A PROJECT REPORT ON

Image Classification

BY

VAMSIKRISHNA NEELAM (U01074399)

neelam.11@wright.edu

UNDER THE GUIDENCE OF

Dr. Wen Zhang

Assistant Professor

WRIGHT STATE UNIVERSITY

2023 - 2024

Image Classification

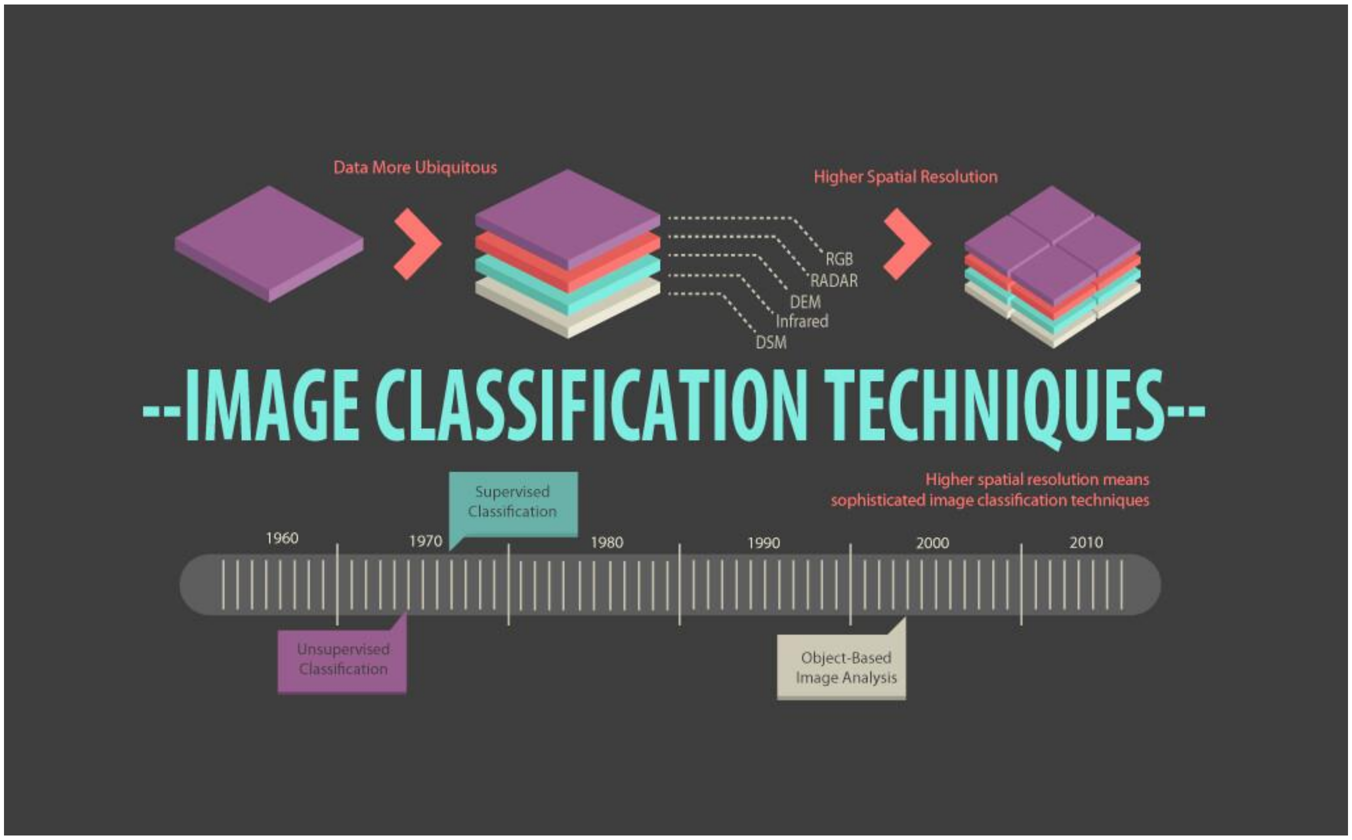

Data More Ubiquitous
Higher Spatial Resolution
RGB
RADAR
DEM
Infrared
DSM
--IMAGE CLASSIFICATION TECHNIQUES--
Supervised Classification
Higher spatial resolution means sophisticated image classification techniques
1960
1970
1980
1990
2000
2010
Unsupervised Classification
Object-Based Image Analysis

