Deep Learning and Computer Vision

Coursework 3

Guanzhen Wu
161189284
g.wu@se16.qmul.ac.uk
EECS School

***Abstract*—Deep Convolutional Neural Networks has attracted more and more attention since it continues to achieve dazzling results in the ImageNet Large Scale Visual Recognition Challenge. The author takes VGG and ResNet as an example, and analyzes from the perspectives of model architecture, model parameters, computational complexity and model robustness, and horizontally compares the advantages and disadvantages of the two models. Using MNIST and CIFAR10 data sets for testing and evaluation, it proves the strong image feature extraction ability of VGG and the low complexity of ResNet. Finally, the author gives future research directions.**

Keywords—Deep CNN, VGG, ResNet, model evaluation, MNIST, CIFAR10

I. INTRODUCTION

Deep learning have been widely used since its proposed, especially in the field of computer vision. In 2006, Hinton et al.[1] published a paper in Science, whose main ideas were: 1) artificial neural networks with multiple hidden layers have excellent feature learning ability; 2) layer-wise pre-training can be used to effectively overcome the difficulties in training deep neural networks. This led to the study of Deep Learning. Among them, Convolutional neural network (CNN) is the most popular model in deep learning due to its excellent feature extraction and its shift, scale, and distortion invariance through local receptive fields, shared weights, and sub-sampling. Numerous CNN-based deep learning models have come out on top in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), such as VGG, ResNet, GoogleNet and etc. The new deep CNN networks are a significant improvement over shallow artificial neural networks for handwritten digits or other image recognition tasks. This article focuses on the VGG and ResNet network models as examples, conducts experiments on the MNIST handwriting and CIFAR10 datasets to investigate the impact of different deep network model architectures and different recognition tasks.

II. CRITICAL ANALYSIS

There are 3 factors that affect the performance of CNN: the number of layers, the number of feature maps and the network architecture. It has been found that 1) increasing the depth of the network can improve accuracy, 2) increasing the number of feature maps can also improve accuracy, and 3) adding a convolutional layer can achieve a higher accuracy than adding a fully connected layer. The deeper the CNN structure and the greater the number of feature maps, the larger the feature space can be represented and the greater the learning capability of the network. However, it will also make the network more complex and more likely to be overfitted. Therefore, the network depth, the number of feature maps, the size of the convolution kernel and the step size of the convolution sliding should be selected appropriately in order to obtain a good model and reduce the training time.

With the development of hardware, the impact of computational complexity is gradually reducing, making people pay more attention to the network architecture. Based on this, VGG and ResNet were proposed. The VGG model is comprehensively evaluated on networks with increasing depth using an architecture with very small (3×3) convolutional filters, which shows that significant improvements over state-of-the-art configurations can be achieved by pushing the depth to 16-19 weights . This is also the basis for the VGG model proposed by Karen Simonyan, which participated in the 2014 ImageNet Challenge and won 1st and 2nd place in the localization and classification tracks respectively. [2]

Kaiming He presented a residual learning framework to ease the training of networks that are substantially deeper than those used previously. They explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. Comprehensive empirical evidence was provided to show that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset, the original article evaluated residual nets up to 152 layers deep - eight times deeper than VGG nets, but still with lower complexity. The set of these residual nets achieved an error of 3.57% on the ImageNet test set. This result won first place in the ILSVRC 2015 classification task[3].

These efforts have achieved new current best performances on a wide range of classification and regression tasks. In contrast, although the history of these methods goes back many years, the theoretical understanding of the way in which these systems achieve excellent results is still lagging behind. Indeed, many of the current results in computer vision use CNNs as black boxes, an approach that works, but the reasons

why it works are so obscure that it seriously fails to meet the requirements of scientific research. In particular：(1) In terms of what is being learned, such as convolutional kernels, what exactly is being learned? (2) In terms of architectural design, such as the number of layers, the number of kernels, the pooling strategy, and the choice of nonlinearities, why are certain choices superior to others? The answers to these questions will not only improve our scientific understanding of CNNs but also their usefulness.

