# Deeper Networks for Image Classifications

Syeda Mehwish Naseem

Department of EECS, QMUL

[eecsXXX@qmul.ac.uk](mailto:eecsXXX@qmul.ac.uk)

### Introduction

In recent years deep convolutional neural networks have achieved great success in computer vision tasks such as image classification and detection. The advancements in computer vision is not only attributed to more powerful hardware, large datasets or bigger models but is mainly a consequence of new ideas, algorithms and improved network architecture. The aim of this paper is to investigate two such networks, VGG16 and ResNet16 and their effectiveness for image classification. The dataset chosen for this task is MNIST and CIFAR10. Each model is implemented using PyTorch library and is designed in a fashion so it can easily be extended and modified.

### Related Work

VGG was developed by Simonyan and Zisserman [1] in a paper addressing depth of convolutional neural networks by performing thorough analysis on ImageNet. Their technique utilized a large number of 3x3 convolutional layers on top of each other increasing the depth of the network while at the same time keeping all other parameters fixed. At the time of publication increasing depth of deep neural networks was a mainstream idea to improve the performance. Similar improvements were achieved by Goodfellow el al. [2] by using deeper network architecture for the street number recognition task. Another famous architecture named GoogleNet by Szegedy et al. [3] is a very complex deep convolutional network architecture that was able to win the ILSVRC-2014 competition. 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.

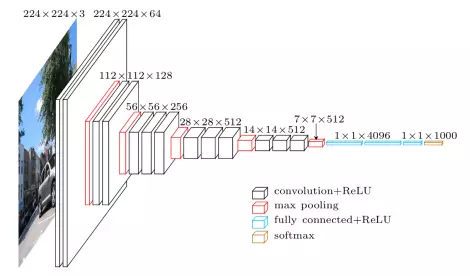
In the following sections VGG and ResNet models will be explored in detail.

### Model Description

#### 3.1 Model Architecture

##### 3.1 (i) VGG Architecture

One key finding of the VGG paper was that increasing depth of the convolution network from 11 layer to 19 layers lead to a decrease in classification error on ImageNet. Furthermore, compared to previous models the number of parameters were not increased greatly.