Image Classification

Abstract:

In the modern era, there is much advancement in Artificial Intelligence and Machine Learning. There are various domains in this field. Deep Learning is a subset of Artificial Intelligence and Machine Learning. So as a part of Deep Learning there are various domains like Image Classification, Speech Recognition, Face Detection, Text Translation, Object Detection, Image recognition, Visual Art Processing, Bioinformatics, Drug discovery and Toxicology etc. The Image classification project presented here is expected to work on the United States of America Vehicle Number Plates or License Plates dataset. All the data used in the project is being taken from the Kaggle website[1]. The dataset mentioned here requires a preprocessing step where the images of the larger size are being transformed into small size images and all the images are expected to be of the same size. After the preprocessing step, we are considering the two deep learning models called Resnet 18 and Alex Net which are trained on these images and are expected to classify the new unseen images. The performance of the models along with various details have been mentioned in the report clearly.

Objectives:

The main goal or objective of the Image classification domain is to take a set of input images and learn various features by extracting various patterns from those images and then classify the unseen new images the category to which it belongs to. We are training each of the selected two deep learning models separately by making the models to learn the features or extract the feature patterns from the images that are being shown during the training phase. Then we expect each of the models to classify the unknown and new unseen images to the class to which it belongs from the previous experience i.e., which it has learnt from the training phase. By doing this we can use the models that we have used here for real-world applications like predicting the car that it belongs to a particular state in the USA in the event of an accident, theft or in any other circumstances. The main potential goal of the Image Classification task that we are doing here is to make our chosen deep learning models to accurately predict the class of the images.

Dataset Description:

The data set called the United States Vehicle Number Plates or License Plates data was taken from Kaggle Website. The Data mainly comprises of all the different number or license plate images from 56 States in the USA. Because there are 56 different states in USA it can be said that there are either 56 categories of Images or 56 different classes of the Images in the dataset. The whole dataset is contained in the form of train and test dataset which contains 8161 images and 280 images respectively and a total of 8441 images, out of which the train and test dataset contributes to 96% and 4% of the total images respectively. Each image in any dataset is 224 pixels wide and 128 pixels high. The number plate contributes to around 90% of the area of the image approximately. The size of the image varies from around 15kb to a maximum of 25kb approximately.

Preprocessing:

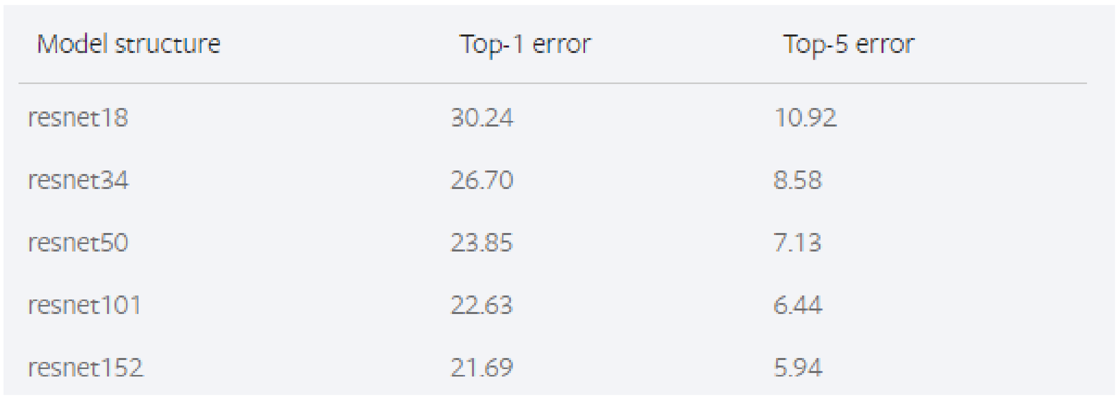
The preprocessing for the chosen image dataset is as described here. Because the number plates occupy approximately around 90% of the images, it is not required to crop or resize the images to remove the unwanted data. The image preprocessing which is used to preprocess the images is to convert or resize the images to 128 by 128 pixels for training and testing the models chosen. This makes the training of the Neural Network much better and faster. Then, after this transformation we will convert each image into a corresponding tensor to start training our chosen deep learning model. Once after the completion of the transformation of the images into the tensors, we are calculating the metrics like mean and standard deviation of all the images to know the different aspects of the images. If the mean is nearer to 0(zero), then we can say that the images in the dataset are dark and vice versa, if the mean is nearer to 1(one), then we can say that the images in the dataset are bright. Hence, the Mean specifies the general brightness of an image. The standard deviation of the image talks about the gross measure of the imprecision or variation about the target value of light intensity at each data point. If the standard deviation of a dataset is close to 0(zero), then we can say that it is close to the mean of the dataset else when the standard deviation has a larger value then we can say that the data is spread further away from the mean. When standard deviation is close to mean, we can say that the data is having most similar values whereas when the standard deviation is away from mean we can say that the data is having different values. So, at this point we are then normalizing the whole dataset around the calculated mean and standard deviation of the images. This normalization improves the performance of the machine learning model thereby reducing the errors in training the models and leading to a faster convergence rate. We are also randomly flipping the images to a certain angle because all the images chosen are at same perfect human readable angle, which is good, but the model may become super-biased, to overcome this problem the images are being flipped at an angle of 10 degrees.

