

**Illinois Institute of Technology**

**CS-584 Machine Learning**

**Deep Neural Networks - CIFAR10 Classification**

**Final Project Report**

**Submitted By**

**Satwika Sriram (A20563950)**  
**ssriram6@hawk.iit.edu**

**Fnu Anvika(A20556800)**  
**aanvika@hawk.iit.edu**

# 1 Introduction

In the area of computer vision, CIFAR-10 is one of the most used benchmarks for image classification tasks. A difficult yet representative dataset for assessing how well different deep learning and machine learning models perform is provided by CIFAR-10.

Our main goal in this research is to create and train deep neural networks that can correctly categorize photos from the CIFAR-10 dataset into the appropriate groups. With this project, we want to investigate how well various neural network topologies, optimization methods, and regularization schemes perform when it comes to picture categorization. CIFAR-10 is a perfect testbed for evaluating the resilience and generalization abilities of deep learning models because of its small size and high picture quality.

In this introduction, we will provide a brief description of the CIFAR-10 dataset, emphasize its importance in the area of computer vision, and describe how we plan to use deep neural networks to tackle the picture classification job.

Our motivation to push the boundaries of deep learning and benefit the larger scientific community doesn't waver as we work with CIFAR-10 to further explore the field of image categorization. We cordially welcome you to go with us as we explore the nuances of neural network-based image classification and seek to outperform the CIFAR-10 benchmark.

## 2 Problem Description

Our project's main goal is to categorize photos from the CIFAR-10 dataset into one of ten pre-established classes. 60,000 32x32 color pictures that fall into one of the following categories make up CIFAR-10: vehicles, trucks, birds, cats, deer, dogs, frogs, horses, and aircraft. The goal is to create a strong deep neural network model that can correctly classify and recognize these various animals and objects.

Several variables make the CIFAR-10 classification challenge extremely difficult:

**1.Low Resolution Pictures:** Low Resolution pictures: Compared to datasets with greater resolution, the CIFAR-10 dataset's pictures have comparatively low resolution (32x32 pixels), which makes it challenging for models to identify minute details and features.

**2.Class Imbalance:** There may be discrepancies in the distribution of pictures across the various classes in CIFAR-10, which might make training and evaluating the model more difficult, particularly for the classes with less data.

**3.Variability in Object look:** There might be a lot of variation in the look of images in the same class, which makes it difficult to acquire discriminative characteristics that work well in various instances of the same class.

**4.Interclass Similarities:** A model must account for minute distinctions in order to prevent incorrect classifications in some CIFAR-10 classes, such as dogs and cats or trucks and cars.

Our study intends to use optimization methods, regularization techniques, and cutting edge deep neural network topologies to tackle these problems. To increase the model's resilience and performance, we will investigate a number of strategies, such as ensembling, data augmentation, transfer learning, and convolutional neural networks (CNNs).

We want to improve our knowledge of deep learning methods for image identification tasks by addressing the CIFAR-10 classification issue. We also hope to provide insights that may be used in more general computer vision applications, such object detection, picture segmentation, and scene comprehension. We want to create models that demonstrate excellent generalization capabilities across unknown data and real- world circumstances, in addition to achieving high accuracy on the CIFAR-10 benchmark via thorough testing and analysis.

### 3 Dataset Description

In the realm of computer vision, one benchmark dataset that is often used is CIFAR- 10. There are 60,000 32x32 color pictures total, divided into 10 classes of 6,000 pictures each. Ten thousand test photos and fifty thousand training images make up the dataset.

The CIFAR-10 dataset has the following 10 classes:

1. Airplane
2. Automobile
3. Bird,
4. Cat,
5. Deer,
6. Dog,
7. Frog,
8. Horse,
9. Ship,
10. Truck

Every picture in the collection is an RGB image with three color channels (red, green, and blue), measuring 32 by 32 pixels. For every channel, the pixel values are integers between 0 and 255.

The dataset is extensively used for applications including object identification, picture categorization, and machine learning model benchmarking. Due to its tiny size and wide range of classes, it offers a difficult but doable challenge for testing different algorithms and models.

## **4 Literature Survey**

### **Tensorflow**

A free and open-source machine learning software library is called TensorFlow. Although it has many uses, its main emphasis is on deep neural network inference and training. Tensorflow is a symbolic math toolset based on dataflow and differentiable programming. It is used by Google for both production and research. TensorFlow was developed by the Google Brain team for usage inside the company. It was made available under the Apache License 2.0 in 2015. TensorFlow is the name of Google Brain's second-generation system. The release of version 1.0.0 occurred on February 11, 2017. While the standard version only operates on a single device, TensorFlow may run on many CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computation on graphics processing units). TensorFlow may be accessed on 64-bit operating systems such as Linux, macOS, Windows, and iOS and Android mobile platforms.

