Methodology project: Convolutional Neural Network

Created by:-

Vedant Patel - 50413479

Rushabh Shah - 50375759

Parth Dixit – 50428131

Github Link: https://github.com/vedant-patel42/CNN_Methodology

Contribution :

Parth Dixit: Literature Review, Abstract, Motivation of using CNN, Feed Forward Network Implementation(Code), Conclusion(Report + Slides).

Vedant Patel: Implementation of CNN model architecture(Code), data preparation, augmentation, and Results, Applications, Future challenges or limitations of CNN (Report + Slides).

Rushabh Shah: Implementation of Random Forest from scratch(Code), CNN structure and methodology description (Report + Slides), Comparison of Random Forest and CNN.

Abstract

Convolution Neural Network (CNN) is a major breakthrough in the field of Deep Learning when we are dealing with classification, detection and recognition problems where a model has to learn features from image data. While, Traditional Neural Network has poor performance and takes too much time to learn millions of pixels, CNN is an extension of NN which uses fewer parameters and gives better performance. This project covers motivation for using CNN,introduction to CNN's and illustrates its implementation by performing image classification on CIFAR-10, a dataset of 60,000 32x32 RGB images. To understand the true potential of CNN, we perform a similar classification using two other well known algorithms, random forest algorithm and Traditional Neural Network on the given dataset. Based on the result we achieved, CNN has far better training and testing accuracy as compared to both random forest algorithm and traditional neural network.

1.History and Literature

There have been many kinds of Convolution Neural Networks developed so far. The first CNN architecture was introduced in 1989 known as "Neocognitron". The architecture used the concept of feature extraction, pooling layers and convolution operation for classification and recognition. 9 years after the introduction of this model, LeNET-5 was developed for handwritten digit recognition tasks with 60000 parameters. In the following years, the models were improved as AlexNet and ZFNet were developed. In this architecture, the Imagenet dataset was used with many convolution and fully connected layers. For the first time, the

Rectified Linear Unit (ReLU) activation function was introduced and 60M parameters were used. In recent years, VGGNet and ResNet were introduced with even more parameters and significantly less error rate.

2.Motivation for using CNN

Image classification is a captivating and rapidly growing area of research in machine learning. Accurately extracting features and classifying images is a prime problem in the field of computer vision. This simple technique of image classification forms a base for extremely complex problems like automated vehicles and so on. To deal with the problem of image classification many tools and methods have been used. Most of these methods are based on a simple mathematical operation called convolution which forms the core of a convolutional neural network. CNN uses the advantages of traditional neural networks and combines it with some novel techniques to create a powerful classification model. A few ways in which traditional neural nets differ from a CNN are:-

1) Sparse Interactions:- Allows a CNN to detect small, meaningful features which leads to fewer parameters unlike neural nets where each unit is connected to every other unit
2) Use of Filters:- To deal with the problem of temporal and positional dependencies CNN makes use of filters to focus on a specific portion of an input.

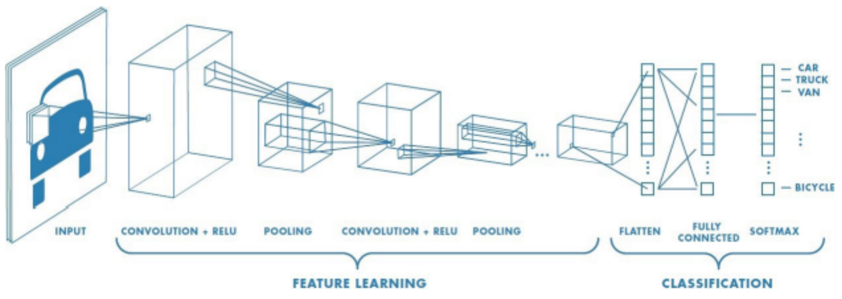


Figure 1: Architecture of Convolution Neural Network.

3. Implementation: Convolution Neural Network

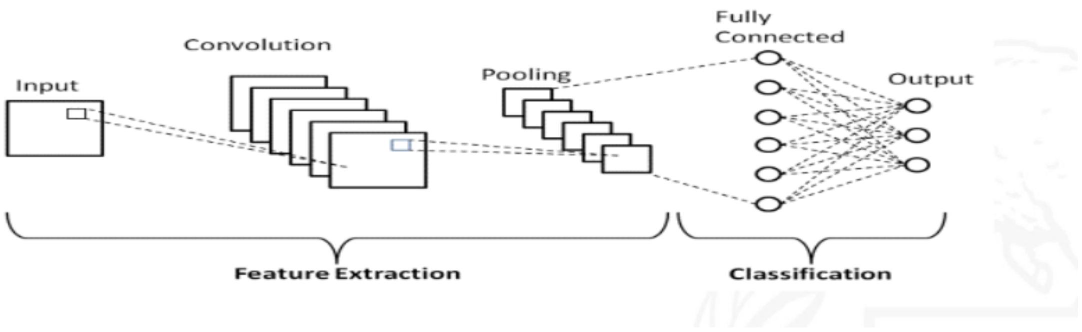


Figure : Structure of CNN

Structure of Convolutional neural consists of 2 parts:

1) **Feature extraction** :- This part performs the core operation of convolution. It furthers contains two layers

   Convolutional layer :- It extracts the features from the image by using learnable filters of size M by M. The filters slide over the input image performing convolutions and creates a feature map using dot product.

