN: 2581-9941

Computer vision in Deep Learning

Giridhary Kumar Tamuli
Research Scholar, Career Point University Kota
Kota, Rajasthan

Abstract: *Computer vision in Deep Learning is implemented through Convolutional Neural Network (CNN) by building the network from scratch, or through transfer learning and fine tuning. When building from scratch, CNN architecture consists of convolution layers, pooling layers and a fully connected dense neural network for generating the output. During convolutional operation, filters are applied to input image to generate feature representation e.g. as in facial detection and recognition system. The low-level features in facial detection and recognition system consist of edges, dark spots, etc., Mid-level features are eyes, ears, nose, etc., and High-level features are facial structures. Along with convolutional operation, non-linearity is applied often with ReLU (Rectified Linear Unit) activation function which replaces all negative values by zero through pixel-by-pixel operation. Down sampling on each feature maps is done during pooling which reduces dimensionality and preserve invariance. After the convolutional and pooling operations, the output is still two dimensional. Hence, the output is flattened to feed into a dense feed forward fully connected neural network to generate the desired output. However, there is loss of spatial information in a fully connected neural network and all neurons in hidden layer are connected to all neurons in input layer.*

Transfer learning consist of a pretrained CNN architecture previously trained on a large image dataset e.g. VGG 16 network architecture pretrained on ImageNet Dataset. ImageNet dataset consist of 1.4 million images belonging to 1,000 different classes. Feature extraction during transfer learning uses representations learned from previous network to extract features from new samples and then these features are run through a classifier that is freshly trained. Another way of using transfer learning is fine tuning where top few layers of a CNN model is unfrozen and then jointly trained along with a newly added classifier.

Keywords*: Convolutional Neural Network, Transfer Learning.*

I. Introduction

With increased computational speed and proliferation of big data technologies, implementation of deep learning systems became possible with previously never known speed. This also led to development of computer vision technology through convolutional neural network (CNN) or convnet or simply convolutional network. During implementation of convolutional neural network, filters are applied to input image through convolution operation to extract feature map representations. Non linearity through ReLU (Rectified Linear Unit) activation function is than applied to these extracted feature map representations and down sampling is done through pooling operation to reduce the number of parameters. The reduced dimensionality of the extracted feature map representations is flattened and then fed through a densely connected feedforward neural network to generate output according to activation function (sigmoid activation for regression or continuous output / softmax activation for classification output).

Convolutional neural network can be built from scratch from an available dataset or an be built through transfer learning with or without data augmentation. Transfer learning consist of feature map representations learned from previously trained network to extract features from new samples and then run these features through a classifier to generate output e.g. VGG 16 network architecture previously trained on image net dataset (1.4 million images with 1,000 different classes). Various pretrained CNN architectures presently available are LeNet-5, Alexnet, Resnet, Inception Network/ GoogleNet, DenseNet, MobileNet, UNet, etc. Another transfer learning is Fine Tuning which consist of unfreezing top few layers of convolutional model and then jointly train along with nely added classifier.

II. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN), also called are neural network where feature map extraction is done through convolution operation instead of matrix multiplication e.g. image data which use 2D data or time-series data which use 1D data.

2.1 Convolution

Convolution operation is given by

$$s(t) = \int x(a)w(t-a)da.$$

where x and t are real valued with x as input to the convolution, t is time, w(a) is the weighting function and w is the kernel (which should be valid probability density function and should be 0 for all negative arguments). The output is a feature map.

Convolution is denoted by asterisk, hence

$$s(t) = (x * w)(t).$$

and discrete convolution as

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a).$$

For a two-dimensional image I as input and two-dimensional kernel K, convolution is given

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n).$$

Convolution being commutative, the above equation can be equally written as

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n).$$

Flipping the kernel is necessary for commutative property. So, when m increases, the index increases for the input and for the kernel decreases.

Cross-correlation (convolution without flipping the kernel) is given by

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n).$$

Convolutional networks have sparse connectivity due to smaller kernel size. They also share parameters across the network. These properties of Convolution network reduce the memory size and improve statistical efficiency.