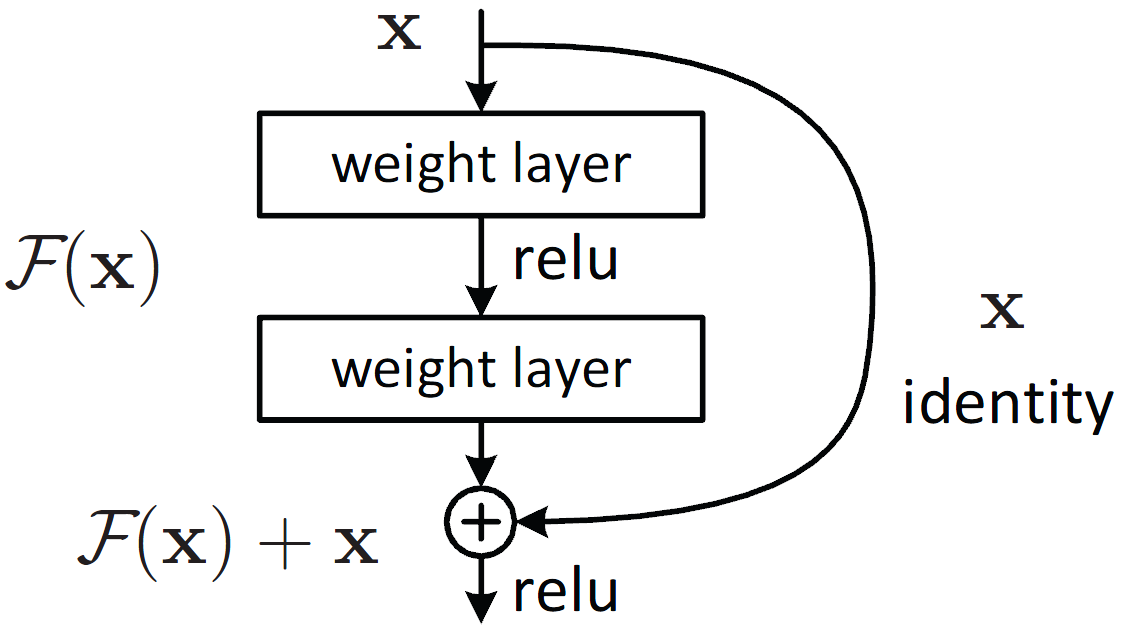


###### Figure 1

The above diagram shows the VGG16 architecture as described in the original paper. The input image is passed through multiple convolutional layers with the filter size of 3x3 (which is the minimal size of filter required to capture the surrounding pixel information. In addition to the 3x3 filter, this model also uses a 1x1 convolution filter in one of the configurations which can be seen as a linear transformation of input channels. The stride length chosen is 1 pixel. 1 pixel padding of the convolution layer of such that the spatial resolution is preserved after the convolution. Five max-pooling layers are used in between convolution layers with 2xx2 pixel window and stride length of 2. The stack of convolution and pooling layers are followed by three fully connected layers. VGG network used mini-batch stochastic gradient descent with batch size of 256 and momentum of 0.9. Regularization consisted of L2 weight decay with a penalty multiplier of 0.0005 and two dropout layers after the first two fully connected layers.

##### 3.1 (ii) ResNet Architecture

In an effort to effectively train deep neural networks, He et al. introduced a deep residual learning framework named ResNet. Their aim was to address the issue of degradation in accuracy which occurred with very deep networks. In order the solve this problem they proposed a network with residual connection as show below



###### Figure 2.

Instead of hoping every few stacked layers directly fit a desired underlying mapping, they explicitly let these layers for a residual mapping. The formulation of F(x)+x can be realized by feedforward neural networks with shortcut connections. Shortcut connections are those skipping one or more layers shown in Figure 2. The shortcut connections perform identity mapping, and their outputs are added to the outputs of the stacked layers. By using the residual network, there are many problems which can be solved such as:

* ResNets are easy to optimize, but the “plain” networks (that simply stack layers) show higher training error when the depth increases.
* ResNets can easily gain accuracy from greatly increased depth, producing results which are better than previous networks.

The paper compared this architecture to the VGG network - the baseline network used has fewer filters and lower complexity than VGG. For example, VGG19 has 19.6 billion FLOPS, while both the plain baseline and residual networks with 34 layers have 3.6 billion FLOPS. Additionally, they utilized batch normalization which ensures that forward propagated signals have non-zero variance, ensuring in turn that the gradient vanishing problem is dealt with in both directions.

#### 3.2 Training

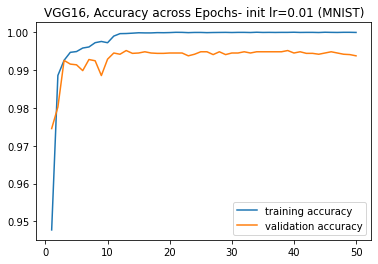
In total four models were investigated, VGG13, VGG16, ResNet20 and ResNet44. Experimentation was done by trying different learning rates, batch sizes and

##### 3.2 (i) VGG Training

The network is trained on MNIST and CIFAR datasets. VGG is not mainly designed for a dataset with small input image size like the MNIST which has the dimensions 28x28. Two options were considered to address this issue.

1. Resize the MNIST from 28x28 pixels to 224x224 without making any changes to the network.
2. Resize MNIST to the minimum allowed size for VGG and change the network accordingly.

To speed up the training process, the choice to lower the resolution to 32x32 was made. This meant that the convolution layer gradually reduced the dimensions to 1x2 with 512 filters. Therefore, when reaching fully connected layers, the input size became 512x1x1 compared to the original expected size of 512x7x7. After experimenting with multiple output sizes the final VGG model consisted of the same number of layers as the original paper with convolutional layers unchanged and fully connected layers as FC(512, 128) - ReLU - Dropout(0.2) - FC(128,128) - ReLU - Dropout(0.2) - FC(128,10).



Multiple approaches were tried to training VGG with varying initial learning rate and batch sizes. The best results were obtained with an initial learning rate of 0.01 and dropping to a learning rate 0.001 after ten epochs using batch size of 128. The training process showed that the VGG model was quickly able to converge at around 10-15 epochs and further training did not provide much improvements in the accuracy of the model.