Furthermore, current approaches to implementing CNNs require large amounts of training data and design decisions have a significant impact on the performance of the results. A deeper theoretical understanding should alleviate the reliance on data-driven design. Although empirical studies have investigated how the implemented networks operate, so far these results have largely been limited to visualization of internal processing with the aim of understanding what is happening in the different layers of a CNN. [4]

III. Model Description

1) VGG

The network structure of VGG is shown in Table 1. It contains 6 VGG models with different layers in the original article. According to the convolution kernel stack, different convolution layer operations are implemented to form VGG models with different computing capabilities. What is implemented in this paper is a VGG network configured as D, with a total of 16 convolutional/fully connected layers. The size of the convolution kernel of VGG is 3*3, instead of the 11*11 convolution kernel like AlexNet. Such a structure can obtain better recognition results.

Table 1 ConvNet configurations [2]

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					

2) ResNet

As the depth of network deepens, the model generates a large number of parameters while having more feature maps. These parameters, in the process of forward or backward propagation, may easily cause the problem of gradient vanishing or gradient explosion, which is why the deep neural network cannot add layers simply. For this issue, the ResNet model proposed by Kaiming He solves it well. As Figure 1 shown, while obtaining a new layer of output F(x), the input layer X is added, which makes the network not focus too much on the pervious output, but takes into account the original data. The new output will be F(x) + X which shows the principle of Residual learning.

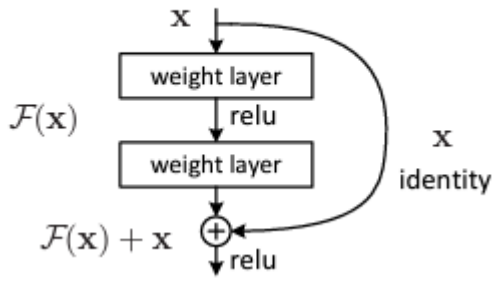

Figure 1 Residual learning: a building block [3]

This article implements the model structure of resnet18, as shown in Table 2 which is an improved version of ResNet. The difference between is that, in the bottleneck blocks which requires down sampling, the original one has stride = 2 in the first 1x1 convolution, whereas our model has stride = 2 in the 3x3 convolution.

According to the pytorch document, this difference makes ResNet slightly more accurate (~0.5% top1) than before but comes with a small performance drawback (~5% imgs/sec) [5].

Table 2 Architectures for ImageNet

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

IV. Experiments

1) Datasets

a) MNIST

MNIST dataset of handwritten digits 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 centered in a fixed-size image. [6] The dataset examples show as Figure 2. The classes are divided into 0 to 9. There are 6000 handwritten images in each class. The pixels of the images are 28*28 in grey scale, which converted to one-hot is 784.

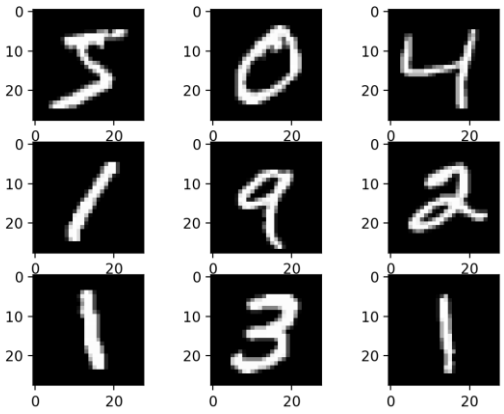

Figure 2 MNIST digits samples

b) CIFAR10

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The ten classes of CIFAR 10 is airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The samples of CIFAR 10 dataset is shown as Figure 3. Each sample of the class is a 32*32 RGB image [7].

