Deeper Network for Image Classification

1st Shuaihan Xu
EECS
Queen Mary, University of London
UK, London
shuaihan.xu@gmail.com
https://orcid.org/0000-0001-7707-636X

***Abstract*—In this work, 8 different models will be tested on 3 different datasets (MNIST, CIFAR10, CIFAR100) to evaluate their performance. There are three kinds of model among 8 different models, including neural network, VGG, ResNet.**

***Index Terms*—MNIST, VGG, ResNet, Neural network**

I. Introduction

Image classification means assigning the label to input image from fixed set of categories. In image classification domain, convolutional neural network (CNN) is the most common and popular approach. CNN has multiple layers namely convolutional layer, pooling layer and fully connected layer. The main objective of convolutional layer is to obtain the features of an image by sliding smaller matrix (kernel or filter) over the entire image and generate the feature maps. The pooling layer used to retain the most important aspect by reducing the feature maps. Fully connected layer interconnect every neuron in the layer to the neurons from the previous and next layer, to take the matrix inputs from the previous layers and flatten it to pass on to the output layer, which will make the prediction.

The aim of this work is to analyse and evaluate different types of neural network. The evaluation also includes effectiveness. In this work, the effectiveness will be judged by time. All the models will be trained and tested based on MNIST datasets. Further evaluation will be carried on CIFAR-10 and CIFAR-100 datasets.

II. Related Work

Recognition of handwritten digits was one of the first application where CNN architecture was successfully implemented. Since the creation of CNN, there has been continuous improvement in networks with the innovation of new layers and involvement of different computer vision techniques.

In 2012, the AlexNet[3], showed excellent performance on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[5]. It showed that a deep convolutional neural network has a high potential in dealing with such problem. The success of AlexNet has become the inspiration of different CNN model such as VGG[7], GoogleNet[8], ResNet[1] in the following years.

Among these works, VGG showed an architecture which combines a set of convolutional layers and fully connect layers. If we name such architecture as VGG-Like architecture, then ResNet is an improvement on VGG-Like architecture. It uses residual block to make sure the feature won't lose. At the same, the residual block helps the network to converge more efficiently.

Few of the researchers have shown a comparison between the human subject and a trained network's detection abilities on image datasets. In some cases, the prediction accuracy of model is even better than human.

III. Model Description

In this paper, I use various deeper networks for evaluating the effectiveness of deeper CNN models for image classification on MNIST.

A. Model Architecture

(I) VGG

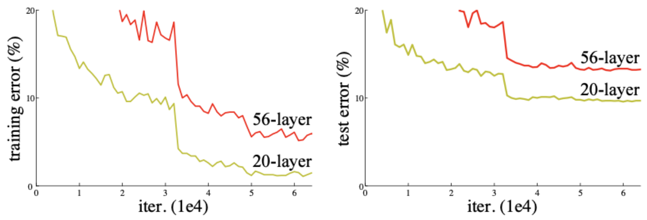
Fig. 1. VGG-Architecture

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

In the original paper[7], it showed 5 different kinds of VGG network architecture. According to the depth of model, they can be identified as VGG-11, VGG-13, VGG-16, VGG-19. All these networks have the same architecture, which is each convolution layer followed by a ReLU activation layer and each block is followed by a maxpool layer,but with different layers. The idea behind VGG is combine a set of convolutional neural networks and a set of fully connect layers.

(II) ResNet
As previous paper showed that accuracy can benefit from increasing the depth of network. But, does it mean that as long as the depth increased, the performance would be better? From discussion[1], the performance of network will not be better but worse by increasing the depth of the network. Such phenomenon is called degeneration.

Fig. 2. Degeneration



The graphs above showed that a 56-layers network has higher training loss and test error compared to a 20-layers network.
Basically, ResNet can be seen as an improved network of VGG. The idea of ResNet is to make an identical mapping[6] between feature and deep layers. Thus, it would feed previous layer's output to deeper layer's input. The structure of Basic Block and ResNet Architecture are shown as below.