##### 3.2 {ii) ResNet Training

For ResNet no changes to the dimensions of input image were performed. The suggested batch size of 128 didn’t work well; instead a batch size of 32 over 200 epochs worked well for CIFAR dataset. The implementation of ResNet is done by using 3x3 convolution layers with global average pooling layers. Batch size of 32 is used with 200 epochs and the weight decay of 0.0001 and momentum 0.9 was used for training. A learning scheduler with the initial learning rate of 0.1 is used dropping to 0.01 after 100 epochs and then further dropping to 0.001 after 150 epochs. Several data augmentation tricks are also employed, including padding four pixels on each side, cropping randomly to 32x32, random rotation and horizontal image flipping.

### Experiments

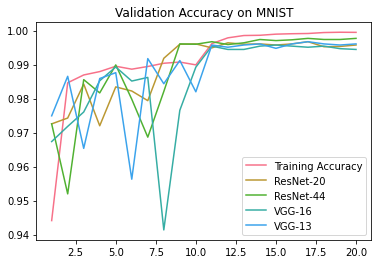
#### 4.1 Datasets

The MNIST dataset of handwritten digits has a training set of 60,000 examples and a test set of 10,000 examples with each image having resolution of 28x28 pixels. The images are grayscale and are divided in ten classes with each class representing one digit ranging from 0 to 9.

The CIFAR-10 dataset consists of 60,000 coloured images with 32x32 pixels resolution divided in ten classes. The dataset is divided into 50,000 training set and 10,000 test set images.

#### 4.2 Testing Results

Below tables show the best accuracy achieved for each model on MNIST and CIFAR10 dataset with the same batch size of 128, SGD optimizer and an initial learning rate of 0.1 with a schedule to drop to 0.01 after 15 epochs.



|  |  |  |
| --- | --- | --- |
| **Model** | **MNIST** | **CIFAR10** |
| VGG13 | 99.6 | 86.39 |
| VGG16 | 99.44 | 86.95 |
| ResNet20 | 99.57 | 89.87 |
| ResNet44 | 99.77 | 91.52 |

##### 4.2 (i) VGG Experiments

VGG model accuracy hit a plateau after reaching 10 epochs. The model was trained for 50 epochs with the best results obtained by starting with the learning rate of 0.01 and dropping to 0.001 after 10 epochs. Below are some examples of misclassification done by the model although the confusion matrix showed that no specific classes were getting misclassified at a higher rate compared to others.



Experimentation during training was done to achieve the best performance. A variety of settings for the optimizer were used witht best results being obtained from the original values from the ResNet model.

|  |  |
| --- | --- |
| **Parameters (VGG16)** | **Accuracy (CIFAR)** |
| Adam, lr=0.1 | 83.44 |
| SGD, lr=0.1 | 83.87 |
| SGD, lr=0.1, momt=0.9, wd=0.0001 | 84.08 |
| SGD, lr=0.1, momt=0.9, wd=0.0001 + decreasing schedule | 85.14 |

