ECS795P CW3 : Deeper Networks for Image Classification

Animesh Devendra Chourey
Department of Electronic Engineering and Computer Science
Queen Mary University of London
London, UK
ec21949@qmul.ac.uk

***Abstract*—Convolutional Neural Networks (CNN) are the reason behind the recent breakthroughs and developments in deep learning. They are easily the most popular and are ubiquitous in the image data space, and work exceptionally well on computer vision tasks like image classification, object detection and image recognition. In this paper, Image Classification is performed using the models VGG, ResNet networks.The datasets used to evaluate these networks are MNIST and CIFAR datasets.**

***Index Terms*—CNN, VGG, ResNet, GoogleNet, MNIST, CIFAR**

I. Introduction

Image Classification is a challenging task, and comes difficult for computers to perform them. Early image classification relied on breaking down the images into individual pixels. However, the issue with this was that images of two same thing can look extremely different because of different background, angles, illumination and poses, which made it considerably difficult to categorise the images. Deep Learning allows to identify and extract features from images. Image classification with deep learning most often involves CNN. In the last decade the researchers have managed to achieve high accuracy in solving image recognition and classification tasks with deep convolutional networks. With mass availability of training data and drastic improvements in GPUs, and better algorithms, the training of deeper networks has become more feasible. The paper focuses on the performance of the deeper CNN networks such as *VGG* [6], *ResNet* [1] on the well established datasets *MNIST* [4], *CIFAR* [2] and evaluate them.

II. Critical Analysis

There has been explosive growth and development in the fields of machine learning and artificial intelligence in the recent decades many which has been fuelled by the research breakthrough in Deep Learning. The intense study and application of CNNs through 2012 to 2016 for the *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) challenge resulted in rapid development of the computer vision tasks and innovations in the architecture of the CNN models.

The first successful and important application of CNN prior to the *ILSVRC* was LeNet-5 [5].The paper describes the network as having seven layers with input grayscale images having the shape 32×32, the size of images in the MNIST dataset. The model proposes a convolutional layer followed by an average pooling layer. The pattern is repeated two and a half times before the output features maps are flattened and fed to a number of fully connected layers for prediction. Compared to modern application the number of filters are small. However the trend of increasing the number of filters with the depth of the network also remains a common pattern in the modern usage of technique.

A. AlexNet

AlexNet [3] is the architecture that should be credited for renewing the spark back into the neural network and deep learning. The architecture achieved a top-5 error rate of 15.3% with the next best score trailing far behind at 26.2%. It implemented some methods not widely used at the time but are now a requirements when using CNNs for image classification. It used *rectified linear activation function* (ReLU) after each convolutional layer instead of *tanh* or *logistic* functions(more common at the time). It also used *softmax* activation function in the output layer. The average pooling used in LeNet-5 was replaced with max-pooling. To address overfitting, the newly proposed *dropout method* was used between the fully connected layers to improve the generalization error. The architecture of AlexNet is deeper and extends some patterns from the LeNet-5. The model consists of 5 convolutional layer for the feature extraction part and 3 fully connected layers for the classifier part of the model. Another new thing implemented here was that the pattern of convolutional layer directly fed to another convolutional layer. Also, data augmentation was used for training the model in order to increase the size of the training setand thus giving the model more opportunity to learn same features in different orientations.

B. VGG

VGG model made significant changes over the AlexNet (revolutionary improvement over traditional CNNs) in terms of architectural design and improved its performance when compared to AlexNet. Till date there exists two prominent variations of the VGG model i.e. *VGG16* and *VGG19* where each consists of 16 and 19 convolutional layers respectively. That is the only significant difference between the variants. The base implementation in both the network's architecture remains the same. The network takes the input size of (224x224).

The center 224x224 patch in each image has been cropped out in order to keep the input size of image consistent. While performing the convolutional operations, filters of very small receptive fields were taken of size 3x3 (which is the smallest size to capture the notion of left/right, up/down, center). 1x1 convolution filter were also taken in one of the configuration as linear transformation of input channels. Using large number of these smaller filter has been accepted in plenty studies down the road. As there are three ReLU units instead of just one (in AlexNet), the decision function is now more discriminative. The number of parameters in the model are also fewer (27 times the number of channels as compared to AlexNet's 49 times the number of channels).