Fig. 3. Basic Block

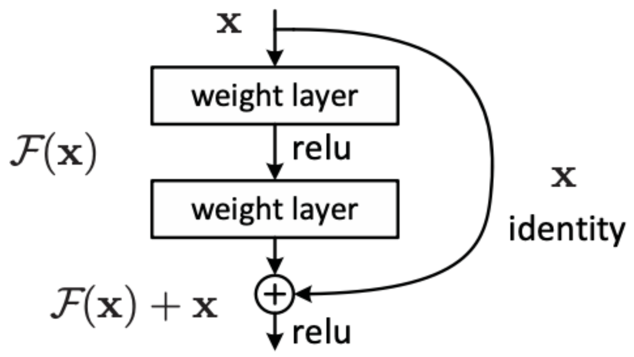
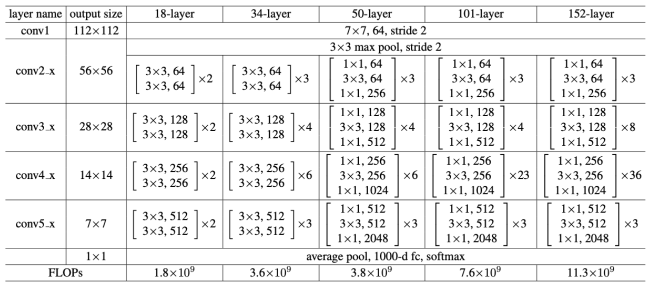

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 4. ResNet Architecture



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

(III) Neural Network
To make a comparison between different network. I made a model which doesn't contain convolutional layers for classification.
This model start with a flatten layer, it converts input into a single dimension vector and pass it to next linear layer. Each linear layer is followed by ReLU activation layer. A LogSoftmax layer is used to make the final decision for the category of input.
In this work, I would name it non-convolutional(NC) in order to distinguish from others.

IV. Experiments

A. Datasets

Three different datasets are introduced for training and evaluating models. They are MNIST datasets, CIFAR-10 and CIFAR-100 datasets.

The MNIST database [4] of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size- normalized and centred in a fixed-size image.

Fig. 5. MNIST Datasets

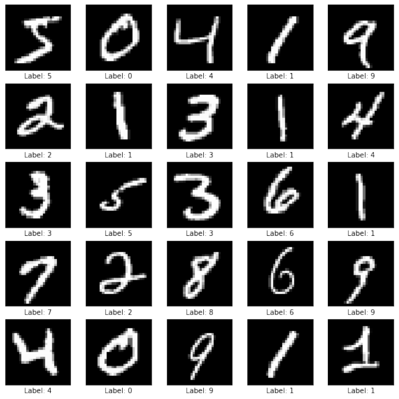


Fig. 6. MNIST Samples

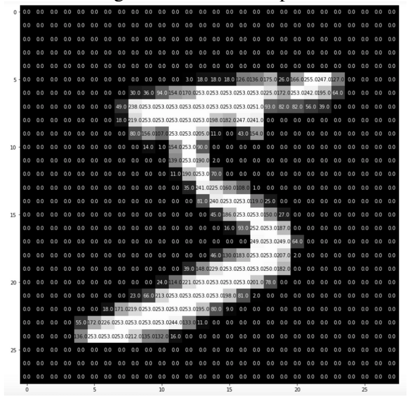


Fig. 7. MNIST Single Input

To further evaluate the performance of model, another two new datasets will be introduced. The CIFAR-10[2] is a collection of images that are commonly used to train machine learning and computer vision algorithms. It contains 60,000 of 32x32 colour images in 10 different classes.

CIFAR-100 is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses.