Choice of the Models:

The two different deep learning models that are chosen for the image classification task are ResNet18 model and Alex-Net model. All the models chosen for our project are non-pretrained. The models that are being used here for the Image classification purposes are being imported from Py-Torch Framework and they are being used as they are without any modifications to the number of layers, nodes etc., because they are being stated as the efficient models after a lot of research has been made.

Details of the ResNet-18 model:

The different available versions of the Resnet as of now are ResNet18, ResNet34, ResNet50, ResNet101, ResNet152. As the number in the Resnet name is increasing the Top-1 error and the Top-5 error are improved very significantly as shown in Table 1 below.



Model structure	Top-1 error	Top-5 error
resnet18	30.24	10.92
resnet34	26.70	8.58
resnet50	23.85	7.13
resnet101	22.63	6.44
resnet152	21.69	5.94

Table 1

Of all the above models, the chosen model for the image classification task is ResNet18. It has 18 deep layers. The main building block in the ResNet18 is the Basic Block with each basic block consisting of a combination of the Convolutional Layer1, Batch Normalization Layer1, ReLu layer, Convolutional Layer2 and a final Batch Normalization Layer2. So, each layer in the Resnet has 2 basic blocks accounting to a total of 4main layers and a final average pool layer.

The basic architecture of the ResNet18 model is as shown in Table 2 and Figure 1 below.

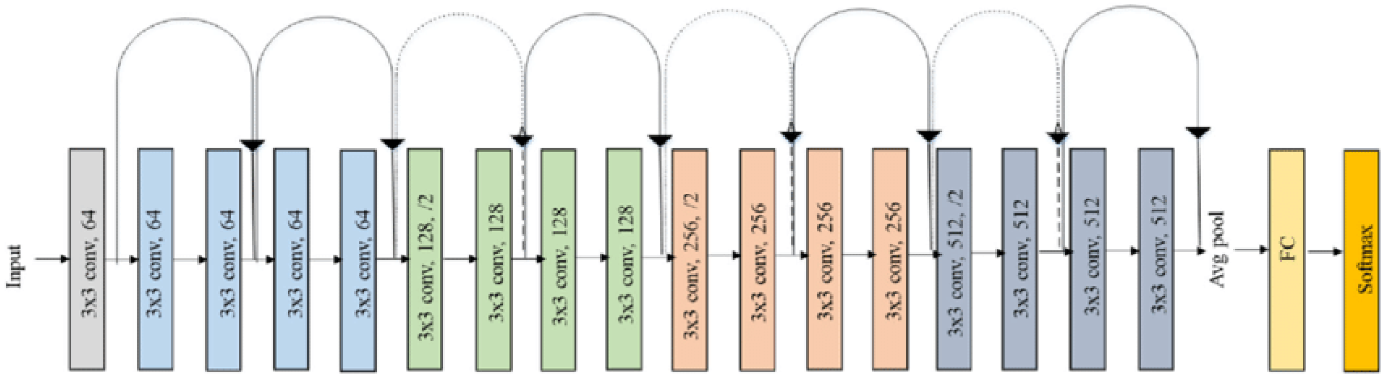
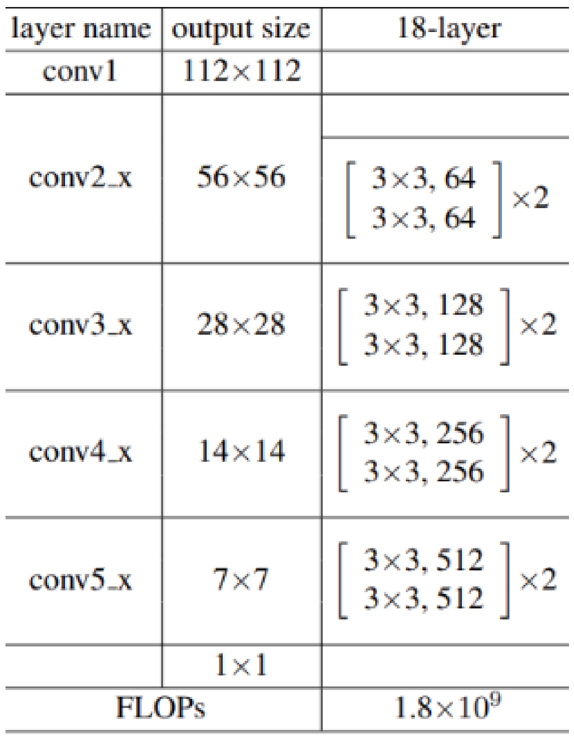