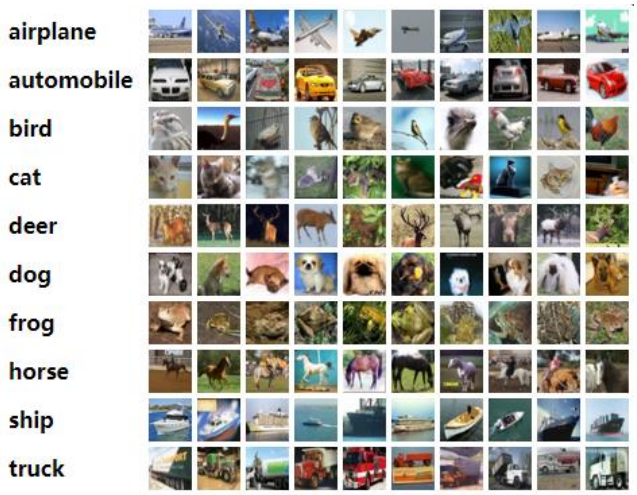

Figure 3 CIFAR 10 samples

2) Testing result

The results of the VGG16 model trained using MNIST handwriting are shown in Figure 4. The loss of the model decreases very quickly and significantly during the first two epochs of training. After the first epoch, the accuracy of the model reaches 27.9%, while the accuracy of the model reaches 94.4% at the second epoch. The loss of the model decreased from 1.93 to 0.19. After the second epoch, the model converged and finally the training loss stabilized at about 0.03, with an accuracy of 99.2%.

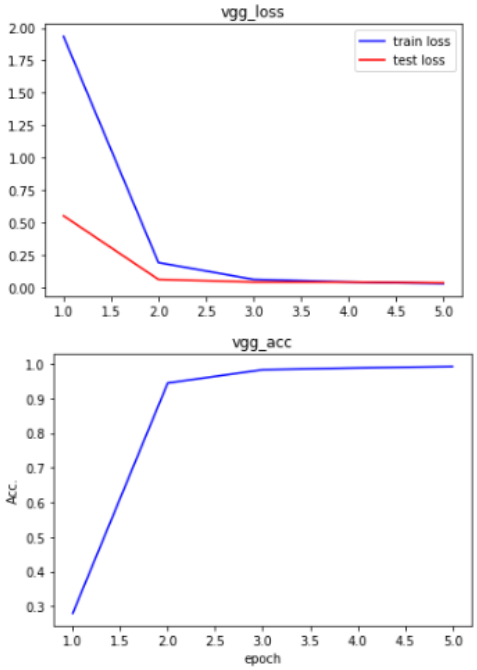

Figure 4 VGG-MNIST loss&acc.

The confusion matrix can be used to find out what the model has for the classes that are easy to find errors and calculate the recall rate and F1 Score. As shown in Figure 5, some numbers that are similar in morphological structure are easy to be classified incorrectly, such as 4 and 9, 3 and 5 and so on.

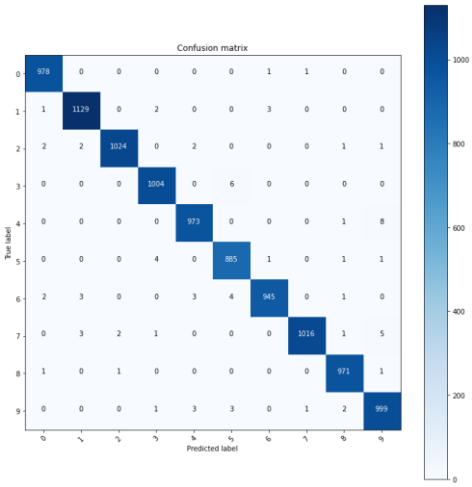

Figure 5 VGG-MNIST Confusion matrix

In Figure 6, the misclassified images and labels are printed out, the numbers on the left represent the predicted labels, and the numbers on the right represent the ground truth. It can be seen

that different handwriting methods can easily lead to misclassification by the machine.

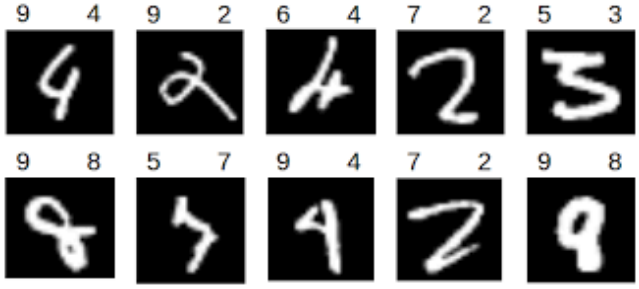

Figure 6 Misclassification samples

Overall, the model can converge within 3 epochs and achieve an accuracy rate as high as 98%, indicating that the network structure of VGG16 can handle the handwritten dataset very well.