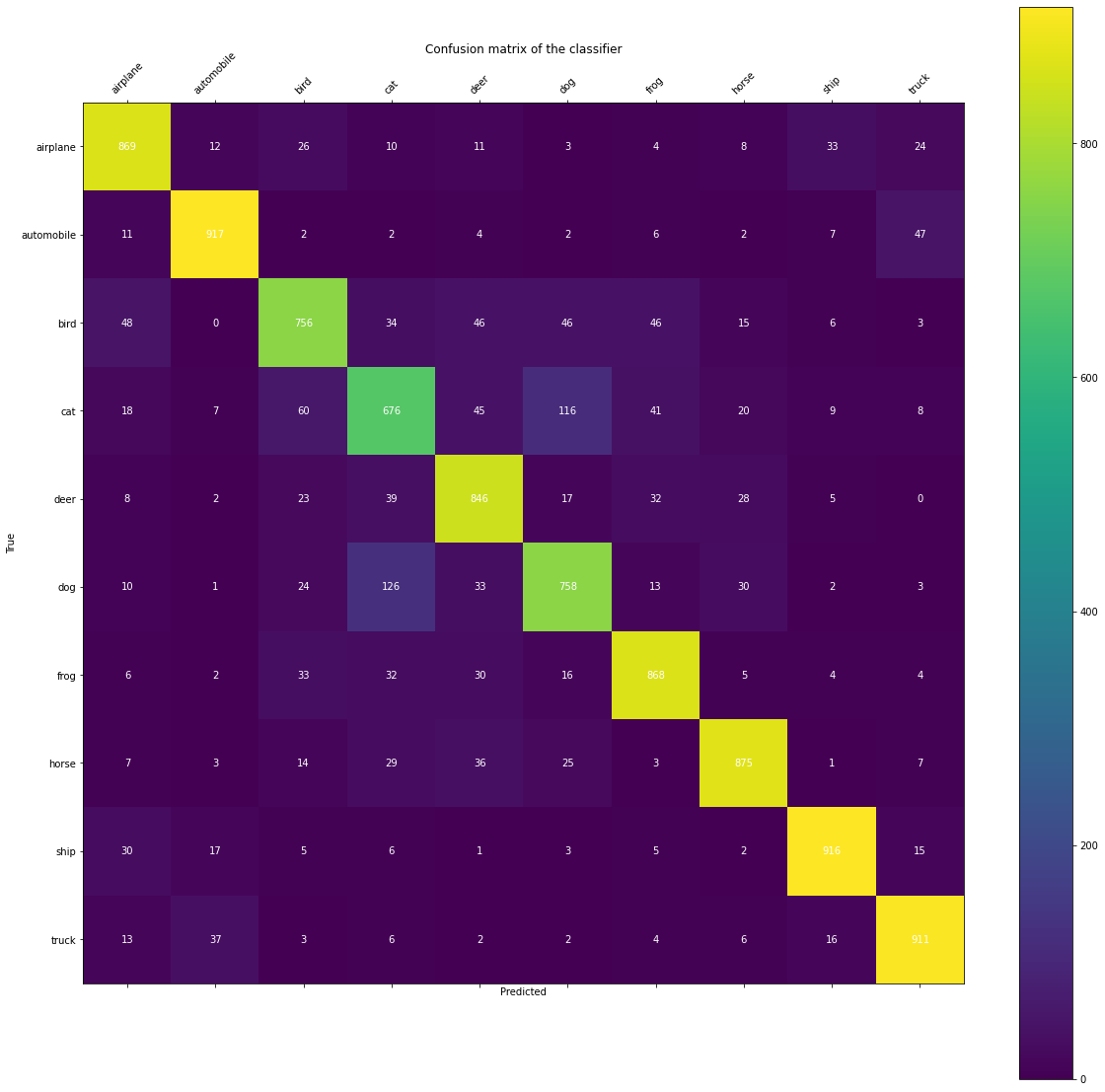
Experiments with different batch sizes (32, 64, 100, 128) showed the best results being observed witht the batch size of 100.

##### 4.2 (ii) ResNet Experiments

ResNet model produced best overall results with an initial learning rate of 0.1 and dropping to 0.01 after 100 epochs and then further dropping to a learning rate of 0.001 at 150 epochs. Out of all the models investigated, ResNet44 achieved the best accuracy.



On the MNIST dataset the misclassifications were rare but for the CIFAR dataset according to the confusion matrix the most misclassification occurred between cat and dog.



###### Figure 3

Below are a few examples of mis-classified images from MNIST and CIFAR dataset.



###### Figure 4



###### Figure 5

### Conclusion

In this paper I evaluated two deep convolutional networks named VGG and ResNet respectively. From the results it was observed that the representation depth is beneficial for classification accuracy, and the state-of-the-art performance on the dataset can be achieved using a deep convolutional neural network. The architecture of VGG is much simpler than ResNet but the training time of ResNet is much less compared to VGG. Additionally, it is evident that augmentation to model improves the generalisation of model resulting in improved accuracy.