Figure 1



layer name	output size	18-layer
conv1	112×112	
conv2_x	56×56	
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$
	1×1	
FLOPs		1.8×10^9

Table 2

Details of the AlexNet model:

The AlexNet consists of 5 convolution layers, 3 max-pooling layers, 2 Normalized layers, 2 fully connected layers and 1 SoftMax layer. Each convolution layer consists of a convolution filter and a non-linear activation function called ReLu. The architecture diagram of the Alex Net is as shown in Figure 2 below.

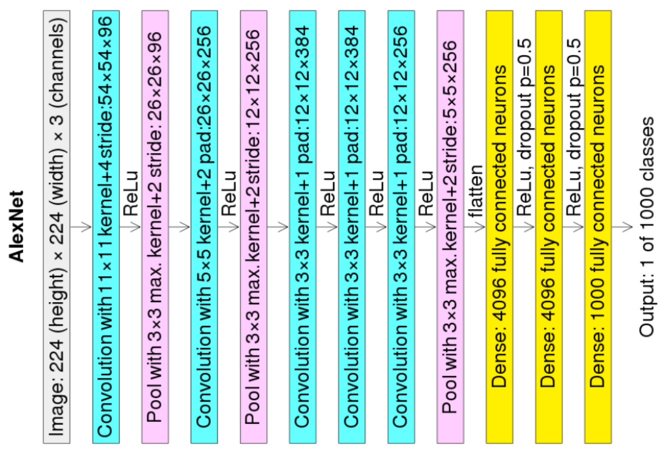

AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max.kernel+2stride:12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max. kernel+2 stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

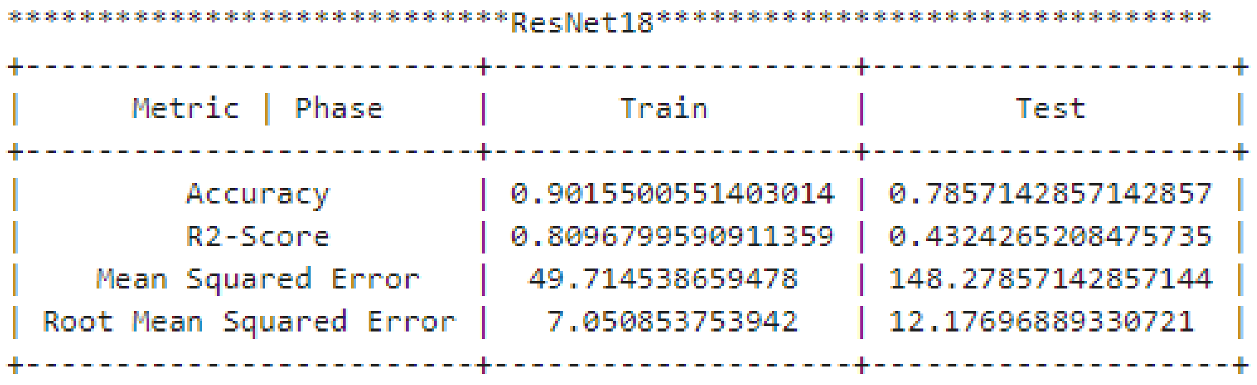
Figure 2

Training and Testing:

The training process that is being used to train the neural network efficiently is the mini batch gradient descent with a batch size of 32. So, all the images are processed by the neural network in batches of 32 images per batch which also makes training the neural networks efficient and faster. All the training dataset when seen once completely, we will start testing the accuracy of the model by sending the unseen images to the model and we expect it to identify the images correctly. During the training phase of our model, it is given various Hyper Parameters like the Learning Rate, Momentum and Weight Decay. The optimizer that is being used for training both models is the Stochastic Gradient Descent (SGD) and the loss is calculated using the Cross Entropy Loss function. The main reason for using the SGD is because it is very efficient when compared to other optimizers and helps to reduce the cost function based on the prediction error in backward and forward propagation of the neural network training process. Besides this, it also speeds up the training process. The cross-entropy loss function used here tells how close and how accurately the model can classify the images from the test dataset. After adjusting the hyper parameters for several training and testing runs over a range of 100 epochs for each of the Models, the ResNet18 on the chosen dataset is found to perform best at the learning rate of 0.01 and weight decay of 0.003. Similarly, the Alex Net performed best at learning rate of 0.01 and weight decay of 0.001.

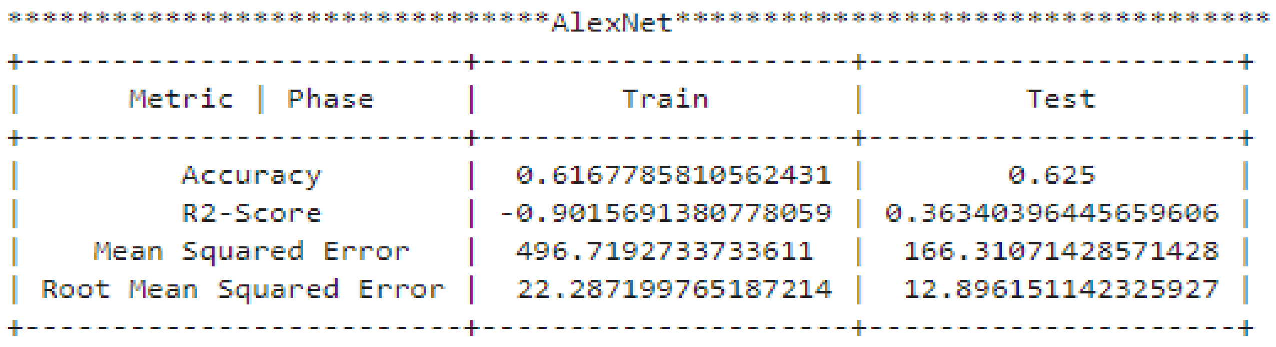
Results:

The training and the testing accuracies at the best performance and over a range of 100 epochs is shown in Table 3 and Table 4 for ResNet18 and Alex Net respectively.



```
*****************************ResNet18**********************************
+--------------------------+---------------------+---------------------+
|      Metric | Phase      |        Train        |        Test         |
+--------------------------+---------------------+---------------------+
|         Accuracy         | 0.9015500551403014 | 0.7857142857142857 |
|         R2-Score         | 0.8096799590911359 | 0.4324265208475735 |
|    Mean Squared Error    |  49.714538659478   | 148.27857142857144 |
| Root Mean Squared Error  |   7.050853753942   | 12.17696889330721  |
+--------------------------+---------------------+---------------------+
```

Table 3



```
********************************AlexNet***********************************
+-------------------------+----------------------+----------------------+
|      Metric | Phase     |        Train         |         Test         |
+-------------------------+----------------------+----------------------+
|        Accuracy         |  0.6167785810562431  |        0.625         |
|        R2-Score         | -0.9015691380778059  |  0.36340396445659606 |
|   Mean Squared Error    |  496.7192733733611   |  166.31071428571428  |
| Root Mean Squared Error |  22.287199765187214  |  12.896151142325927  |
+-------------------------+----------------------+----------------------+
```