C. GoogLeNet

The network known as Inception Net was the winner of *ILSVRC* competition in 2014. As the name suggests it was proposed by a team at Google. The first version of Inception Net was known as *GoogleNet*. The key innovation was inception module. The block contains parallel convolutional layers with different sized filters i.e.(1x1), (3x3), (5x5) and a (3x3) max pooling layer. All their results are concatenated. Performing the convolutions with larger filter sizes (of 3 and 5) can become computationally expensive if there are large number of filters. To tackle this, convolutional layers of size (1x1) are used to reduce the number of filters in the inception model. This is specifically done before the (3x3) and (5x5) convolutional layers and after the pooling layer. This saved significant amount of time to train by remarkably reducing the number of parameters to 4M from 60M parameters present in the AlexNet. However, it should be noted that the network actually became wider. The GoogLeNet architecture solves the issue of vanishing gradients problem by connecting the output at different points. This was done by creating small off-shoot output networks from the main network that were trained to make a prediction.

D. ResNet

The model is known as *Residual Network*. The key to the model design is the idea of residual blocks that make use of shortcut connections where the input is kept as it is (i.e. not weighted) and passed on to a deeper layer (i.e. skipping the next layer). These residual blocks are defined by modifying the plain network by adding shortcut connections. Typically the shape of the input for the shortcut connection is the same size as the output of the residual block. A residual block is a pattern of two convolutional layers having ReLU activation where the output of the block is combined with the input to the block, e.g. the shortcut connection. The model won the competition and achieved a top-5 error rate of 3.57% at the *ILSVRC* challenge. The architecture featured heavy batch normalization and had 152 layers. One thing to note here is that even after having this many layers it had lower complexity than the VGG model. The residual network was introduced to solve the problem of vanishing gradients which was done exceptionally well by the concept of *skip connections*.

III. Method / Model Description

In this paper, I used various deeper networks for evaluating the effectiveness of deeper CNN models for image classification. Specifically, I used VGG and ResNet models on two datasets *MNIST* and *CIFAR10*.

A. VGG-16

Amongst the two variants mentioned before, from VGG16 and VGG19, I have implemented the VGG16 model. VGG16 consists of 13 convolutional layers and 3 fully connected layers. Input image dimensions are fixed to a size of (224x224). In a pre-processing step the mean RGB value is subtracted from each pixel in an image. Images are then passed to a stack of convolutional layers with small receptive filters. The first two convolutional layer consists of 64 filters followed by a max-pooling layer. Then again two more convolutional layers follow having 128 filters with another max-pooling layer. Filters of size 256 are used in the next 3 convolutional layers. Max-pooling layer is then attached nest to them. Again three convolutional layers are used after them but this time the filter size increases to 512. A max-pooling layer is added and after it same number of convolutional layers as before having same number of filters i.e. 512 followed by last max-pooling layer. Finally three fully connected layers are used with the first two having 4096 neurons and the last layer haivng 1000 neurons. This indicates that the original paper intended to predict 1000 classes. The model ends with having softmax layer in the end. Every hidden layer in the network has *ReLU* activation function. For every convolutional layer the stride is fixed to 1 whilefor all the max-pooling layers stride is set to 2. For the convolutional layers either filter size is of size (3x3) or of size (1x1). All the max-pooling layers carries out spatial pooling having pixel window of size (2x2). In order for us to fit the *MNIST* and *CIFAR* dataset to the model, in the last fully connected layer the number of neurons are changed to 10.

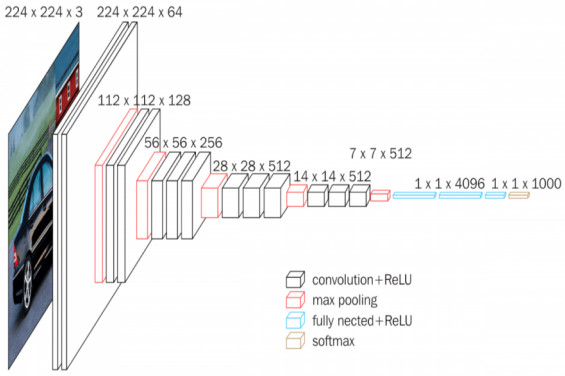

Fig. 1. VGG16 Architecture

B. ResNet

ResNet also have different variants such as ResNet18, ResNet34, ResNet50 and many more. I have implemented the

ResNet34 model amongst these variants. Similarly to VGG network, the input image dimensions here are also fixed to (224x224). The plain baseline (middle of fig2) has been inspired from the philosophy of VGG nets (left of fig.2). The convolutional layers here mostly have (3x3) filters. The convolutional layers for the same output feature map have same number of filters. Also, to preserve the time complexity of the each layer, if the size of the feature map is halved, the number of filters are doubled.