Fig. 8. CIFAR Datasets

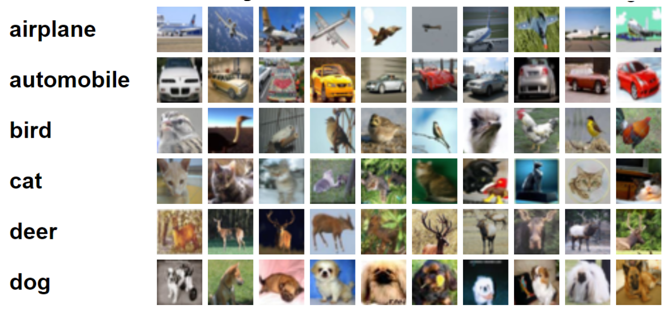


Fig. 9. CIFAR-10

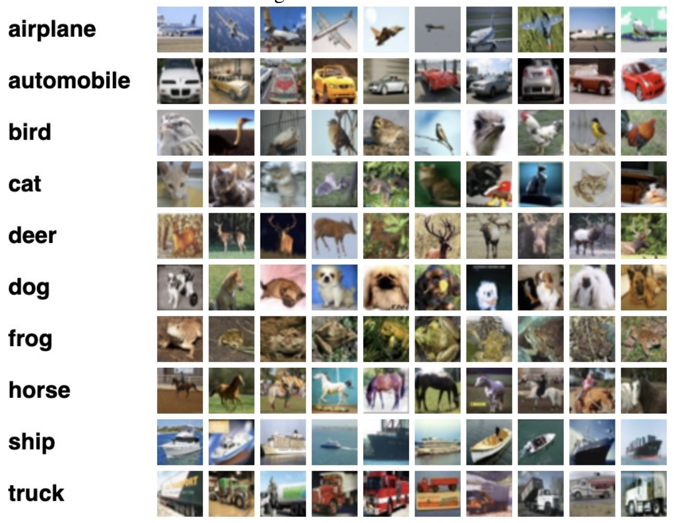


Fig. 10. CIFAR-100

B. Experiments Method

All experiments run on the same hardware and same environment. Summary of the environment is shown as following.

- CUDA version: 10.2
- Python version: 3.6.8
- Pytorch version: 1.7.1

According to the agreement of Google Colab. The arrangement of GPU resource is not guaranteed, which means it is not possible to figure out whether all the experiments run on the same GPU device or not. Thus it is not feasible to measure the efficiency by time. However, it might still be able to compare the efficiency between different models by trend of convergence with the same parameter. Parameters for all experiments will be the same and shown as following.

- Batch size: 100
- Learning rate: 0.001
- Epoch: 5
- Loss function: Cross entropy loss
- Optimizer: Adam

Consider MNIST is a relatively easy dataset, the epoch here is set for 5, since more epochs will not cause any difference. In the further evaluation, the epoch will be set as 20 to train the model.

As previous introduced, image classification is to assign a label to an input image. Take example IV-B, the label of this image is 7. If the model give the output as 7, then it is considered as success case. However, if a model is not accurate enough, it may produce an output which is different from label. In this case, the output of model is 9 thus it is considered as failure case.

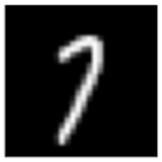


Fig. 11. label : 7

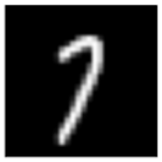


Fig. 12. label : 9

C. Testing Result

TABLE I
TEST RESULTS ON MNIST

	Train Accuracy(%)	Train Loss	Test Accuracy(%)	Test Loss
VGG11	100	0.0035	99	**0.0064**
VGG13	100	0.0029	99	0.0215
VGG16	98	0.0545	99	0.3462
VGG19	99	0.0515	99	0.1396
ResNet18	98	0.0648	99	**0.0158**
ResNet34	100	0.0048	99	0.0855
ResNet50	99	0.0220	98	0.0658
NC	97	0.0968	96	0.0673

From I, it can be fond that all the models have good performance on MNIST datasets. It is hard to tell the differences between them. The performance of non-convolutional network worth discussing. Its ability to abstract features suppose to be weaker than other network. Even though, it still got good performance as it's accuracy reached 97% and it has the second lowest test loss. Such data shows that in MNIST dataset, the convolutional layers are not really leveraged.

As it is discussed previously, it is not feasible to measure the efficiency through time but the trend of convergence. From the following comparison, we may find that the increasing depth of model may reduce the efficiency of network.