### **Keras**

Keras is a TensorFlow-based deep learning API written in Python. It was designed to make it as easy as possible for consumers to explore. Research success requires the ability to move as fast as possible from idea to result. A variety of implementations of fundamental neural-network building blocks, including as layers, goals, activation functions, optimizers, and other tools, are included in Keras to facilitate the handling of image and text input and to minimize the amount of coding needed to construct deep neural network code. Community help is available via a Slack channel and the code is posted on GitHub. In addition to standard neural networks, Keras also supports convolutional and recurrent neural networks. Other widely used utility layers that are supported include batch normalization, pooling, and dropout.

### **Convolutional Neural Networks (CNN)**

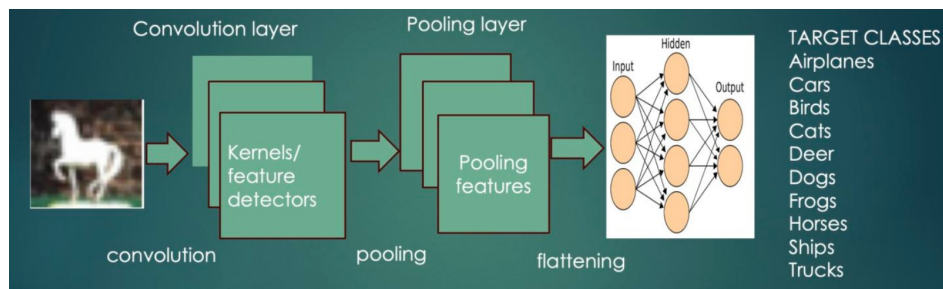
Deep Learning's ability to handle enormous amounts of data has made it an especially helpful method in recent years. The popularity of hidden layers has surpassed that of older methods, especially in pattern recognition. One of the most popular types of deep neural networks is the convolutional neural network. Since AI was still in its infancy in the 1950s, researchers have worked to develop a system. that has the ability to understand visual data Over time,

this field's popularity increased. The name given to this kind of technology is computer vision. In the field of computer vision, the 2012 breakthrough was a quantum leap. An AI model developed by a team of academics at the University of Toronto significantly surpassed the top image recognition systems. The artificial intelligence system was called AlexNet, after its creator. In the 2012 ImageNet computer vision competition, Alex Krizhevsky won. It's amazing that 85 percent of the time is accurate. On the exam, the runner-up scored a creditable 74 percent. AlexNet was built on convolutional neural networks, a special kind of neural network. a neural network that attempts to closely mimic human vision. CNNs have changed over time. It's currently a vital component of many Computer Vision applications.

## Working of CNN Algorithm

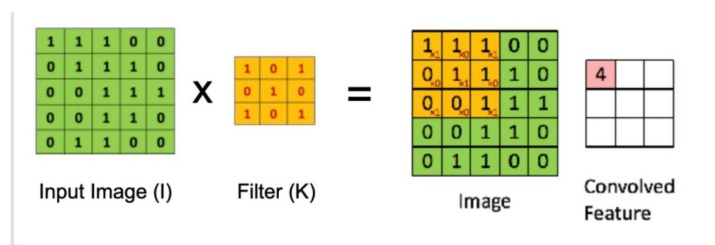
Convolutional neural networks feature several layers, which the input picture must travel through in order to get the intended result. Included are the following layers:

1. Convolutional layer
2. Pooling layer
3. Fully connected layer



### 1. The Convolutional Layer:

Here is where the convolution process is carried out. Filter or kernel: An array of weights learned by backpropagation that is used to extract features and edges from an input picture. Feature Map: the result of convolving a filter over an input picture.



### Convolution operation:

To create a feature map, this method entails convolving a specified filter over an input picture.

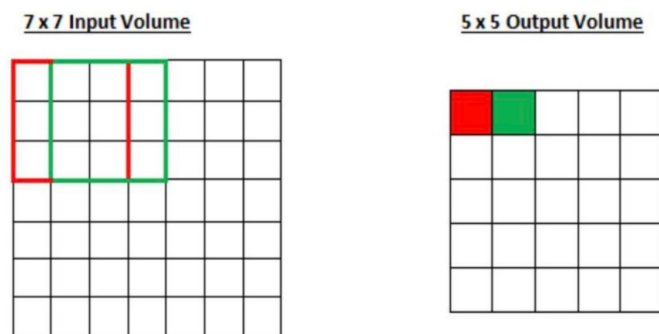
## Padding:

By adding a second layer around the edge of our original image, we preserve its details without reducing them. Traditionally, we pad using zeros. It's a hyperparameter that requires adjustment. Some of the reasons we use padding are as follows:

1. Our input image will shrink and lose its original size until it reaches the last stage since the convolution technique is repeated.
2. Information is lost from the corners when a filter is applied to an input picture because the top left corner appears just once while pixels in the center appear several times.