Convolutional networks are also equivariance to translation i.e. when the function to the input changes, the output of the function changes in the same way. e.g.

$$f(g(x)) = g(f(x))$$

In 1D time-series data, if in an input, an event is moved later in time, the exact same representation will appear in the output just later. For 2D images, if we move the input feature maps than exactly the same amount of representation will move in the output.

However, Convolution is not naturally equivariant to other transformation like scale and rotation.

Convolutional network also known as convolutional base during transfer learning representation consist of linear activation during the convolution operation at the beginning, followed by non-linear activation through ReLU (Rectified Linear Unit) activation function. Here all the negative values are replaced by 0 and creates invariance. This is followed by pooling which is a down sampling operation which makes the input translation invariant.

2.2 Pooling

The output of the convolutional neural network is replaced by summary statistics from the surrounding. Different pooling operations are
Max Pooling
Average Pooling
L^2 norm in a rectangular neighbourhood, and
Weighted average from the central pixel.

Max pooling output maximum and Average pooling output average in a rectangular neighbourhood.

Since, pooling make feature map representation of the input translation invariant, this help to preserve the location of the feature representation and can locate irrespective of where they are present. A strong prior is added to achieve this property which also improves the statistical efficiency. Convolutional network can also select which feature map will become translation invariant, also in which are parameterized separately.

Also, pooling use smaller units which improve the statistical efficiency and also the memory requirement in storing the parameters of the convolutional network. Regardless of the input size, pooling also output the same summary statistics by varying an offset between the pooling regions.

Pooling can also be done dynamically on interesting feature locations which uses a clustering algorithm. Similarly, single pooling feature can be applied to all images. Pooling only complicates Boltzmann machines and autoencoders because of top-down architecture of these networks.

2.3 Convolutional Neural Network (CNN) or Convnet in Python (Building CNN from scratch)

The Convolutional Neural Network (CNN) or Convnet is a stack of Conv2D and MaxPooling2D layers.

After importing requisite tensorflow and keras library layers and models, input tensors of shape 28x28x1 (height, width, channels) is feed into the Convnet as first layer. This layer contains 32 filters of size 3x3, with ReLU activation function to give non linearity and transform all negative values to 0.

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))

MaxPooling operation is than performed to down sample the output to reduce the spatial dimension and keeping only the important features.

model.add(layers.MaxPooling2D((2, 2)))

The output of Convolution and MaxPooling is of shape(height, width, channels). The height and width tend to shrink as we move deeper into the network.

This model contains three convolution layers and two pooling layers for feature extraction from the input grey scale image.

The output tensor is of shape (3, 3, 64) and the model can be summarized as being obtained after the convolution and pooling operation as follows:

1st Convolution layer has output of shape 26, 26, 32
1st Max Pooling layer has output of shape 13, 13, 32
2nd Convolution layer has output of shape 11, 11, 64
2nd Max Pooling layer has output of shape 5, 5, 64
3rd convolution layer has output of shape 3, 3, 64
Total number of parameters being 55,744 with
Trainable parameters: 55,744 and Non-trainable parameters: 0

The output tensor of shape (3, 3, 64) is first flatten before feeding into a densely connected classifier as the classifier process only 1D vectors. The flattened output obtained is of shape (576,). Then a few dense layers are added on top for binary or multiclass classification. Sigmoid being activation function for binary and softmax being activation function for multiclass classification for generating the output.

The summary of the classifier can contain one flatten layer and number of Dense layers according to the problem statement.