Here are testing results graph of different vgg models with different depths. The graph on the left side shows the accuracy of both training and test during. Right side graph shows the loss of both training and test.

From the trend of curve, ResNet18 has the highest efficiency as it has the most sharp curve. VGG19 reached the same

Fig. 13. VGG Model Evaluation on MNIST

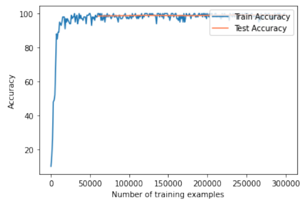


Fig. 14. V11-A

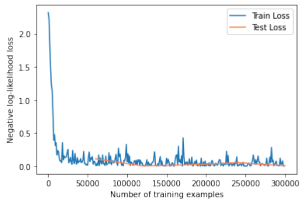


Fig. 15. V11-L

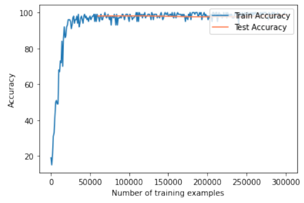


Fig. 16. V13-A

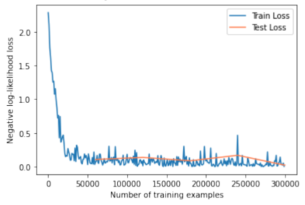


Fig. 17. V13-L

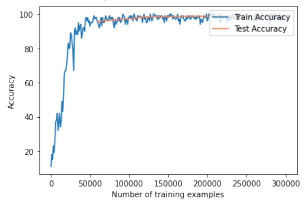


Fig. 18. V16-A

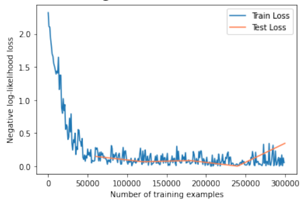


Fig. 19. V16-L

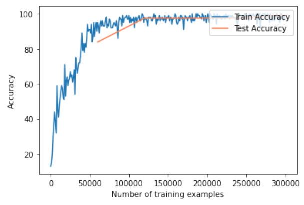


Fig. 20. V19-A

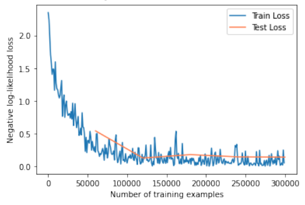


Fig. 21. V19-L

accuracy at final stage but at early stage its performance is not as good as VGG11 or ResNet18. The neural network model's efficiency is almost at the same level as VGG11.

D. Further Evaluation

For further evaluation, the model will be applied to CIFAR-10 and CIFAR-100 datasets. They tested models' performance under a higher complexity input and when the situation which the number of categories increased.

In the table above, the highest test accuracy and lowest test loss are marked by each model in each datasets. VGG13 has the best performance among all the models with the highest test accuracy and lowest test loss at the same time. It might not make sense as it is commonly considered that the performance benefits from increasing the depth.

There are two phenomenons may give some clue and worth discussing.

The graph of ResNet18 shows that it's test loss increased as the training loss reduce. This means ResNet18 started to over-fit in this case. The over-fit could make a model better fit the training data but worse make a prediction on test data.