   It exploits spatially local correlation by enforcing sparse local connectivity between neurons of adjacent layers. The hyperparameters used in this layer are:-

   1) **Padding -** As we are aware that after each convolutional layer, the input image is reduced in pixels which can reduce the size by around 30 percent which means loss of useful information. To handle this CNN makes use of padding specifically to deal with the loss of pixels on the edge or perimeter of the image. To make sure that the pixels from the corners and edges are included in the computations, extra pixels are added near the boundary of the input image. These pixel values are usually set to zero so that they do not interfere with the computations and skew the results.
   2) **Stride -** Stride refers to the number of rows and columns traversed per slide when the filter moves from one position to another in the input image. For example stride of 1 means the filter shifts by one row and one column.

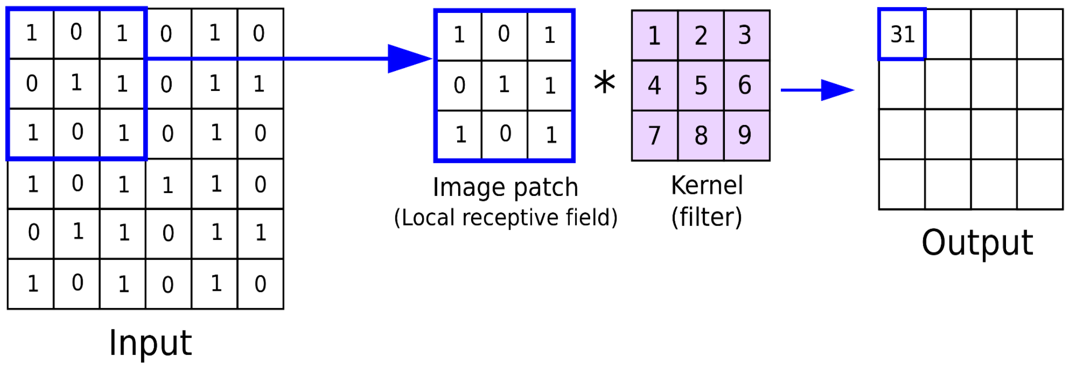
   Pooling layer :- Pooling is a form of non linear downsampling. We can think of pooling as a dimensionality reduction process where the input is reduced in size in order to reduce the computations and make training faster and more efficient. Two major techniques are used to achieve this:-

   Max Pooling:-

   As the name suggests, max pooling selects the maximum value from the grid.

   Average Pooling:-

   It takes the average of all the given values in the grid and outputs them.



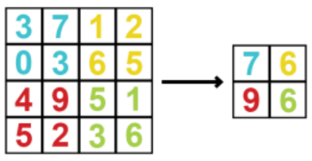


Figure: Example of max pooling

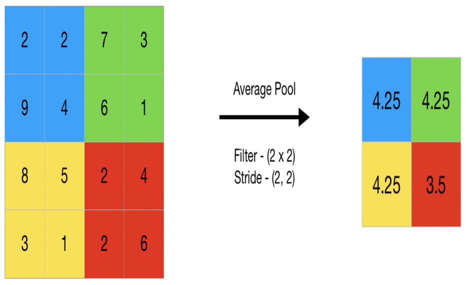


Figure: Example of average pooling

2) **Classification:-** This part handles the classification of images, calculation of loss and its backpropagation to update the weights. It contains a fully connected layer which is nothing but a traditional feed forward network which takes the input and performs non linear activation. Finally, it passeds the output to the output layer where based on the class of problem sigmoid or softmax activation function is used.

3) **Dropout Layer:-** Optionally, a dropout layer can be used which randomly drops certain neurons from the network thereby decreasing the complexity of the model and helps to prevent overfitting

Activation functions

- Used to add non-linearity to the network
- Filters the information that needs to be propagated through the network
- Different types of functions
  1) Sigmoid - used for binary classification and outputs value as 0 or 1
  2) Softmax - used for multiclass classification and outputs a vector of probabilities for each class
  3) Rectified Linear Unit - Outputs the input directly if positive
- Relu is used especially in the convolutional layers(Faster Computation, Gradient Descent does not get stuck)

4. Classification on CIFAR-10

The paper proposed an implementation of CNN model using CIFAR-10 dataset and eventually compared its results with random forest.