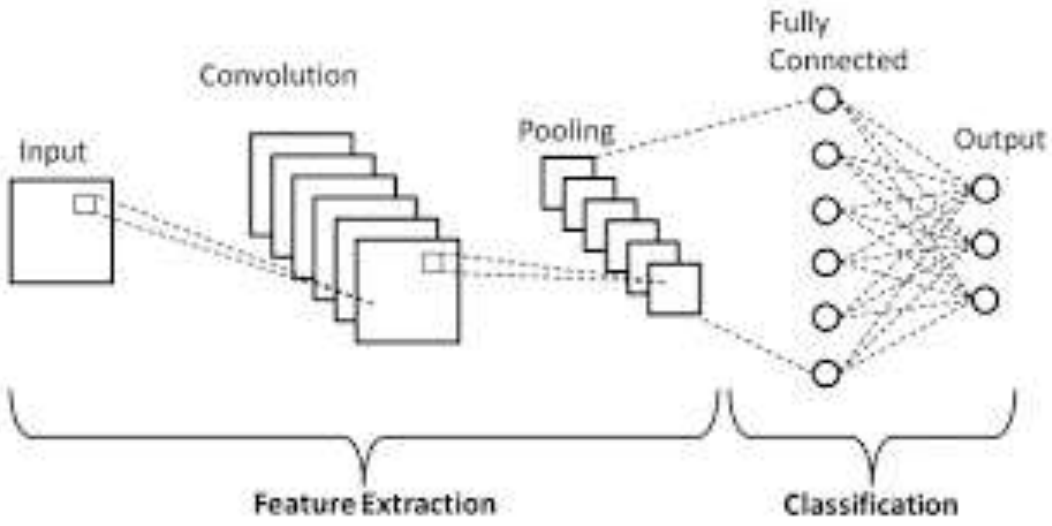

Fig1: CNN Architecture

III. Transfer Learning

Transfer learning is the process of reusing special hierarchy of features learned from a previously pre-trained model for generalizing on another task although the new task is completely different from original task. Generally, the pre-trained network are trained on a large dataset to generalize on a smaller dataset e.g. VGG 16 tre-trained network on ImageNet dataset and generalization is done on limited medical image dataset.

There are two ways of using transfer learning – feature extraction and fine tuning. During feature extraction, representations learned by pre-trained network is used to extract features from new smaller dataset. These features are run through a classifier trained from scratch. The part of CNN or Convnet which consist of the series of convolutional and pooling layers is called convolutional base.

ImageDataGenerator is used to augment and preprocess image data creating generators for training data (with augmentation) and validation data (without augmentation). These generators are then used to train the model using fit method.

Various pretrained CNN architectures presently available are LeNet-5, Alexnet, VGG-16, Resnet, Inception Network/ GoobleNet, DenseNet, MobileNet, UNet, etc.

3.1 LeNet-5

* Proposed in 1998 by Yann LeCun et al in their paper "Gradient based learning applied to document recognition"
* Model was applied for hand written character recognition coming in grayscale centered images (32 x 2 x 1)
* The architecture consists of

Convolutional Layer 1: 6 filters of sizes 5 x 5, Stride 1
Average pooling 1: Size 2 x 2, Stride 2
Convolutional Layer 2: 16 filters of sizes 5 x 5, Stride 1
Average pooling 2: Size 2 x 2, Stride 2

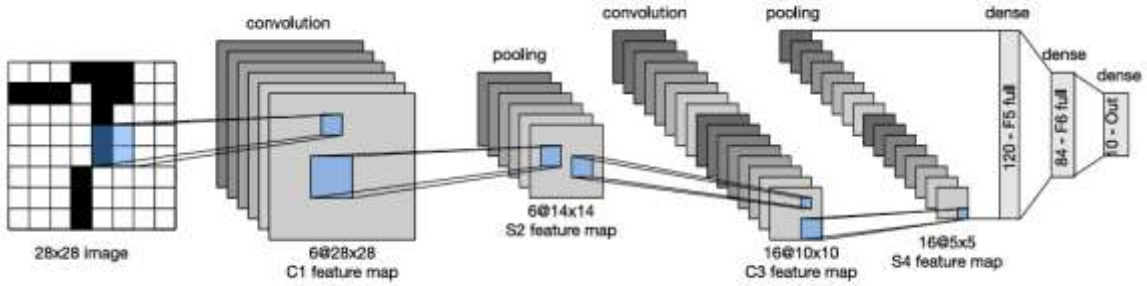

Fig 2: LeNet-5 Architecture

3.2 AlexNet

* Proposed by Alex Krizhevsky et al in 2012 in their paper "Imagenet classification with Deep Convolutional Networks"
* Takes imae of size 227 x 227 x 3 as input.