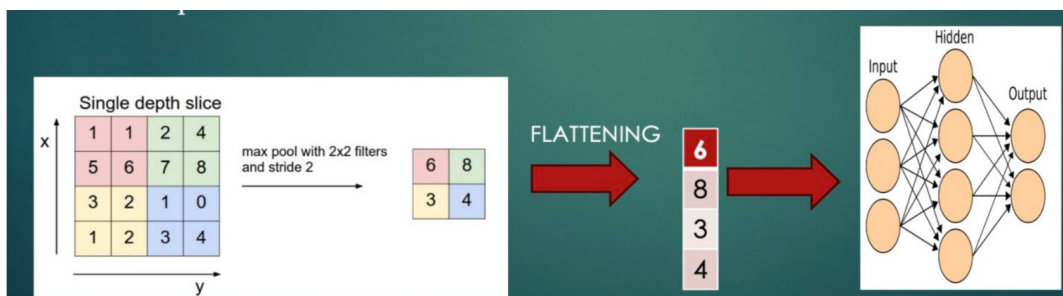
## Stride:

The stride is the amount of movement the kernel makes as it advances over the image. Stride is a hyperparameter in this context that describes the step of the convolution procedure.



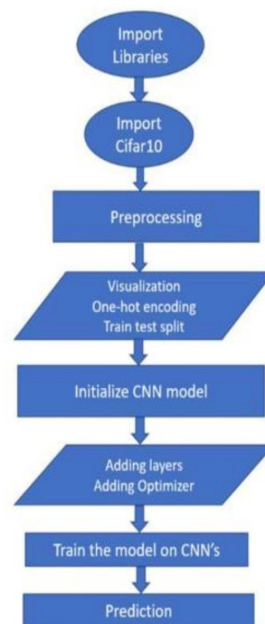
## 2. The Pooling Layer :

ConvNets usually use pooling layers in addition to convolutional layers to reduce the size of the picture, speed up computation, and somewhat strengthen some of the detecting properties. Although there are many other types of pooling, these two are the most well-known: There are two kinds of pooling: average pooling and maximum pooling.

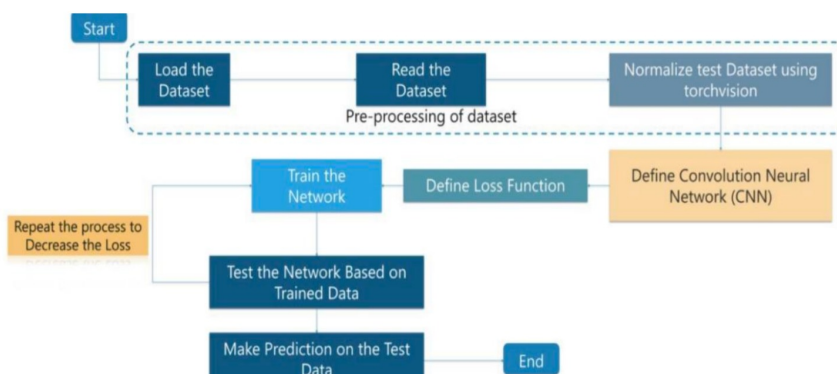


### 3. Fully Connected Layer/ Dense Layer :

In order to generate a prediction, the dense layer—a fully connected layer—feeds the convolutional layers' outputs via one or more neural layers. When a layer is fully connected, every element from every feature in the previous layer is computed for every element of every output feature, which also includes an activation function based on our business requirements. Between the convolutional and fully connected layers comes a layer known as Flatten. A fully connected neural network classifier may receive a vector that is created by flattening a two-dimensional matrix of attributes.



**System Architecture**



**Dataflow Diagram**

## 5 SYSTEM MODULES AND ACCOMPLISHMENTS

The modules in this project comprise of:

1. Importing libraries
2. Importing Dataset
3. Data Visualization
4. Data Preparation
5. Training the model
6. Data Augmentation
7. Model Evaluation

### 5.1 Importing the Libraries

Importing the libraries as required.



```
import pandas as pd # data manipulation
import numpy as np #
import matplotlib.pyplot as plt #data plotting
import seaborn #for data visulization
import tensorflow as tf
from tensorflow.keras.layers import Input, Conv2D, Dense, Flatten, Dropout
from tensorflow.keras.layers import GlobalMaxPooling2D, MaxPooling2D
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.models import Model
```