The test results of ResNet are similar to those of the VGG model. As can be seen from Figure 7, the model achieves very low loss and 96% training accuracy in the first epoch. The learning ability of the model is stronger than that of VGG. A great result is achieved with very little training. The confusion matrix in Figure 8 is similar to that of VGG, and morphologically similar features can also lead to misclassification.

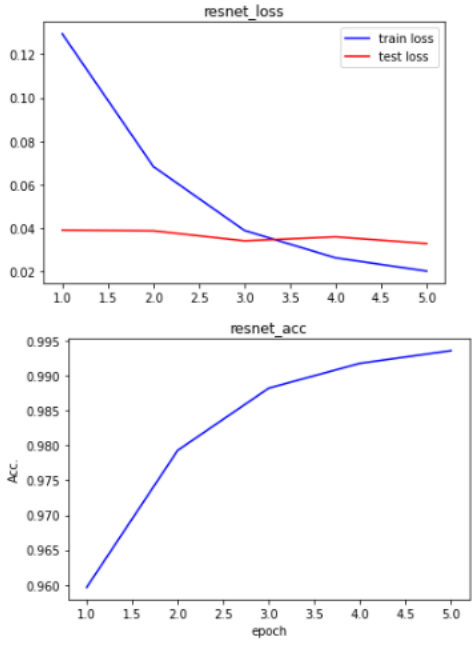

Figure 7 ResNet-MNIST loss&acc.

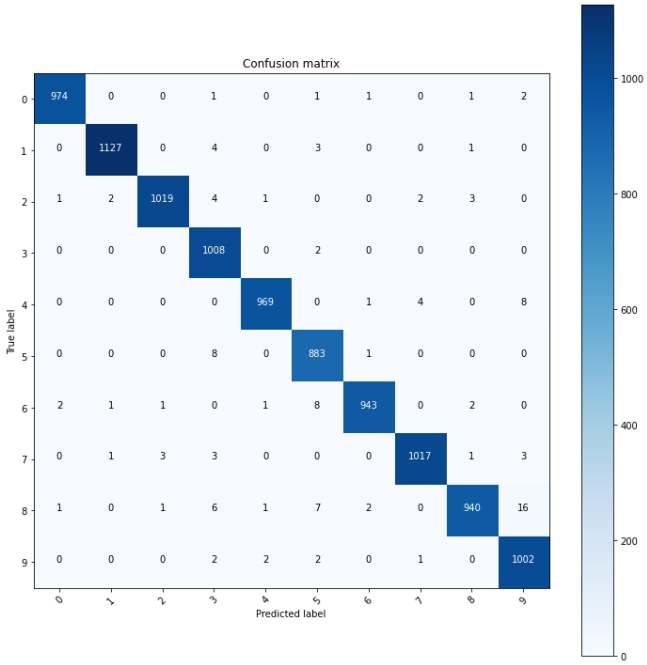

Figure 8 ResNet-MNIST Confusion matrix

Based on the comparison in Table 3, ResNet's training time is nearly a quarter of VGG's, and it has higher accuracy and lower loss. it can be concluded that ResNet's model is more lightweight, but the robustness of the two models cannot be distinguished well due to the similarity of the loss and accuracy and the simplicity of the digital handwriting structure, so a more complex dataset is needed for further evaluation.

Table 3 Test results of VGG and ResNet in MNIST

Model	Loss	Top1 Acc.	Top5 Acc.	Training time
VGG16	0.0333	99.0%	99.9%	10’40’’
ResNet18	0.0329	99.2%	1	2’55’’

3) Further Evaluation

As the data from digital handwriting is too simple to evaluate the model metrics well, the CIFAR dataset, also with ten classes, was chosen for further evaluation. These ten classes have a more complex spatial structure as well as colour, rather than a grey-scale image as in MNIST. As can be seen in Figures 9 and 11, both models took more epochs to converge, and after same number of training batches as the MNIST dataset, the models were still far from converging. VGG and ResNet models both had a training loss around 1 and a training accuracy no higher than 60%. It was not until after 25 epochs that it gradually converged. The comparison between Figure 9 and Figure 11 shows that VGG converges faster than ResNet and eventually reaches a training loss of 0.25, while ResNet reaches a training loss of 0.6 and does not

converge after 25 long epochs. In terms of accuracy, the VGG model is also more accurate than ResNet, with 91% and 78% respectively.