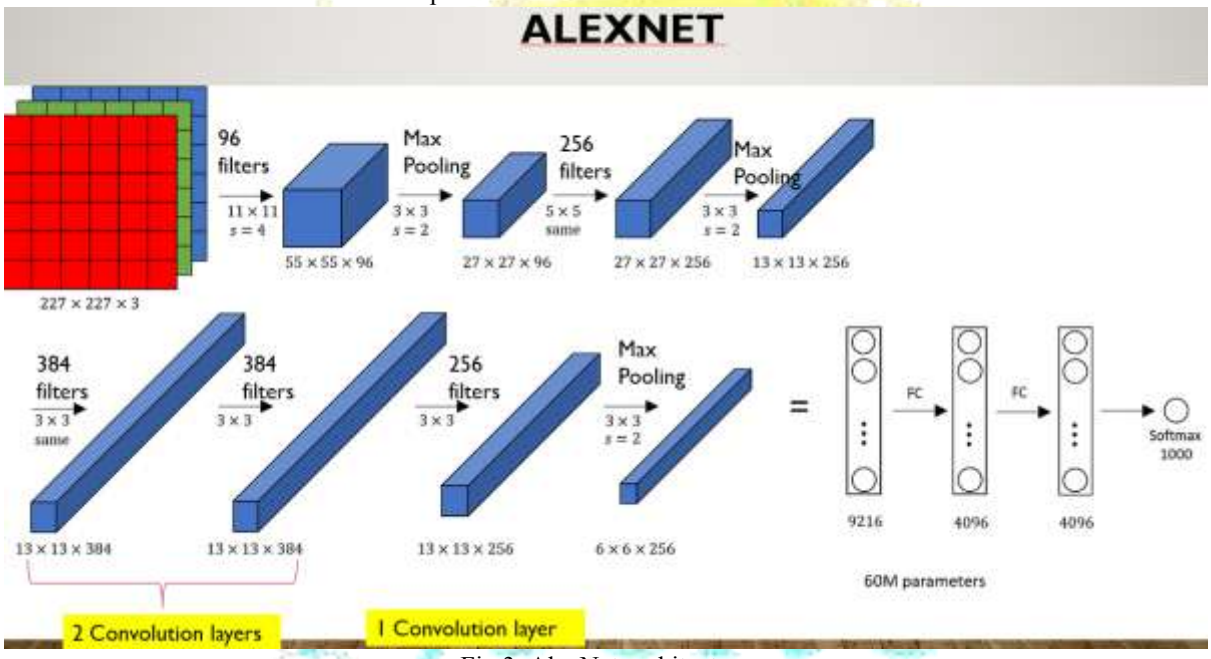

Fig 3: AlexNet architecture

3.3 VGG – 16

* Proposed by Karen Simonyan and Andrew Zissermain in 2015 in their paper "Very deep convolutional networks for large-scale image recognition"
* 13 Convolution layers with 3 x 3 'same' padding and stride as 1, and 5 Max pooling layers with stride 2
* ReLU is used as activation function