Based on the plain network, skip connections are inserted (right fig.2) which turns the network into its counterpart residual version. THe identity shortcuts F(x{W}+x) can be diectly used when the input and output are of same dimensions (solid line shortcuts in right of fig.2). However, if the dimensions increase (dotted line shortcuts in right of fig.2) it considers two options. Firstly, the shortcut performs identity mapping with zero extra padded entries for increasing dimensions. The projection shortcut in F(x{W}+x) is used to match dimensions which is done by (1x1) convolutions. Also , if the shortcuts go accross feature maps of two size, it performs with a stride of 2.

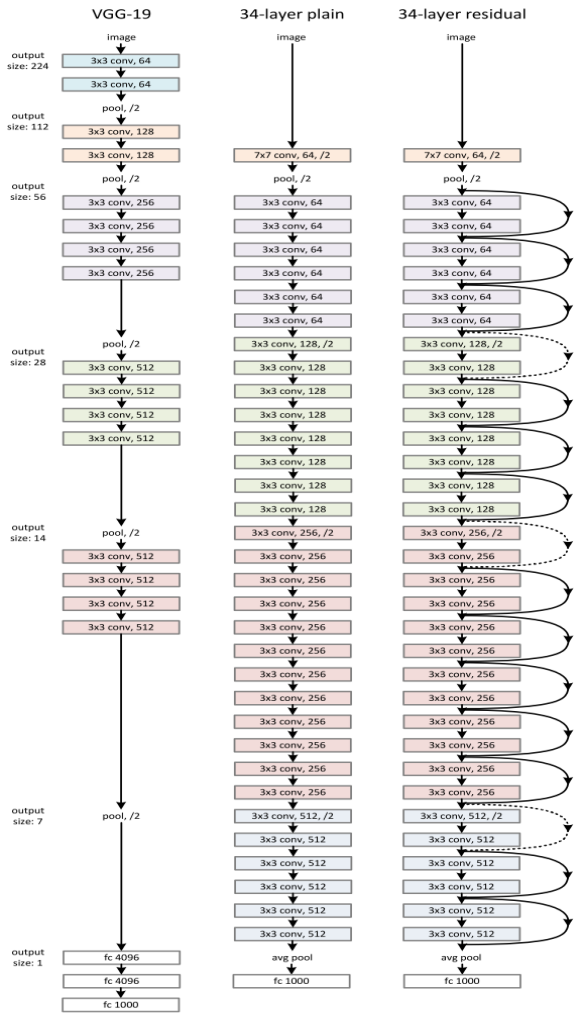

Fig. 2. ResNet34 Architecture

IV. Experiments

A. Datasets

- **MNIST**
  MNIST stands for *Modified National Institute of Standards and Technology*. The dataset was built by Yann Le Cun which consists images that are handwritten digits. The dataset has 10 discrete classes in total (from 0 to 9). This dataset contains 70,000 images in total where they are split into a training set of 60,000 images and a test set of 10,000 images. Here each image is of 28 x 28 pixels in width and height and are grayscale in nature. This dataset is quite frequently used in image classification algorithms.

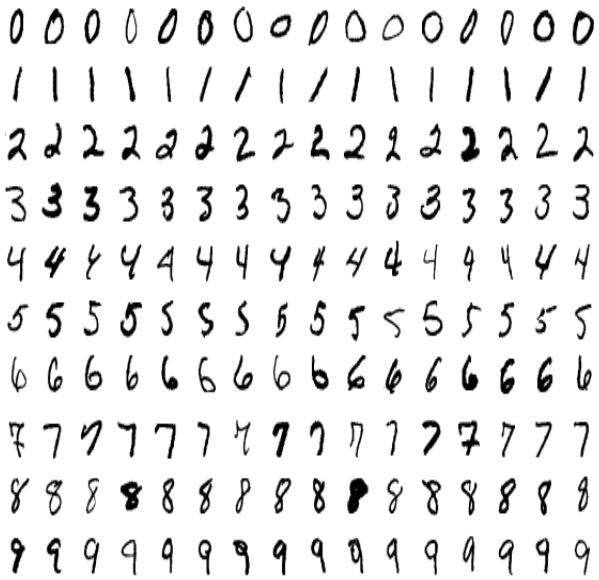

Fig. 3. MNIST Dataset

- **CIFAR**
  CIFAR stands for *Canadian Institute For Advanced Research*. To model our achitectures I have used CIFAR-10 dataset that contains 10 discrete classes amongst it. In total the dataset consists of 60,000 images with each class containing 6,000 images. In contrast to the black and white texture in the MNIST dataset, CIFAR contains coloured images each of size 32x32. The dataset is divided into 50,000 training images and 10,000 test images. The images here are taken in different lighting conditions as well as at different angles. Since images are colored, you can see that there are variations in colour itself of similar objects. CIFAR-10 is considered a good dataset to practice hyperparameter tuning. The classes here are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck.