Fig. 22. ResNet Model Evaluation on MNIST

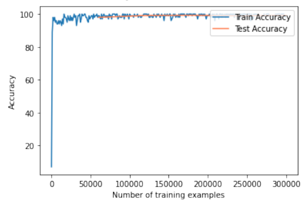


Fig. 23. R18-A

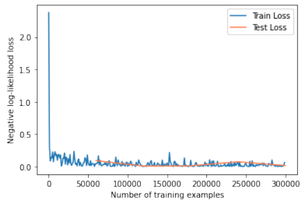


Fig. 24. R18-L

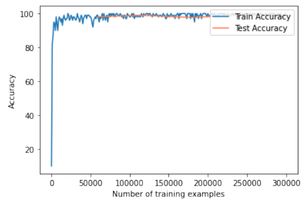


Fig. 25. R34-A

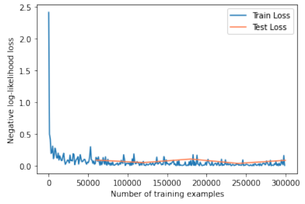


Fig. 26. R34-L

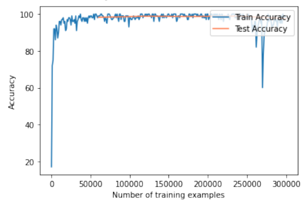


Fig. 27. R50-A

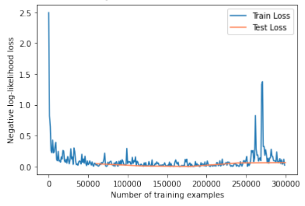


Fig. 28. R50-L

Fig. 29. NC model Evaluation on MNIST

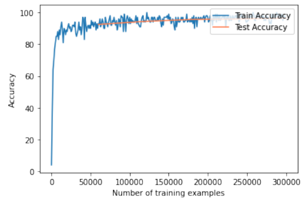


Fig. 30. NC-A

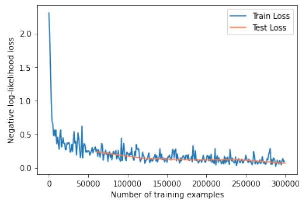


Fig. 31. NC-L

TABLE II
EVALUATION ON CIFAR DATASET(ACCURACY)

	Train Accuracy(%)	Test Accuracy(%)
VGG11-CIFAR10	99	81
VGG13-CIFAR10	97	**86**
VGG16-CIFAR10	92	77
VGG19-CIFAR10	91	84
ResNet18-CIFAR10	99	79
ResNet34-CIFAR10	91	**81**
ResNet50-CIFAR10	85	69
NC-CIFAR10	77	52
VGG11-CIFAR100	85	50
VGG13-CIFAR100	75	**53**
VGG16-CIFAR100	63	45
VGG19-CIFAR100	40	34
ResNet18-CIFAR100	97	**47**
ResNet34-CIFAR100	97	47
ResNet50-CIFAR100	94	47
NC-CIFAR100	48	23

TABLE III
EVALUATION ON CIFAR DATASET(LOSS)

	Train Loss	Test Loss
VGG11-CIFAR10	0.0232	1.2401
VGG13-CIFAR10	0.0728	**0.5925**
VGG16-CIFAR10	0.2716	1.0274
VGG19-CIFAR10	0.2802	0.4772
ResNet18-CIFAR10	0.0172	**1.1264**
ResNet34-CIFAR10	0.0901	1.2319
ResNet50-CIFAR10	0.4317	1.3766
NC-CIFAR10	0.7920	1.7517
VGG11-CIFAR100	0.4543	2.6239
VGG13-CIFAR100	0.7504	**1.9698**
VGG16-CIFAR100	1.2922	2.2797
VGG19-CIFAR100	2.0645	2.4238
ResNet18-CIFAR100	0.0960	**3.3895**
ResNet34-CIFAR100	0.1096	3.5681
ResNet50-CIFAR100	0.1851	3.9801
NC-CIFAR100	2.1397	3.5169

Over-fitting is a fundamental issue in supervised machine learning which prevents us from perfectly generalizing the models to well fit observed data on training data, as well as unseen data on testing set. Because of the presence of noise, the limited size of training set, and the complexity of classifiers, over-fitting happens.

The causes of this phenomenon might be complicated. Generally, we can categorize them into three kinds:

1) Noise learning on the training set: when the training set is too small, or has less representative data or too many noises. This situation makes the noises have great chances to be learned, and later act as a basis of predictions. So, a well-functioning algorithm should be able to distinguish representative data from noises
2) Hypothesis complexity: the trade-off in complexity, a key concept in statistic and machining learning, is a compromise between Variance and Bias. It refers to a balance between accuracy and consistency. When the algorithms have too many hypotheses (too many inputs), the model becomes more accurate on average with lower consistency. This situation means that the models can be drastically different on different datasets.
3) Multiple comparisons procedures which are ubiquitous in induction algorithms, as well as in other Artificial Intelligence (AI) algorithms.s

ResNet50 shows almost the same situation and over fit has caused its test accuracy decreased. Besides, it's training loss increased all of a sudden. This is likely to be caused by improper learning rate. In this work, as we need to control the variable, all the learning rate is set to be the same fixed value. For the best performance of ResNet50, we might need to choose another learning rate for it.