### 5.2 Importing the Dataset

In this step, we will feed the cifar10 data into our model and separate it into train and test sets. We now have 50000 photos in the train set and 10,000 images in the test set out of our total of 60000 shots. The collection's photographs have dimensions of (32,32,3) since they are colored.



```
from keras.datasets import cifar10
(X_train,y_train),(X_test,y_test) = cifar10.load_data()

Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [=====] - 3s 0us/step

[ ] X_train.shape
(50000, 32, 32, 3)

[ ] X_test.shape
(10000, 32, 32, 3)

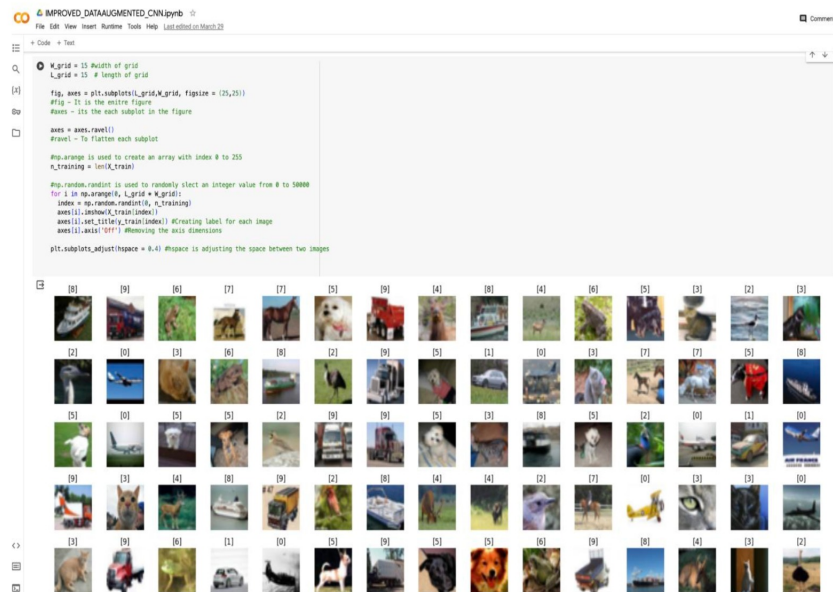
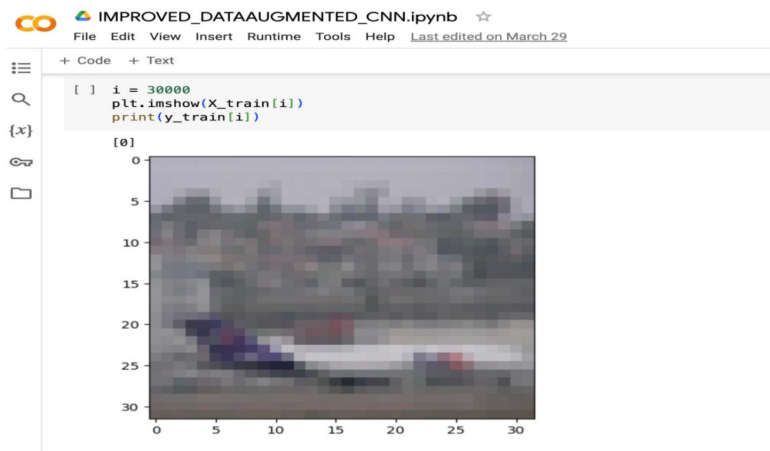
[ ] y_train.shape
(50000, 1)

[ ] y_test.shape
(10000, 1)
```



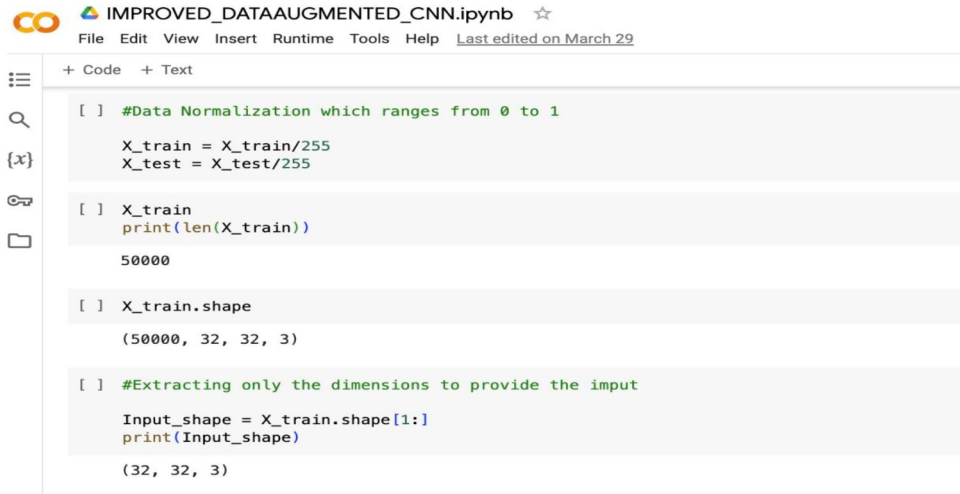
### 5.3 Data Visualization

Information and data are represented graphically in data visualization. Visualizations are an effective tool for conveying complex concepts, patterns, and trends in an understandable and straightforward manner. We may examine data, spot trends and outliers, and share our conclusions with others by utilizing visualizations. With the CIFAR 10 dataset, we created a few visualizations. In data analysis, visualizations are a crucial tool that may reveal insights that would otherwise be hard to perceive. They may also be used to disseminate research results to a larger audience, increasing the accessibility and interest level of data. Effective visualization skills are becoming more and more crucial due to the daily generation of ever-increasing amounts of data.