4.1 Data Description

CIFAR-10 is a subset of the Tiny Images dataset which contains 80 million 32x32 RGB images. CIFAR-10 includes just 10 object classes, with 6,000 images per class for a total of 60,000 32x32 RGB images. We used 50000 images to train our model and the model is evaluated on the remaining 10000 images. During the training, we kept 20% of training data for validation to further make the generalized model.

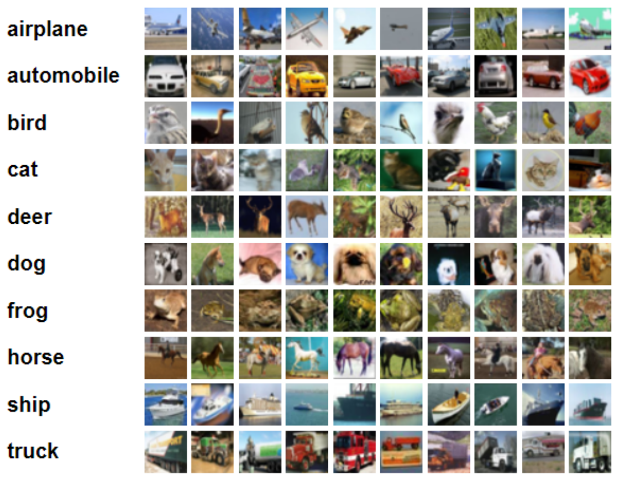
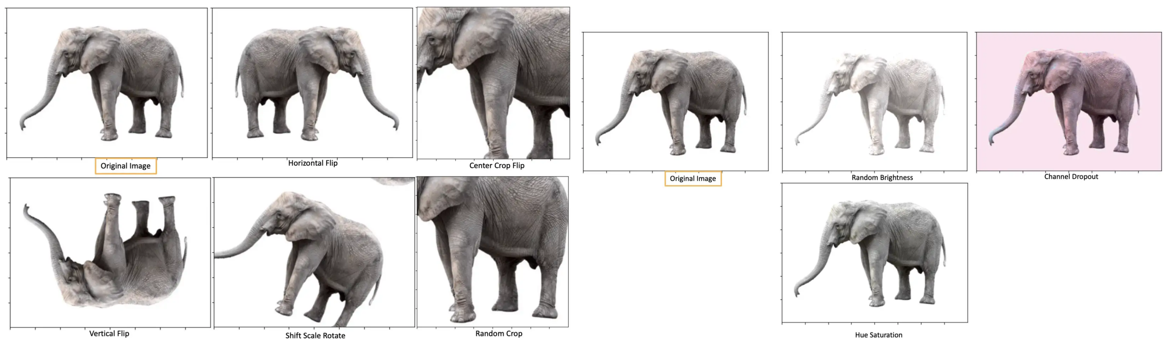


Figure : Training data distribution

The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobiles" includes sedans, SUVs, things of that sort. "Truck" includes only big trucks. Neither includes pickup trucks.

4.2 Pre-processing and augmentation

Pre-processing requires transformation of the raw data into understandable and usable forms**.** As the size of the image is larger, we need to flatten the image and rescale the value. For that image is reshaped in the vector of 32 by 32 and then it is divided by 255. Since 255 is the highest value of pixel all the images will be in the range of [0,1]. By doing so, the number will be smaller and the model computation will become faster. Furthermore, we have converted the training labels into categorical form. By converting labels into categorical data, the values are filled with binary values where 1's present at a specific index represents the class the data belongs to. Since there is such a 60000 data, the training labels are in the form of 2d matrix where number of rows are equal to total samples and columns are equal to total number of target classes.



Finally, we used an augmentation technique to increase the size of our training data by generating artificially new data from existing data points. In our project, we used geometrical transformation (vertical flip) and brightness changes in order to generate more image data. The example is shown in the above figure.

4.3 Model Architecture

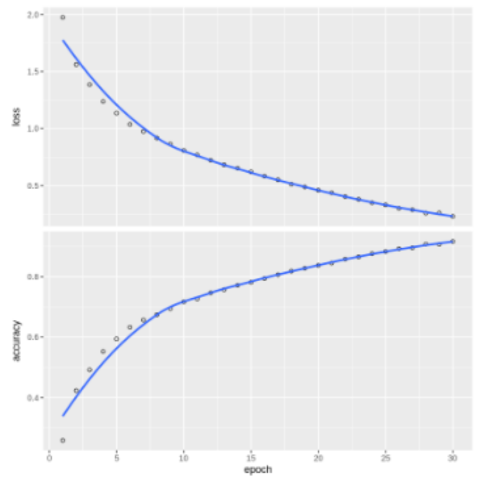
The following figure shows our model architecture. At the beginning, we used 2 convolution layers each with 32 kernels of size (3x3). Convolution layers are used to extract the features. In the following steps, we added max pooling layer and dropout to downsample the features and prevent the overfitting respectively. There are 2 more similar blocks in the following layers. Once the features are learnt, the model is ready to generate the output. Hence, we flattened the layer to convert the 2d matrix to 1d array and added 2 more dense layers to capture more information.