The Last part of test is based on CIFAR-100. In this part we may see some similar experiments outcomes as CIFAR-10.

One point worth noting was that, it is obvious when dealing with larger number of labels. The performance of deep CNNs is significantly better than NC.

The graph of the experiments are included in the appendix.

Fig. 32. VGG Model Evaluation on CIFAR10

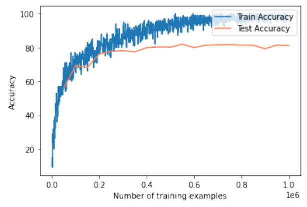


Fig. 33. V11-A

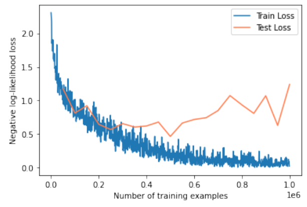


Fig. 34. V11-L

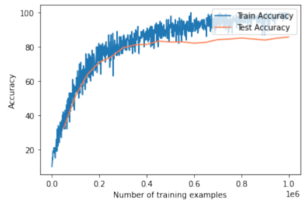


Fig. 35. V13-A

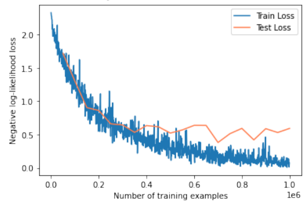


Fig. 36. V13-L

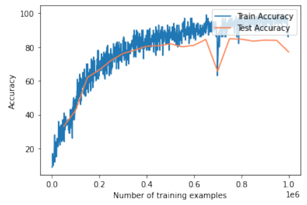


Fig. 37. V16-A

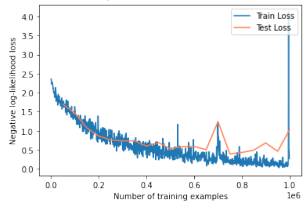


Fig. 38. V16-L

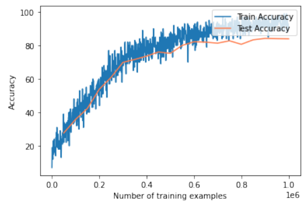


Fig. 39. V19-A

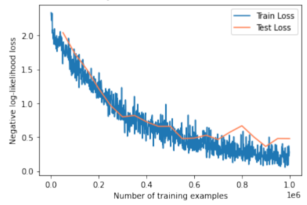


Fig. 40. V19-L

V. CONCLUSION

In this work, performance of different convolutional based deep neural network is tested. Besides, it introduced an easy netowrk, which without convolutional layers, to run on MNIST and achieved great outcome.

We may have the following conclusions from this work.

- The performance of model can not only benefit from increasing the depth of model but also different architecture.
- For a network, an improper learning rate may cause a sudden increase of training loss.
- It might be improper to apply a complicated network to a low complexity datasets.
- Convolutional neural network has better performance with a larger number of labels.

Some more explorations of this work in the future can be carried in following aspects.

- The effectiveness of types and parameters of optimizer
- The relationship between learning rate and model complexity