## 5.4 Data Preparation

Hot encoding is one method of altering data to make it more predictive and ready for an algorithm. Each category value is transformed into a new categorical column and assigned a binary value of 1 or 0 using one-hot. Every integer value is represented as a binary vector.



```
IMPROVED_DATAAUGMENTED_CNN.ipynb ☆
File Edit View Insert Runtime Tools Help Last edited on March 29

+ Code + Text

[ ] #Data Normalization which ranges from 0 to 1
    X_train = X_train/255
    X_test = X_test/255

[ ] X_train
    print(len(X_train))

    50000

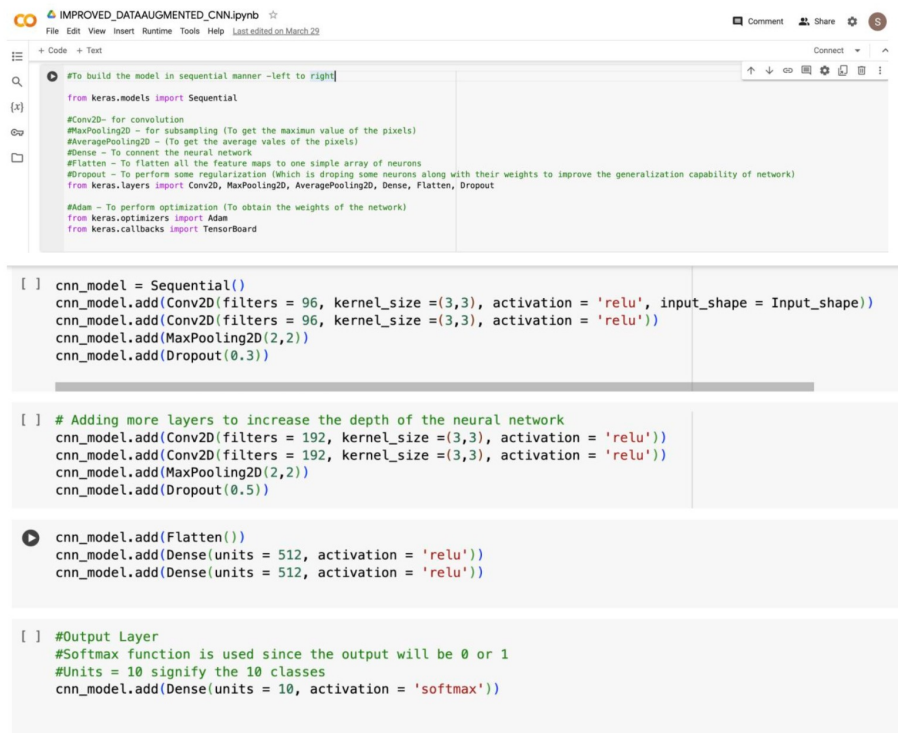
[ ] X_train.shape

    (50000, 32, 32, 3)

[ ] #Extracting only the dimensions to provide the input
    Input_shape = X_train.shape[1:]
    print(Input_shape)

    (32, 32, 3)
```

## 5.5 Training the model



```
IMPROVED_DATAAUGMENTED_CNN.ipynb ☆
File Edit View Insert Runtime Tools Help Last edited on March 29

+ Code + Text

#To build the model in sequential manner -left to right
from keras.models import Sequential

#Conv2D- for convolution
#MaxPooling2D - for subsampling (To get the maximum value of the pixels)
#AveragePooling2D - (To get the average vales of the pixels)
#Dense - To connect the neural network
#Flatten - To Flatten all the feature maps to one single array of neurons
#Dropout - To perform some regularization (Which is dropping some neurons along with their weights to improve the generalization capability of network)
from keras.layers import Conv2D, MaxPooling2D, AveragePooling2D, Dense, Flatten, Dropout

#Adam - To perform optimization (To obtain the weights of the network)
from keras.optimizers import Adam
from keras.callbacks import TensorBoard

[ ] cnn_model = Sequential()
    cnn_model.add(Conv2D(filters = 96, kernel_size =(3,3), activation = 'relu', input_shape = Input_shape))
    cnn_model.add(Conv2D(filters = 96, kernel_size =(3,3), activation = 'relu'))
    cnn_model.add(MaxPooling2D(2,2))
    cnn_model.add(Dropout(0.3))

[ ] # Adding more layers to increase the depth of the neural network
    cnn_model.add(Conv2D(filters = 192, kernel_size =(3,3), activation = 'relu'))
    cnn_model.add(Conv2D(filters = 192, kernel_size =(3,3), activation = 'relu'))
    cnn_model.add(MaxPooling2D(2,2))
    cnn_model.add(Dropout(0.5))

[ ] cnn_model.add(Flatten())
    cnn_model.add(Dense(units = 512, activation = 'relu'))
    cnn_model.add(Dense(units = 512, activation = 'relu'))

[ ] #Output Layer
    #Softmax function is used since the output will be 0 or 1
    #Units = 10 signify the 10 classes
    cnn_model.add(Dense(units = 10, activation = 'softmax'))
```

## 5.6 Data Augmentation

A key deep learning method is data augmentation, especially for picture classification problems with small datasets. By generating altered copies of already-existing pictures, it artificially increases the training data, hence improving the resilience and generalisation capabilities of the model. CIFAR-10, which has only 50,000 training photos, would benefit particularly from this.