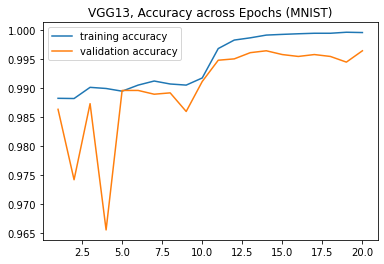
Additionally, hyper parameter tuning was tricky on the CIFAR dataset. The original performance quoted in the papers were hard to achieve indicating room for improvement in the implemented models.

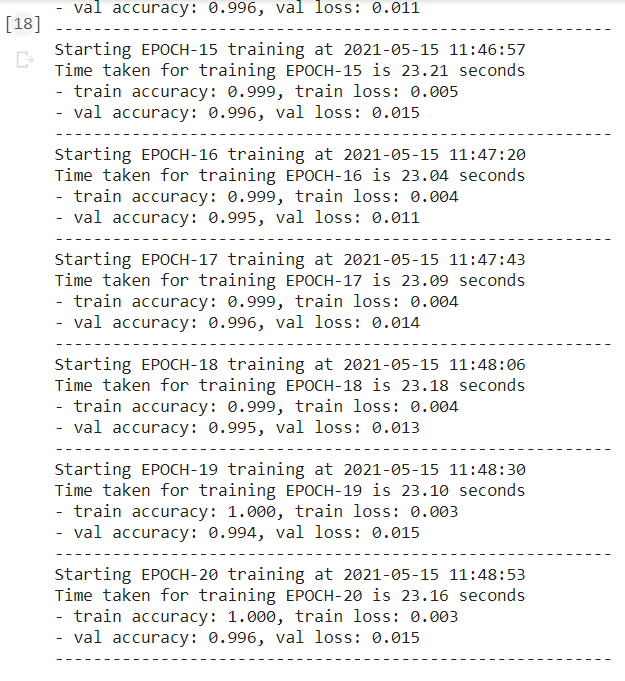
### References:

1. Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015.
2. Ian J. Goodfellow and Yaroslav Bulatov and Julian Ibarz and Sacha Arnoud and Vinay Shet. Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks, 2014.
3. Christian Szegedy and Wei Liu and Yangqing Jia and Pierre Sermanet and Scott Reed and Dragomir Anguelov and Dumitru Erhan and Vincent Vanhoucke and Andrew Rabinovich. Going Deeper with Convolutions, 2014.
4. Olga Russakovsky and Jia Deng and Hao Su and Jonathan Krause and Sanjeev Satheesh and Sean Ma and Zhiheng Huang and Andrej Karpathy and Aditya Khosla and Michael Bernstein and Alexander C. Berg and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2015.