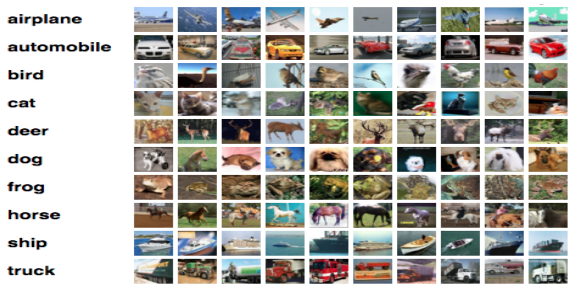

Fig. 4. CIFAR Dataset

B. Testing Results

- *MNIST*

  In order to evaluate the models, both the VGG16 and ResNet34 are firstly trained on **MNIST dataset**. Both the models are trained for 20 epochs on training data and simultaneously evaluated on the test data. The batch size was selected to be at 64. The *cross entropy loss* was selected as loss function for both the models. Also the images are resized to dimensions of (224,224) as those are the input dimensions needed for the VGG and ResNet model. To normalize the images, the values used are the global mean standard deviation of the MNIST dataset. The function used to perform the operation subtracts the mean and divides by the standard deviation of the floating point values in the range [0, 1]. Finally, in terms of optimizer, *Adam* optimizer was selected having the learning rate of 0.0001 to begin with.

  1) *VGG16 on MNIST*:
  Training the VGG16 model on the MNIST dataset gives exceptional results. The model is able to fit quite well as it can be seen that by the end of 20th epoch the model achieves 100% and 99% training and test accuracy respectively. Even the training loss and test loss seems to be decreasing with every epoch which is a good sign.

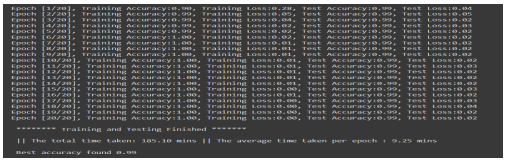

Fig. 5. VGG16 Training and Testing Process on MNIST

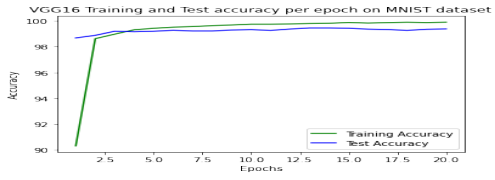

Fig. 6. VGG16 Training and Testing accuracy on MNIST

While looking at the accuracy and loss plot we can clearly see that test accuracy is increasing and the loss is decreasing with each epoch.

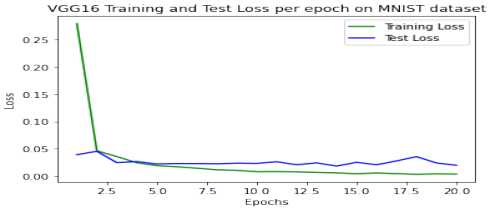

Fig. 7. VGG16 Training and Testing loss on MNIST

The confusion matrix of VGG16 on the dataset is shown in the following figure. The average accuracy for all the class obtained is 98.6%.

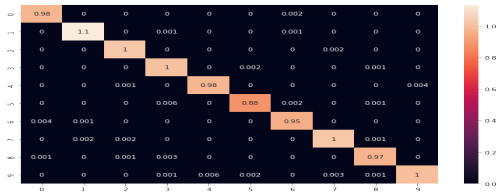

Fig. 8. VGG16 Confusion Matrix on MNIST

2) *ResNet34 on MNIST*:
When it comes to this model, the results are similar to how VGG16 performed on the MNIST dataset. On the training and test data, the model achieved 100% accuracy and surprisingly enough the model achieves 0% loss on the training set while on the test set the loss also seems to be decreasing.