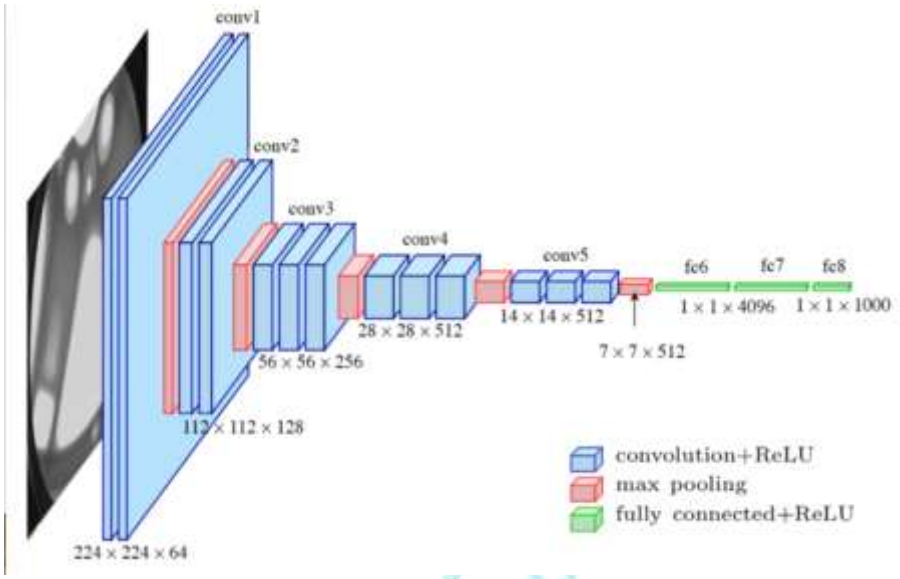

Fig 4: VGG-16 architecture

3.4 ResNet

* Proposed by Kaiming He et al in 2015 in paper "Deep Residual Learning for Image Recognition"
* Very deep network to address vanishing or exploding gradient and degradation of accuracy.
* Similar to feed forward network with 'shortcut connections which perform identity mapping skipping one or more layers
* Different ResNet architecture:

34 layers (using two layers residual block),
50, 101, and 152 layers (using three layers residual block)

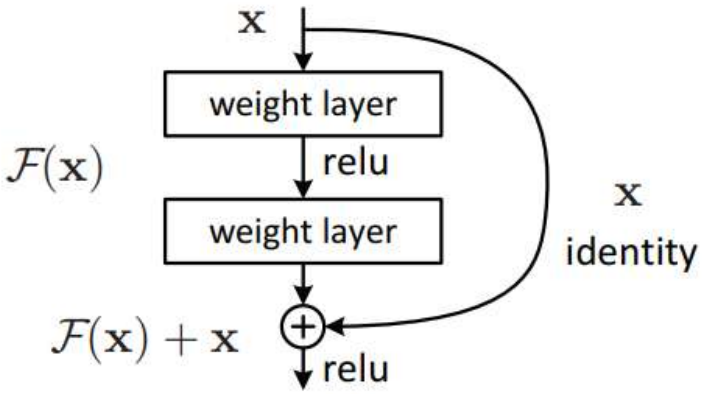

Fig 5: ResNet Block - Residual Networks, or ResNets, learn residual functions with reference to the layer inputs, instead of learning unreferenced functions

3.5 U-Net

* Introduced in the paper "U-Net: Convolutional Networks for Biomedical Image Segmentation" to address the challenge of limited annotated data in the medical field:
* The architecture consists of

Contracting path: each block (three such block) – two convolutional layers and 2 x 2 max pooling layer
Mediate layer: two convolutional layers and no max pooling layer
Expansion path: each block (three such block) – two 3 x 3 convolutional layers followed by 2 x 2 up sampling layer

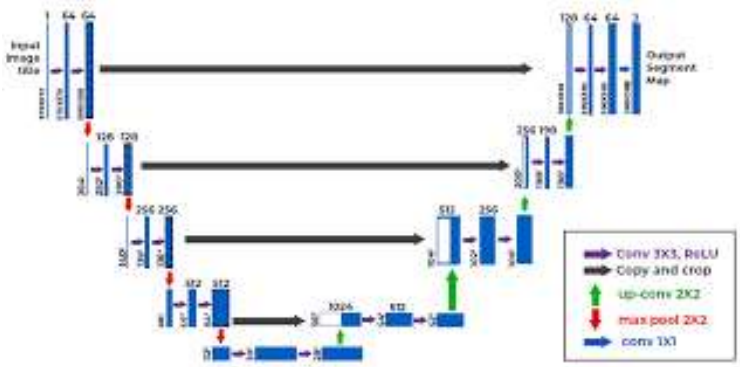

Fig 6: U-Net Architecture

3.6 Inception Network/ GoogleNet

* Proposed by Christian Szegedy in 2015 paper titled "Going Deeper with Convolutions"