### Benefits of Data Augmentation:

**Reduces Overfitting:** Better performance on unseen data results from data augmentation preventing the algorithm from retaining certain characteristics of training pictures by adding variances.

**Increases Generalization:** Better performance on real-world data is the outcome of the model learning to identify objects under different circumstances (viewpoint, illumination, size, location).

### Standard Methods of Data Augmentation for CIFAR-10:

**Random Cropping:** Through the extraction of smaller sub-regions from the original picture, random cropping reduces sensitivity to object position within the image and forces the model to concentrate on informative portions.

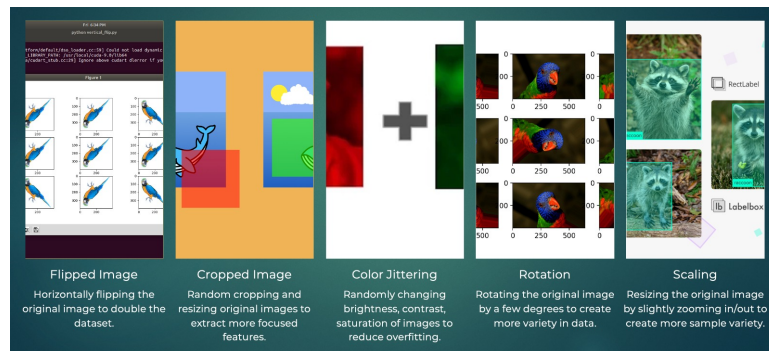
**Random Flipping:** The dataset is expanded and the model is made invariant to object orientation (e.g., automobile facing left vs. right) by random flipping (horizontal/vertical) of the photos.

**Random Rotation:** Random Rotation simulates actual camera orientation fluctuations by rotating the pictures by tiny random angles (e.g., slanted image).

**Random Zooming:** Zooming in or out randomly on the picture introduces scale changes and lessens the model's sensitivity to item size.

**Random Brightness:** Random Brightness/Contrast Adjustment simulates various lighting situations by varying the brightness and contrast of the pictures (e.g., brighter vs. darker images).

**Random Color Jittering:** Boosts picture resilience to color fluctuations (e.g., little color shifts) by slightly perturbing the RGB color channels.