Fig. 41. ResNet Model Evaluation on CIFAR10

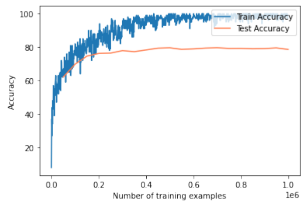


Fig. 42. R18-A

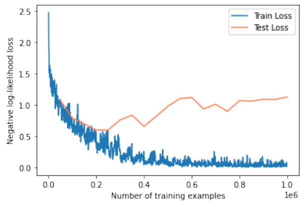


Fig. 43. R18-L

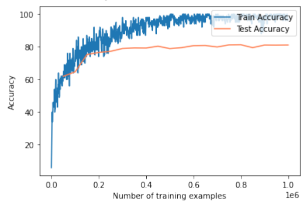


Fig. 44. R34-A

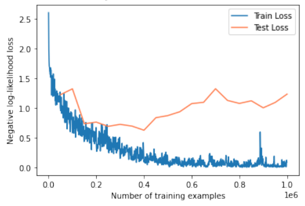


Fig. 45. R34-L

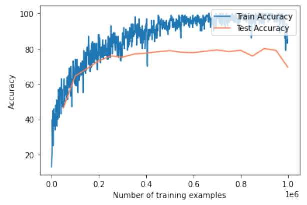


Fig. 46. R50-A

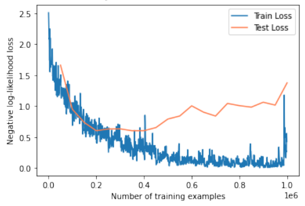


Fig. 47. R50-L

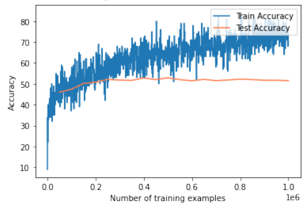


Fig. 48. NC-A

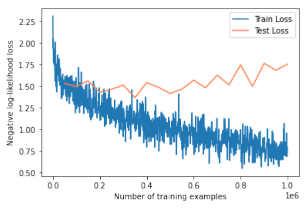


Fig. 49. NC-L

Fig. 50. VGG Model Evaluation on CIFAR10

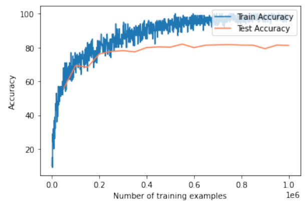


Fig. 51. V11-A

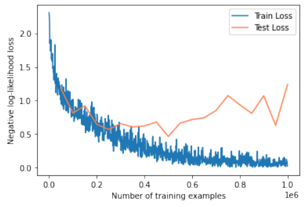


Fig. 52. V11-L

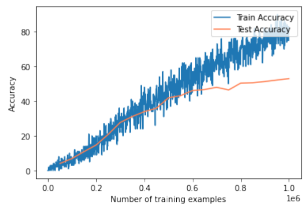


Fig. 53. V13-A

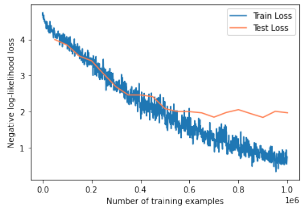


Fig. 54. V13-L

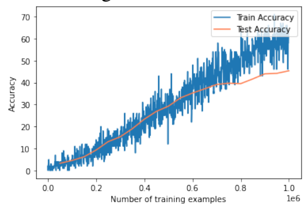


Fig. 55. V16-A

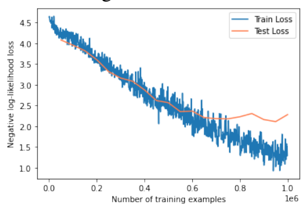


Fig. 56. V16-L

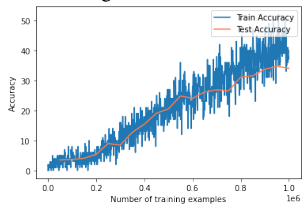


Fig. 57. V19-A

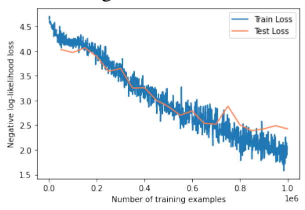


Fig. 58. V19-L

VI. Appendix

References

[1] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[2] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009).

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[4] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[5] Olga Russakovsky et al. “Imagenet large scale visual recognition challenge”. In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[6] Mark Sandler et al. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. 2019. arXiv: 1801.04381 [cs.CV].

[7] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[8] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.