### Appendix:

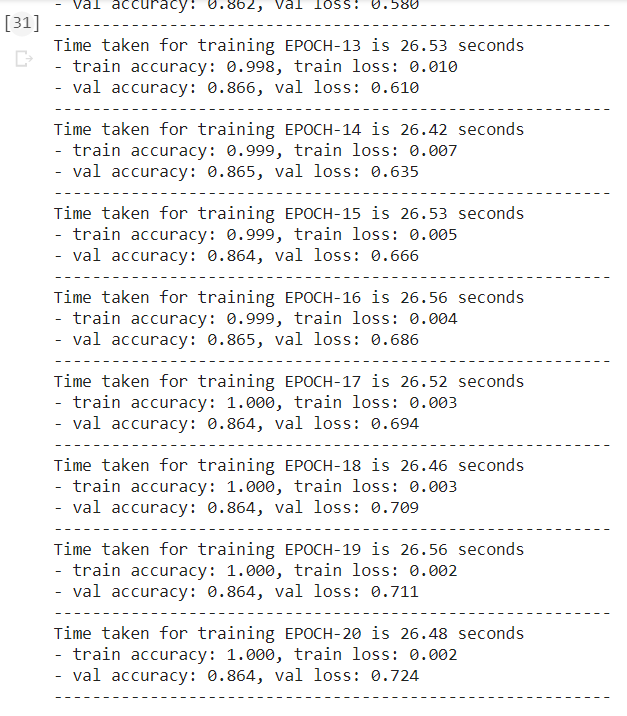
###### VGG13 training on MNIST dataset





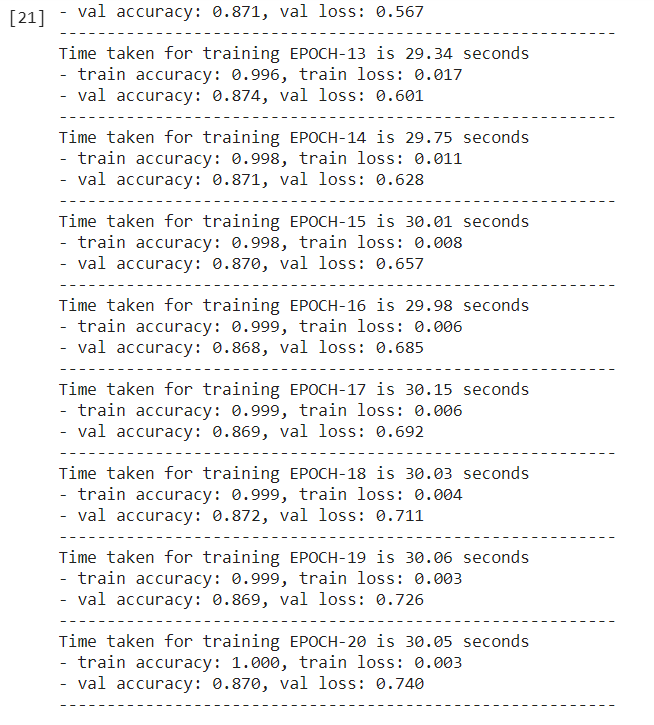
###### VGG13 training on CIFAR dataset





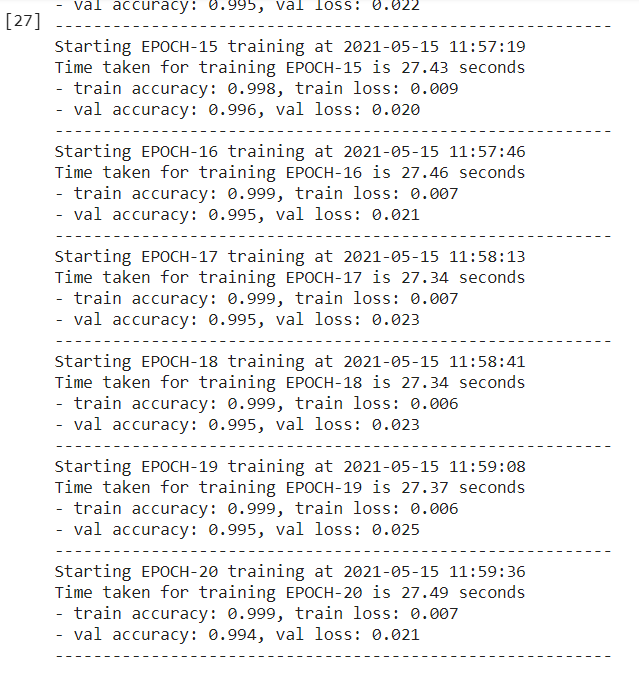
###### VGG16 training on CIFAR



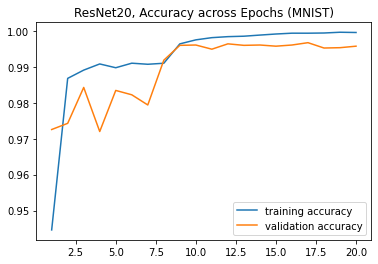


###### VGG16 training on MNIST

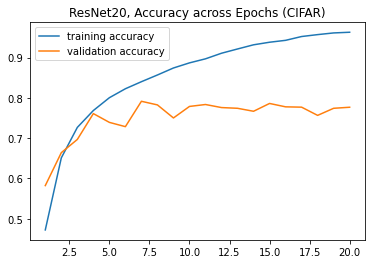