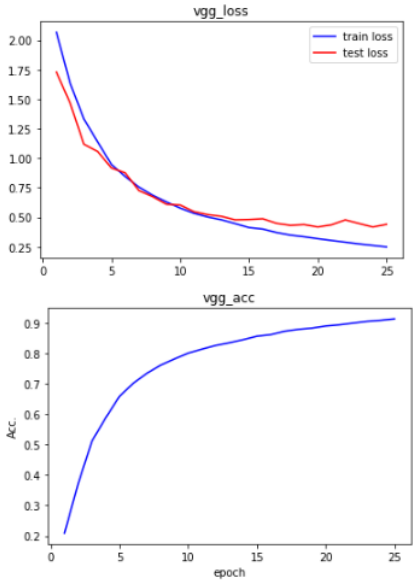

Figure 9 VGG-CIFAR10 loss&acc.

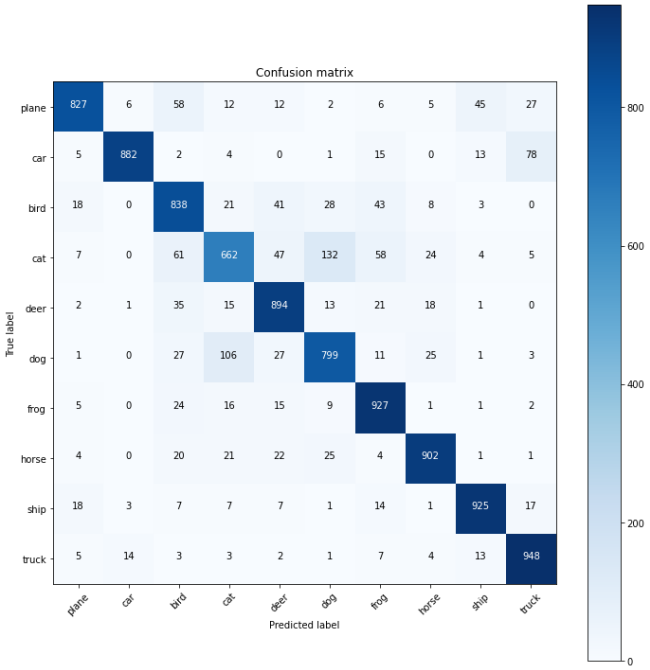

Figure 10 VGG-CIFAR10 Confusion matrix

The confusion matrices in Figures 10 and 12 also show that ResNet misclassifies more than VGG, as the non-diagonal colours in the confusion matrix of ResNet are darker than those of VGG. From Table 4, it is concluded that VGG outperforms ResNet in terms of loss as well as accuracy but takes more time to train than ResNet. This is related to the fact that VGG has more convolutional layers and more feature maps compared to ResNet. More model parameters allow for better recognition of complex shapes, but also result in more storage space and longer training time.

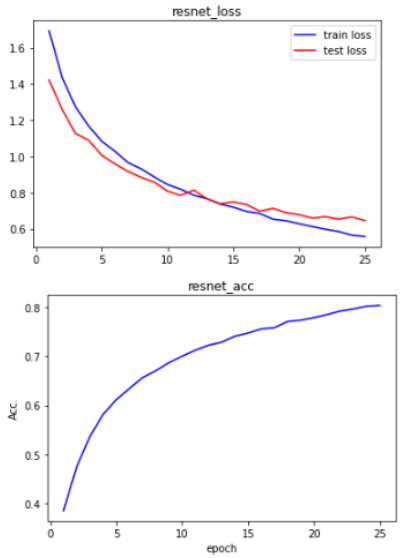

Figure 11 ResNet-CIFAR10 loss&acc.