Table 4

We have also calculated the various metrics for the models like the Accuracy, Precision, Mean Squared Error and Root Mean Squared Error for comparison and evaluation of the models. Accuracy can be stated as the percentage of the correct classifications made by the model. Out of both models the ResNet18 is much more accurate than Alex Net model. The R2-Score talks about the fit of the model and it should be between 0 and 1. An R2-Score closer to 0 indicates that the model is the best fit for the data i.e., there is not much difference between the actual and predicted values. But when we consider the R2-Score for the Alex Net model during training phase it is less than zero. So ResNet18 is better when compared to Alex Net. The Mean Squared Error (MSE) when close to 0(zero) specifies that the model has made minimal errors during the prediction phase. So, ResNet18 has lower MSE values during both training and testing phases making it the best. The model with the lower Root Mean Squared Error (RMSE) says that it has better predictions. So, of both ResNet18 is best. The Epoch Losses, training and testing accuracies when plotted as graphs are as shown in Figure 2 below.

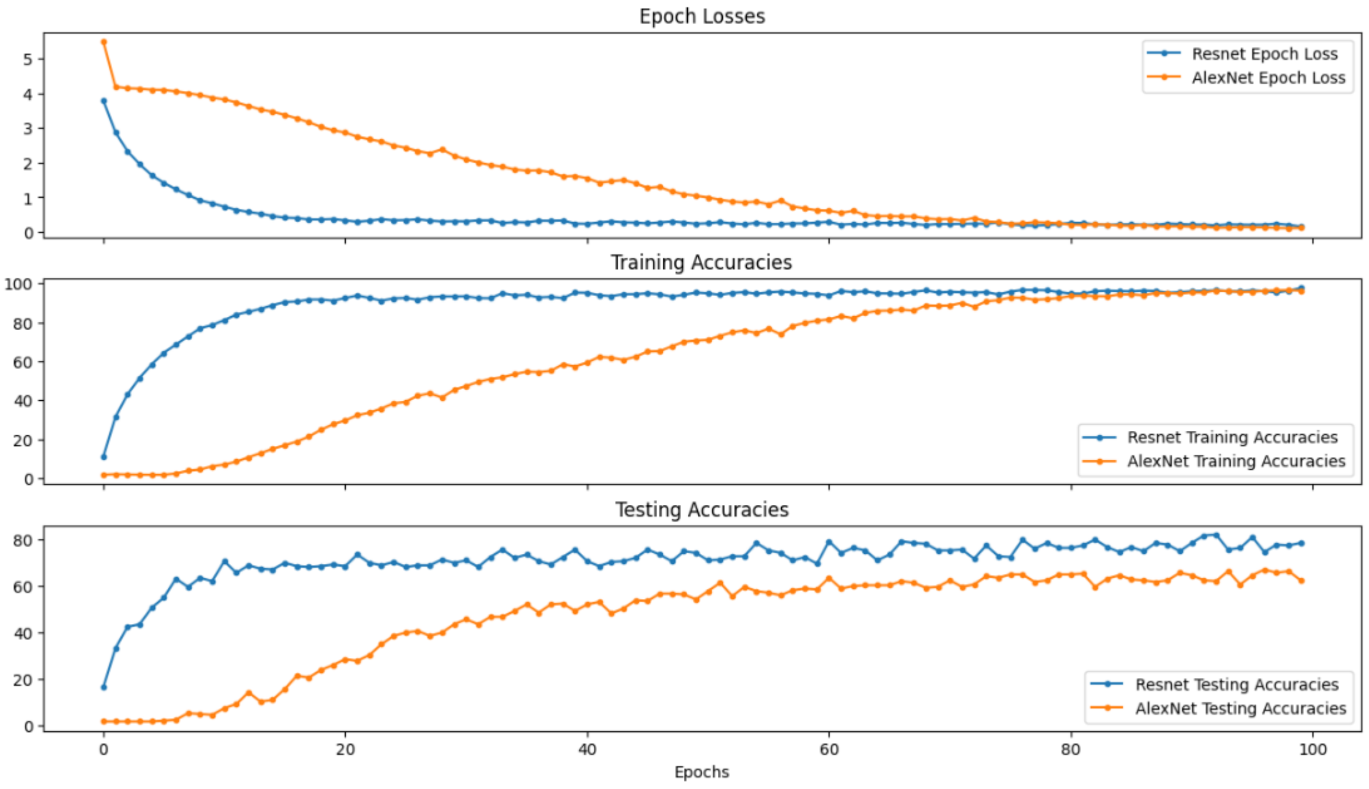
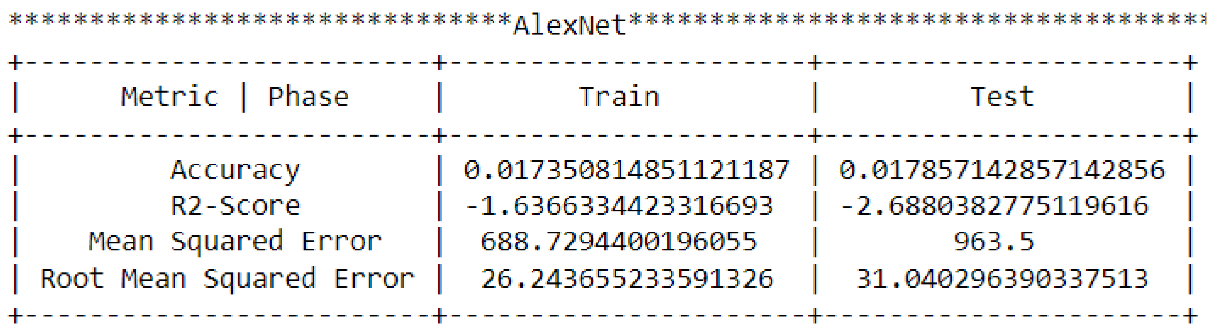