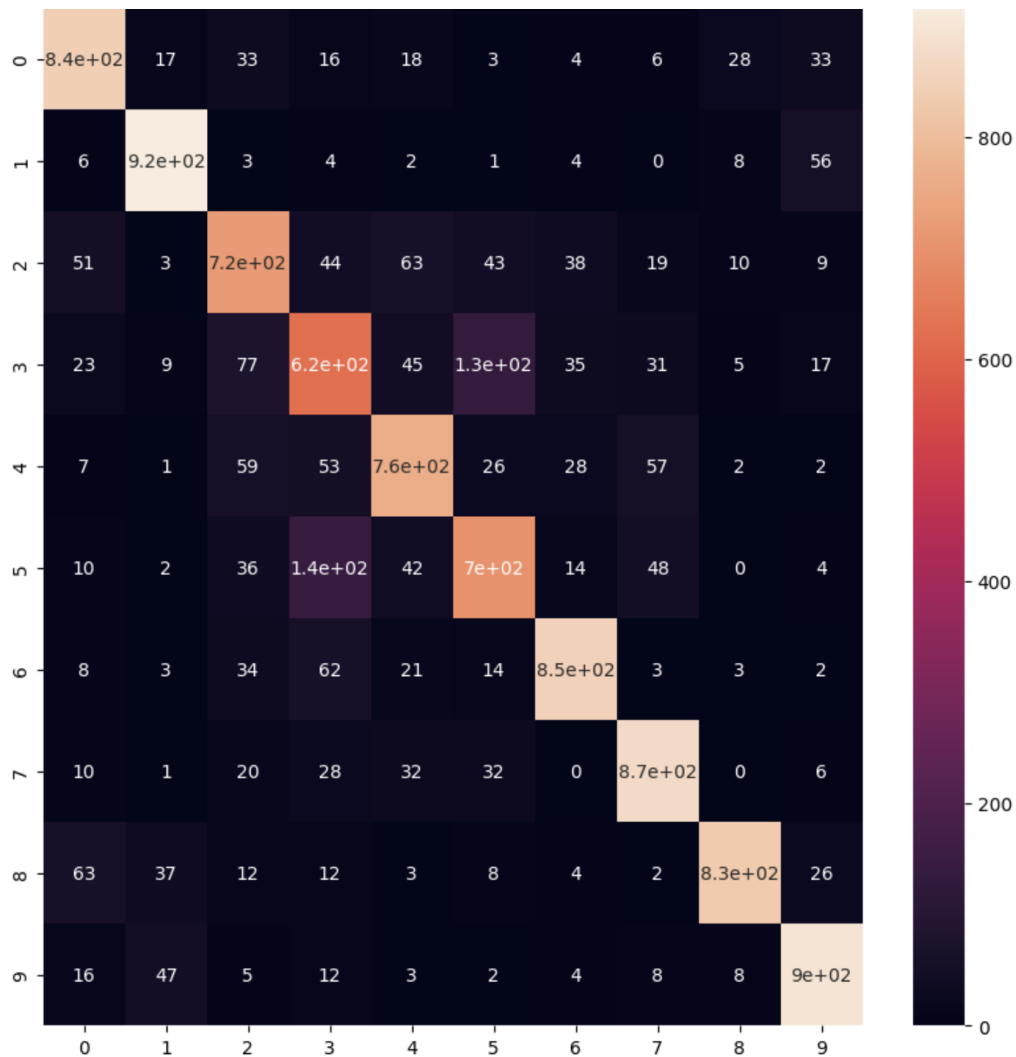


## 5.7 Model Evaluation

Model assessment is the procedure of gauging the effectiveness of a machine learning model on unseen data. This entails analyzing different criteria that reveal the model's capability to forecast the target variable accurately. The objective of model assessment is to ascertain the model's overall performance in making predictions on new data and to pinpoint the most suitable algorithm or model for forecasting test data.

### CONFUSION MATRIX AND HEAT MAP :

A confusion matrix is a table that compares the predicted labels of a piece of data with the actual labels in order to assess how well a machine learning algorithm is doing. It is also referred to as a contingency table or an error matrix. A machine learning model's efficacy may be assessed using a variety of performance measures, including accuracy, precision, recall, and F1 score, which can be computed by examining the values in the confusion matrix.



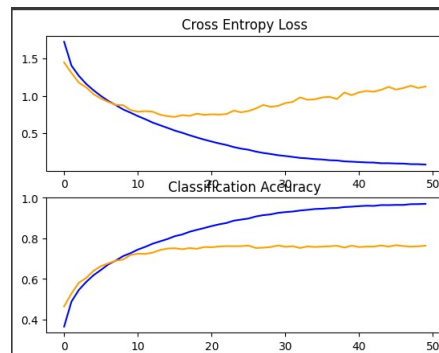
Simple CNN Model gave an accuracy of 76 percent. This can be seen in the figure below.

```

▶ evaluation = cnn_model.evaluate( X_test, y_test, batch_size = 1)
#evaluation = cnn_model.evaluate(X_test, y_test)
print('Test accuracy: {}'.format(evaluation[1]))

```

10000/10000 [=====] - 42s 4ms/step - loss: 1.1269 - accuracy: 0.7643  
 Test accuracy: 0.7642999887466431



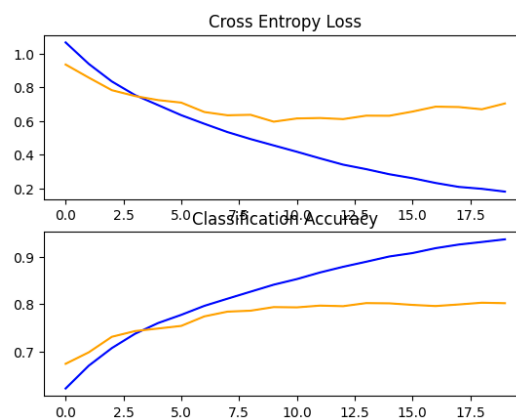
Improved CNN Model gave an accuracy of 80 percent. This can be seen in the figure below.

```

evaluation = cnn_model.evaluate( X_test, y_test, batch_size = 1)
#evaluation = cnn_model.evaluate(X_test, y_test)
print('Test accuracy: {}'.format(evaluation[1]))

```

10000/10000 [=====] - 101s 10ms/step - loss: 0.7036 - accuracy: 0.8016  
 Test accuracy: 0.8015999794006348



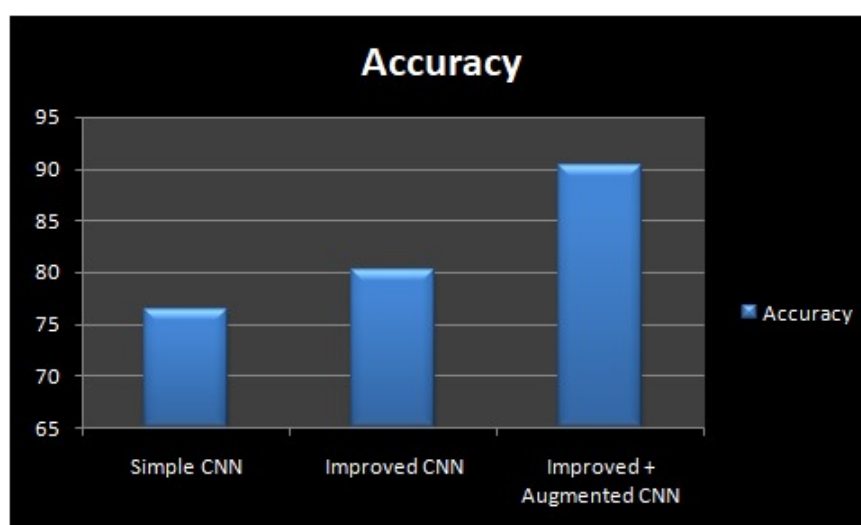
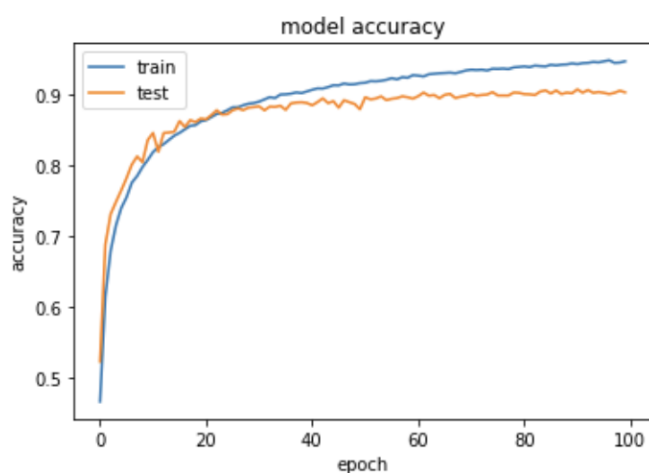
Improved and data augmented CNN Model gave an accuracy of 90 percent. This can be seen in the figure below.

```
from keras.utils import to_categorical

# Assuming y_test is a list of integer labels
y_test_encoded = to_categorical(y_test)

# Now, evaluate the model
score = cnn_model.evaluate(X_test, y_test_encoded, batch_size=1)
print(score)
```