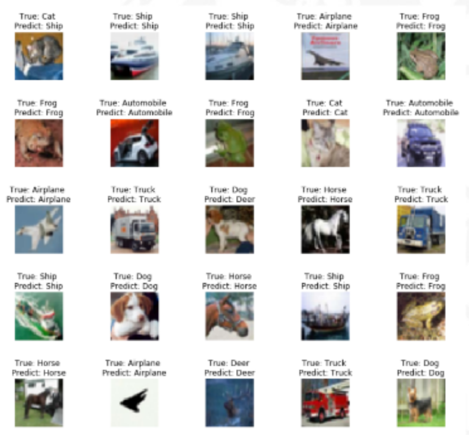
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 32, 32, 32)        896
 conv2d_2 (Conv2D)           (None, 32, 32, 32)        9248
 max_pooling2d_2 (MaxPooling2D)  (None, 16, 16, 32)    0
 dropout_3 (Dropout)         (None, 16, 16, 32)        0
 conv2d_1 (Conv2D)           (None, 16, 16, 64)        18496
 max_pooling2d_1 (MaxPooling2D)  (None, 8, 8, 64)      0
 dropout_2 (Dropout)         (None, 8, 8, 64)          0
 conv2d (Conv2D)             (None, 8, 8, 128)         73856
 max_pooling2d (MaxPooling2D)  (None, 4, 4, 128)       0
 dropout_1 (Dropout)         (None, 4, 4, 128)         0
 flatten (Flatten)           (None, 2048)              0
 dense_1 (Dense)             (None, 1024)              2098176
 dense (Dense)               (None, 512)               524800
 dropout (Dropout)           (None, 512)               0
 Output (Dense)              (None, 10)                5130
=================================================================
Total params: 2,730,602
Trainable params: 2,730,602
Non-trainable params: 0
```

Finally, an output layer is added to generate the classes with softmax activation function. Except for the last layer, we used ReLu as an activation function to bring non linearity into the model. Once the model is built, we computed the loss, training accuracy and evaluated on testing data.

4.4 Result

By fitting the model with epoch 30 and batch size 30 we were able to get the better accuracy, loss and confusion matrix of the model. The loss was 0.2303, while the accuracy recorded was 91.68%.





The test accuracy we got from model evaluation was 80.12% which is far better than random forest. Also the results of CNN are quite impressive as compared to Traditional Neural Network. The accuracy is quite low in a traditional neural network as it only performs normal matrix multiplication operations and hence takes extra memory space and extra time.

5. Applications

Apart from that, there are other applications which came into existence due to the Convolution Neural Network. Facial recognition is one of many which has come through CNN. The model identifies every face in the picture despite having external factors, such as light, angle, pose etc. Analyzing documents becomes easy and quickly with the use of CNN where a machine to be able to scan an individual's writing and compare that to a wide database it has, executes almost a millions commands a minute. The brain tumor is one of the concerning aspects in the medical field when it comes to diagnosing it. It is pretty difficult and tedious to identify the brain tumor manually. So far, Deep Learning is being really helpful to solve those problems easily and quickly and Convolution Neural Network sits on the top when we have to classify and detect tumors accurately and makes medical treatment easy. There are other ample applications such as understanding climate changes,advertising etc.

6. Conclusion and Future challenges

Although there is no other neural network that performs better than Convolution Neural Network for image related tasks, there are some limitations which are to be prevented in near future. For example, "Translation Invariance" which means the object changes its position and orientation and the model will treat it differently and produce different feature values as the position changes. This is because CNNs do not encode the position and orientation of the object. This problem can be reduced by an augmentation technique, however it is impossible to get rid of this problem completely. Another challenge is the use of max pooling layers. Max pooling is useful to capture important features and passes to the next layers, however, it might lose some important information and ignores the relation between the part of the object and the whole object which eventually leads to wrong prediction.

Despite having these limitations, CNN still works well on image related problems such as classification, recognition and detection.

References

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/2961012104553482/4462572393058129/1806228006848429/latest.html

https://rpubs.com/sunartorusli/cifar10

https://indiaai.gov.in/article/top-5-applications-of-convolution-neural-network