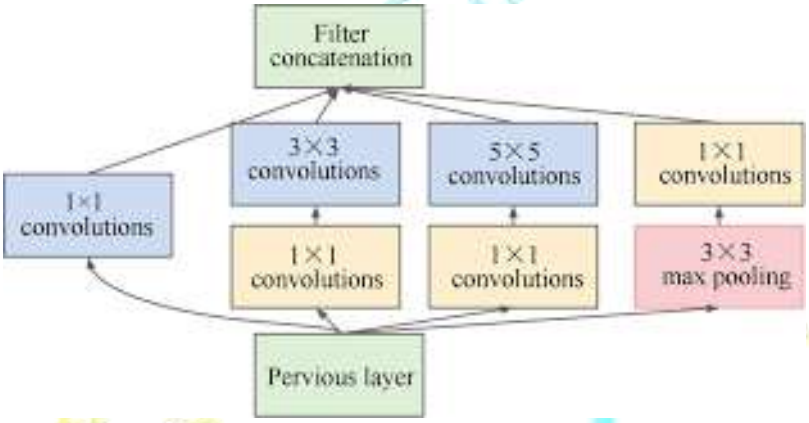

Fig 7: Inception Block

* 1 x 1 convolution is applied before other filters.
* Max pooling: 'same' padding to maintain height and width same as convolutions
* 1 x 1 convolution is applied after max pooling so that output will have same number of channels as the input activation.

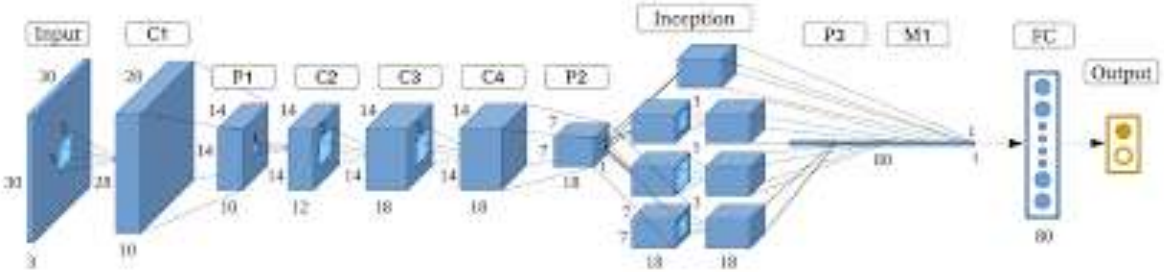

Fig 8: Inception Network/ GoogLeNet Architecture

IV. Conclusions

Convolutional Neural Networks (CNN) or Convnet are utilized for computer vision tasks of regression and classification. It is extensively utilized in object detection, segmentation, medical image diagnosis, accessibility, robotics, self-driving cars (autonomous vehicles), facial recognition systems.

RNNs can be used for generation of captions. Overfitting and underfitting situations can be tried to overcome by using LSTM and GRU. Vision transformers can be used to improve the accuracy metrics of the CNN model.

References and Bibliography

[1]. AstonZhang, ZackC.Lipton, MuLi, AlexJ.Smola, Dive into Deep Learning, 1st Edition; Cambridge University Press, December 7, 2023.
[2]. Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning.; MIT Press, 2016. http://www.deeplearningbook.org
[3]. Michael A. Nielsen, Neural Networks and Deep Learning; Determination Press, 2015.
[4]. Chollet Francois, Deep Learning with Python, First Edition; Manning Publications Co., 2018.
[5]. Amit Kumar Das, Deep Learning; Pearson India Education Services Pvt. Ltd, 2021.

IJSTRE