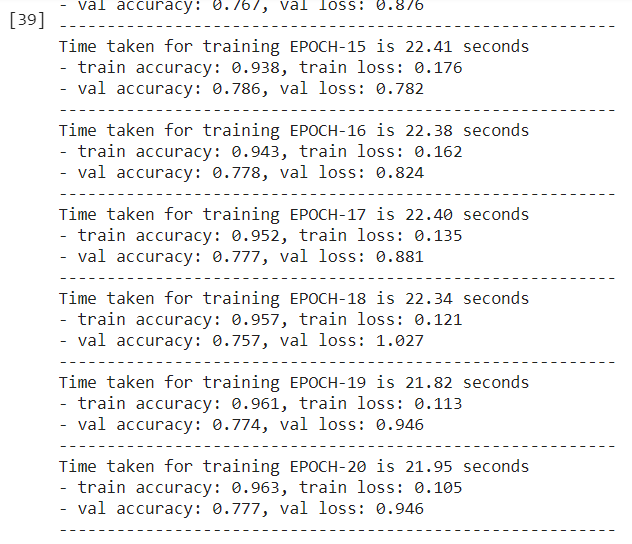


###### ResNet20 training on MNIST dataset

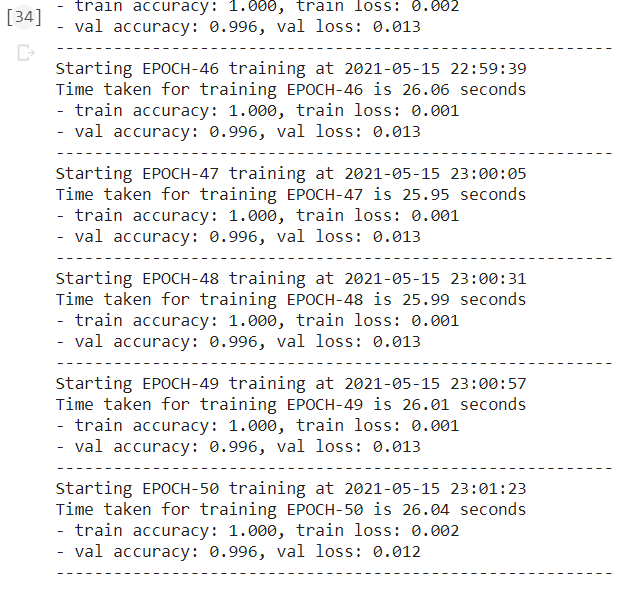


###### ResNet20 training on CIFAR dataset



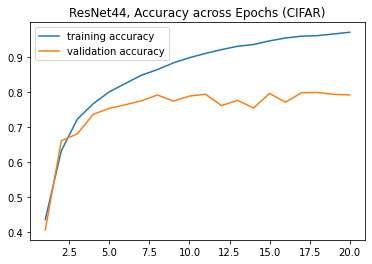


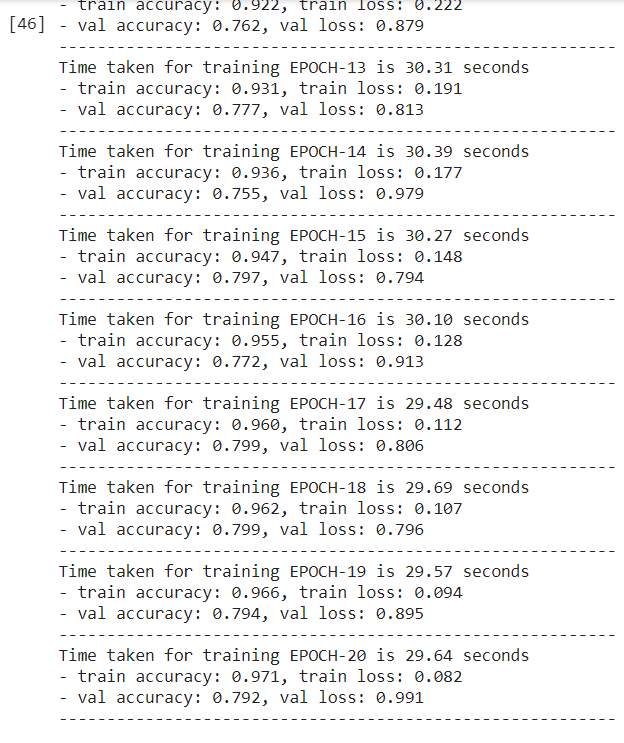
###### ResNet44 training on MNIST dataset





###### ResNet44 training on CIFAR dataset





###### CIFAR validation accuracy for all models