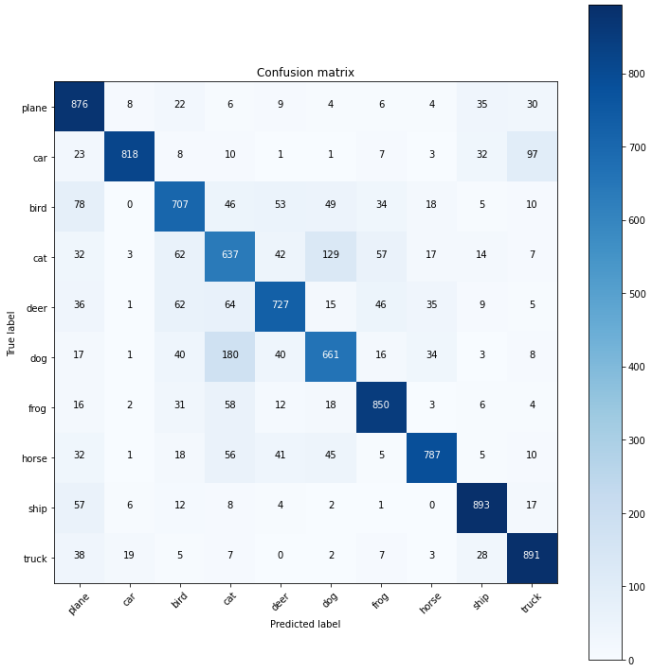

Figure 12 ResNet-CIFAR10 Confusion matrix

Table 4 Test result of VGG and ResNet in CIFAR10

Model	Loss	Top1 Acc.	Top5 Acc.	Training time
VGG16	0.4397	86.5%	99.3%	18’6’’
ResNet18	0.6462	78.5%	98.6%	10’4’’

V. CONCLUSION

This paper compares two different deep convolutional neural network models, VGG and ResNet, and presents an all-around comparison in terms of dataset complexity, model parameters, model architecture, training results and evaluation. It is demonstrated that the VGG model has more

convolutional kernels and parameters, which makes it easier to analyze and classify complex images, but also leads to a longer training time and a larger footprint due to this feature. In contrast, ResNet has a lighter network structure and can remember the properties of the original input better, taking less time to train in the same epoch. Future research will investigate how to visualize the training process of the two models and understand how their weights update during training.

REFERENCES

[1] Geoffrey Hinton, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science,2006,313(5786):504-507

[2] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[3] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[4] Hadji, I. and Wildes, R.P., 2018. What do we understand about convolutional networks?. arXiv preprint arXiv:1803.08834.

[5] https://catalog.ngc.nvidia.com/orgs/nvidia/resources/resnet_50_v1_5_for_pytorch

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.

[7] Learning Multiple Layers of Features from Tiny Images, Alex Krizhevsky, 2009.