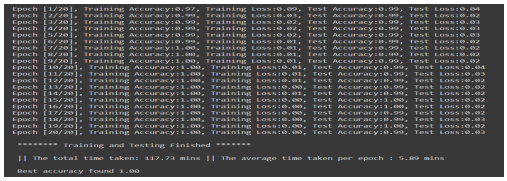

Fig. 9. ResNet34 Training and Testing process on MNIST

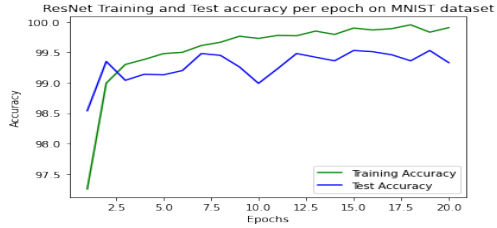

Fig. 10. ResNet34 Training and Test accuracy on MNIST

The test loss seems to be fluctuating but it is extremely minor so it does not make much difference.

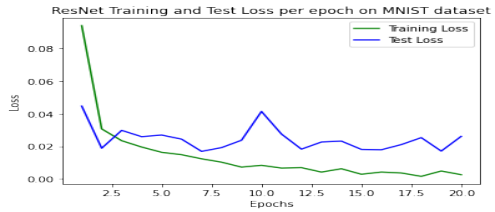

Fig. 11. ResNet34 Training and Test loss on MNIST

The confusion matrix of ResNet34 on the dataset is shown in the following figure. The average accuracy for all the class obtained is 98.5%.

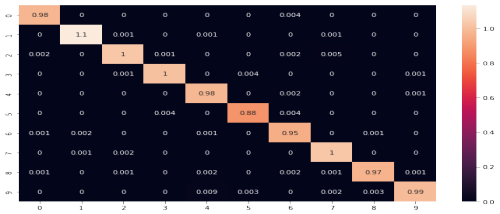

Fig. 12. ResNet34 Confusion Matrix on MNIST

- *CIFAR*

Same process has been followed for both the models on the **CIFAR10** dataset. Parameters with the same values have been taken into account. As the same models are used similar shape was taken for resizing the input images. Also, same loss function and optimizer has been used here. However, for the normalization process different values have been taken as different dataset is being used here. The only major difference for both the models has been that since CIFAR dataset has RGB channels the number of in_channels has been set to 3.

1) *VGG16 on CIFAR10*:
While training the model, it can be seen that it performed exceptionally well achieving training accuracy of 99% by the time 20th epoch finishes. However, on the test data it is only able to achieve 76% of accuracy and the loss also seems to be fluctuating instead of showing signs of receding. Therefore, one could say that there are chances of the model overfitting to the training data. Different parameters can be used or one can try a different loss function in order to overcome this issue.

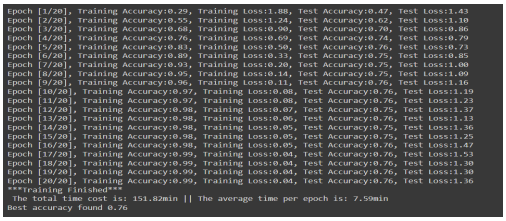

Fig. 13. VGG16 Training and Testing Process on CIFAR10

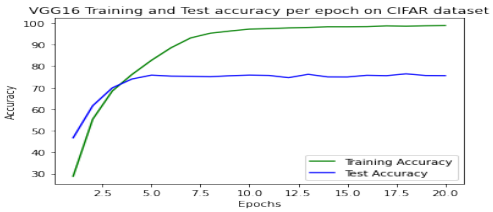

Fig. 14. VGG16 Training and Test Accuracy on CIFAR10

As from the loss plot we can see that the test loss is increasing indicating that there could be presence of overfitting.

Fig. 15. VGG16 Training and Test Loss on CIFAR10

The confusion matrix of VGG16 on the dataset is shown in the following figure. The average accuracy for all the class obtained is 75.4%.

Fig. 16. VGG16 Confusion Matrix on CIFAR10

2) *ResNet34 on CIFAR10*:
ResNet34 training accuracy seems to be increasing and loss seems to be decreasing however, the accuracy and loss on the test set remains almost constant with showing signs of very slight increase throughout the 20 epochs.