10000/10000 [=====] - 4s 436us/step  
[0.34665883693695071, 0.90259999999999996]



Comparison of the three models

## 6 Conclusion

In this study, we explored the effectiveness of Convolutional Neural Networks (CNNs) for a classification task. We experimented with three variations of CNN models: a simple CNN, an improved CNN with additional layers and regularization techniques, and a data-augmented CNN. Our goal was to assess the impact of model complexity and data augmentation on classification accuracy.

The simple CNN model provided a baseline performance, achieving an accuracy of 76 percent on the test dataset. Through architectural improvements such as adding more convolutional and pooling layers, along with dropout regularization, we observed a noticeable enhancement in performance. The improved CNN model achieved an accuracy of 80 percent, representing a 4 percent increase over the simple CNN.

Furthermore, we investigated the benefits of data augmentation techniques such as rotation, flipping, and zooming to expand the training dataset. The data-augmented CNN model demonstrated robustness to variations in input data, resulting in a significant boost in accuracy. The model attained an accuracy of 90 percent, marking a 10 percent improvement over the improved CNN model.

Overall, our findings suggest that both architectural enhancements and data augmentation play crucial roles in improving CNN performance for classification tasks. The results highlight the importance of model complexity and data diversity in achieving higher accuracy rates. Future work could explore additional optimization strategies and alternative augmentation techniques to further enhance model performance.

## 7 References

1. Doon, R., Kumar Rawat, T., Gautam, S. (2018). Cifar-10 Classification using Deep Convolutional Neural Network. In 2018 IEEE Punecon. 2018 IEEE Punecon. IEEE.  
<https://doi.org/10.1109/punecon.2018.8745428>
2. Aslam, S., Nassif, A. B. (2023). Deep learning based CIFAR-10 classification. In 2023 Advances in Science and Engineering Technology International Conferences (ASET).  
<https://doi.org/10.1109/aset56582.2023.10180767>
3. Vinay, S. B., Balasubramanian, S. (n.d.). A COMPARATIVE STUDY OF CONVOLUTIONAL NEURAL NETWORKS AND CYBERNETIC APPROACHES ON CIFAR-10 DATASET. In International Journal of Machine Learning and Cybernetics (IJMLC) (Vol. 1, Issue 1). <https://iaeme.com>

4. Emonds, Y., Xi, K., Fröning, H. (2024). Implications of Noise in Resistive Memory on Deep Neural Networks for Image Classification. <http://arxiv.org/abs/2401.05820>
5. Seetharamarao, V., Deepak, S. N., Srihari, N., Kumar, S. (n.d.). IMAGE CLASSIFICATION OF CIFAR10 USING CNN.[www.irjmets.com](http://www.irjmets.com)
- 6.<https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/>
- 7.<https://www.geeksforgeeks.org/cifar-10-image-classification-in-tensorflow/>
8. . Divya, B. Adepu and P. Kamakshi, "Image Enhancement and Classification of CIFAR-10 Using Convolutional Neural Networks,"2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2022, pp. 1-7, <http://doi.org/10.1109/ICSSIT53264.2022.9716555>.
- 9.<https://medium.com/@jayramchaudhury20/project-on-image-classification-on-cifar-10-dataset-94db0ff6baf5>
- 10.R. C. Çalik and M. F. Demirci, "Cifar-10 Image Classification with Convolutional Neural Networks for Embedded Systems,"2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA), Aqaba, Jordan, 2018, pp. 1-2, <https://doi.org/10.1109/AICCSA.2018.8612873>.

**Git repository link:**

<https://github.com/Fnu-Anvika/CIFAR-10-Project>