Figure 2

The other training phases and the results over a range of 100 epoch are as shown in table 5 and table 6 with a learning rate of 0.1 and weight decay of 0.0003 for ResNet18 and learning rate of 0.01 and 0.003 weight decay for the Alex Net model. The corresponding epoch loss, training accuracy and testing accuracies plotted as a graph are as shown in Figure 3.

```
****************************ResNet18********************************
+--------------------------+--------------------+--------------------+
|      Metric | Phase      |       Train        |        Test        |
+--------------------------+--------------------+--------------------+
|         Accuracy         | 0.8179745129273374 | 0.7464285714285714 |
|         R2-Score         | 0.646239640801423  | 0.5178263841421736 |
|    Mean Squared Error    | 92.40767798063963  | 125.96785714285714 |
|  Root Mean Squared Error | 9.612891239405533  | 11.223540312346062 |
+--------------------------+--------------------+--------------------+
```

Table 5



```
*******************************AlexNet***********************************
+--------------------------+----------------------+----------------------+
|      Metric | Phase      |        Train         |         Test         |
+--------------------------+----------------------+----------------------+
|         Accuracy         | 0.017350814851121187 | 0.017857142857142856 |
|         R2-Score         | -1.6366334423316693  | -2.6880382775119616  |
|    Mean Squared Error    |  688.7294400196055   |        963.5         |
| Root Mean Squared Error  |  26.243655233591326  |  31.040296390337513  |
+--------------------------+----------------------+----------------------+
```