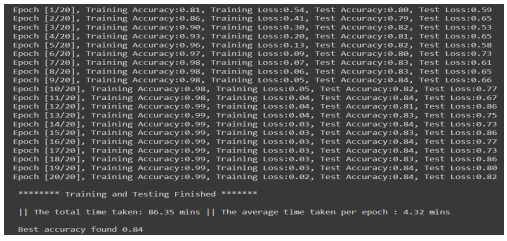

Fig. 17. ResNet34 Training and Testing Process on CIFAR10

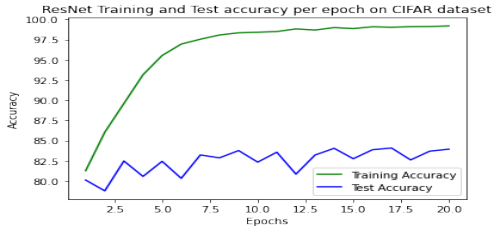

Fig. 18. ResNet34 Training and Test Accuracy on CIFAR10

The results on the test set here in case of ResNet also indicate that there is a chance of overfitting of the model on the dataset.

Fig. 19. ResNet34 Training and Test Loss on CIFAR10

The confusion matrix of ResNet34 on the dataset is shown in the following figure. The average accuracy for all the class obtained is 84.1%.

Fig. 20. ResNet34 Confusion Matrix on CIFAR10

C. Model Comparison

- *VGG16 vs ResNet34 on MNIST*

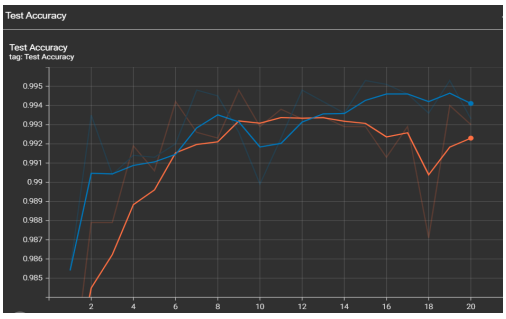

Fig. 21. ResNet34 vs VGG16 Test Accuracy on MNIST

As we can see that in both the situations RestNet achieves slightly higher accuracy and slightly lower loss.

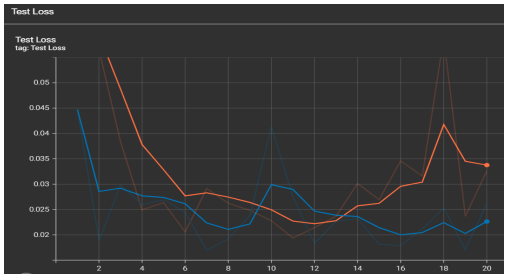

Fig. 22. ResNet34 vs VGG16 Test Loss on MNIST

- *VGG16 vs ResNet34 on CIFAR10*

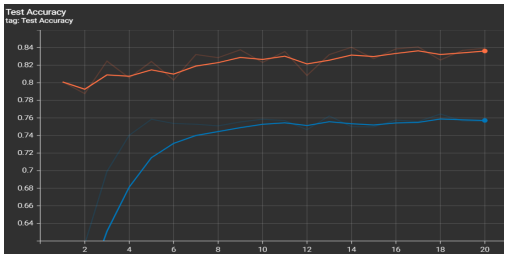

Fig. 23. ResNet34 vs VGG16 Test Accuracy on CIFAR

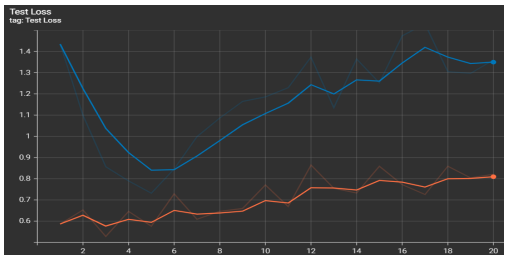

Fig. 24. ResNet34 vs VGG16 Test Loss on CIFAR

In the case of CIFAR dataset, the ResNet34 model is overfitting, thus showing that the test loss is increasing (blue line).

V. Conclusion

CIFAR dataset is slower to converge as compared to MNIST dataset because it contains RGB images which makes it more complex. Increasing the layers does not necessarily mean that the classification tasks will be done more effectively which ResNet has proved to be true. You need some mechanism that is able to extract features more effectively and efficiently. We can improve the accuracy by using different activation functions. If the model is overfitting, learning rate can be further reduced to tackle this. Also, as can be seen from the run time results, ResNet takes much less time and is computationally lighter as compared to VGG which is in accordance to the original paper published. To further evaluate the performance of the models we can perform data augmentation or we can use much complex and larger datasets. ResNet seems to be the optimal choice for the image classification tasks, atleast till the next coming years.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Alex Krizhevsky and Geoff Hinton. Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 40(7):1–9, 2010.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[4] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[5] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.