Table 6

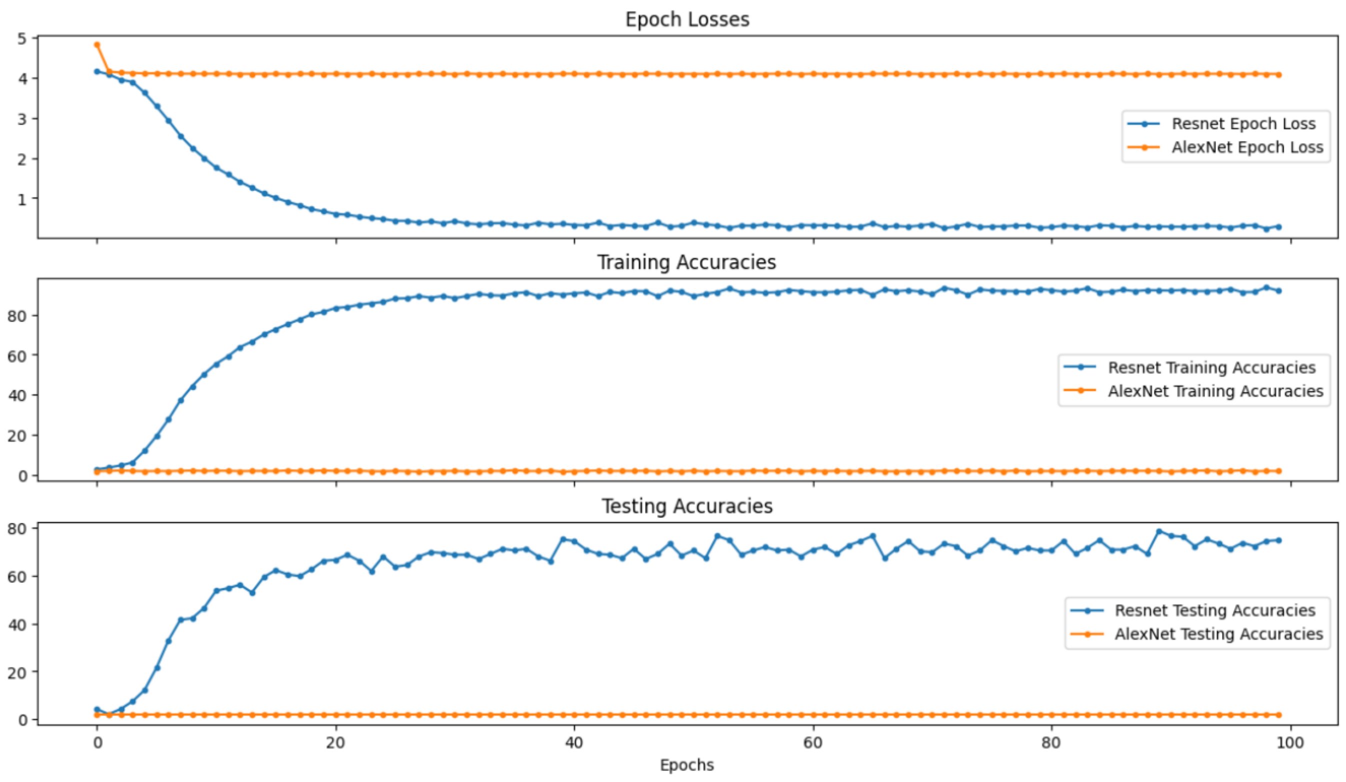


Figure 3

Limitations and Challenges:

The ResNet18, as discussed earlier, requires a lot of computational power and computing resources which is one of the limitations when there are limited computing resources. The Resnet 18 when used on very smaller datasets it leads to a problem of Overfitting, so to avoid the problem of overfitting regularization must be applied. The ResNet is generally less interpretable i.e., it can learn and understand the patterns among the complex and crucial data, but these cannot be expected or considered where the interpretability is most important such as in cases of fraud detection or medical diagnosis. One more limitation of the Neural networks presented here is that they both expect all images to be uniformly sized and the sizes to be as 124*124 pixels, 224*224 pixels etc. The AlexNet is not deep enough to de the best image classification because it is only 8 layers deep. The AlexNet is not able to solve the problem of the Vanishing Gradients which has been solved by the other deep neural networks like ResNet18 and other versions of the ResNet.

Summary:

Based on all the above observations and results of the two models we can state that the ResNet18 model is most suitable for the chosen dataset and much better when compared to the AlexNet model. The main reason for not considering the ResNet32 or other high end deep neural network models is because of the time constraint. The other high-end models have much deeper layers requiring more computing resources, processing power and much time for completing training and testing.

Contributions:

Using the Py-Torch Framework, I have developed the code for the ResNet18 and AlexNet models used here by choosing the best optimizer and the best loss function along with the fine tuning of the hyper parameters for achieving the best results. As a pre step I also Performed the Data preprocessing. Trained and tested the models on the chosen dataset and calculated the metrics for the models. Represented the training and testing accuracies of the models along with epoch loss as a line graph making easy to get a quick understanding of the models.

Technologies and Frameworks used:

Python 3.11.6 along with several other packages

PyTorch Framework

Jupyter Notebook

Scikit Learn

References:

[1]. https://www.kaggle.com/datasets/gpiosenka/us-license-plates-image-classification
[2]. https://pytorch.org/hub/pytorch_vision_resnet/
[3]. https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8
[4]. https://debuggercafe.com/implementing-resnet18-in-pytorch-from-scratch/
[5]. https://medium.com/@siddheshb008/alexnet-architecture-explained-b6240c528bd5