Appendix:
(For more information, please see the evidence file)

```
VGG16__MNIST
start time: 2022-05-10 21:24:07.963220
---------- epoch 1 ----------
Finish Training 1 epoch, Loss: 1.935613, Acc.: 0.279883
Test Loss: 0.551407, Top1_Acc: 0.832200 , Top5_Acc: 0.986400
---------- epoch 2 ----------
Finish Training 2 epoch, Loss: 0.190141, Acc.: 0.944633
Test Loss: 0.060028, Top1_Acc: 0.982500 , Top5_Acc: 0.999200
---------- epoch 3 ----------
Finish Training 3 epoch, Loss: 0.060506, Acc.: 0.982817
Test Loss: 0.040221, Top1_Acc: 0.988600 , Top5_Acc: 0.999500
---------- epoch 4 ----------
Finish Training 4 epoch, Loss: 0.040922, Acc.: 0.988167
Test Loss: 0.039874, Top1_Acc: 0.988600 , Top5_Acc: 0.999600
---------- epoch 5 ----------
Finish Training 5 epoch, Loss: 0.027715, Acc.: 0.992200
Test Loss: 0.033347, Top1_Acc: 0.990100 , Top5_Acc: 0.999900
End time: 2022-05-10 21:34:47.521734
```

```
---------- epoch 18 ----------
Finish Training 18 epoch, Loss: 0.348576, Acc.: 0.880220
Test Loss: 0.431639, Top1_Acc: 0.854000 , Top5_Acc: 0.993700
---------- epoch 19 ----------
Finish Training 19 epoch, Loss: 0.334653, Acc.: 0.884460
Test Loss: 0.437796, Top1_Acc: 0.852800 , Top5_Acc: 0.992300
---------- epoch 20 ----------
Finish Training 20 epoch, Loss: 0.317953, Acc.: 0.891360
Test Loss: 0.417147, Top1_Acc: 0.863300 , Top5_Acc: 0.994100
---------- epoch 21 ----------
Finish Training 21 epoch, Loss: 0.302812, Acc.: 0.895520
Test Loss: 0.435785, Top1_Acc: 0.855600 , Top5_Acc: 0.992900
---------- epoch 22 ----------
Finish Training 22 epoch, Loss: 0.288269, Acc.: 0.901100
Test Loss: 0.476912, Top1_Acc: 0.847900 , Top5_Acc: 0.992900
---------- epoch 23 ----------
Finish Training 23 epoch, Loss: 0.273275, Acc.: 0.906660
Test Loss: 0.446430, Top1_Acc: 0.852800 , Top5_Acc: 0.993400
---------- epoch 24 ----------
Finish Training 24 epoch, Loss: 0.261046, Acc.: 0.910000
Test Loss: 0.417030, Top1_Acc: 0.869200 , Top5_Acc: 0.993500
---------- epoch 25 ----------
Finish Training 25 epoch, Loss: 0.248183, Acc.: 0.914820
Test Loss: 0.439679, Top1_Acc: 0.865200 , Top5_Acc: 0.993900
End time: 2022-05-12 19:16:58.268821
```

```
resnet__MNIST
start time: 2022-05-11 11:43:40.064923
---------- epoch 1 ----------
Finish Training 1 epoch, Loss: 0.134794, Acc.: 0.957950
Test Loss: 0.042557, Top1_Acc: 0.985900 , Top5_Acc: 1.000000
---------- epoch 2 ----------
Finish Training 2 epoch, Loss: 0.066932, Acc.: 0.978833
Test Loss: 0.046474, Top1_Acc: 0.985700 , Top5_Acc: 1.000000
---------- epoch 3 ----------
Finish Training 3 epoch, Loss: 0.036302, Acc.: 0.988583
Test Loss: 0.039360, Top1_Acc: 0.987800 , Top5_Acc: 0.999900
---------- epoch 4 ----------
Finish Training 4 epoch, Loss: 0.024770, Acc.: 0.992117
Test Loss: 0.029929, Top1_Acc: 0.990600 , Top5_Acc: 0.999900
---------- epoch 5 ----------
Finish Training 5 epoch, Loss: 0.019236, Acc.: 0.993817
Test Loss: 0.039289, Top1_Acc: 0.988200 , Top5_Acc: 0.999700
End time: 2022-05-11 11:46:35.971253
```

```
---------- epoch 12 ----------
Finish Training 12 epoch, Loss: 0.785911, Acc.: 0.722260
Test Loss: 0.813934, Top1_Acc: 0.716600 , Top5_Acc: 0.979200
---------- epoch 13 ----------
Finish Training 13 epoch, Loss: 0.767137, Acc.: 0.728460
Test Loss: 0.765834, Top1_Acc: 0.731000 , Top5_Acc: 0.980700
---------- epoch 14 ----------
Finish Training 14 epoch, Loss: 0.736950, Acc.: 0.740700
Test Loss: 0.738825, Top1_Acc: 0.740600 , Top5_Acc: 0.982100
---------- epoch 15 ----------
Finish Training 15 epoch, Loss: 0.719890, Acc.: 0.747040
Test Loss: 0.749275, Top1_Acc: 0.750600 , Top5_Acc: 0.978400
---------- epoch 16 ----------
Finish Training 16 epoch, Loss: 0.696388, Acc.: 0.755520
Test Loss: 0.734280, Top1_Acc: 0.755500 , Top5_Acc: 0.980200
---------- epoch 17 ----------
Finish Training 17 epoch, Loss: 0.685441, Acc.: 0.758060
Test Loss: 0.698052, Top1_Acc: 0.763300 , Top5_Acc: 0.984600
---------- epoch 18 ----------
Finish Training 18 epoch, Loss: 0.653687, Acc.: 0.770560
Test Loss: 0.713412, Top1_Acc: 0.755400 , Top5_Acc: 0.982500
---------- epoch 19 ----------
Finish Training 19 epoch, Loss: 0.645040, Acc.: 0.773240
Test Loss: 0.689965, Top1_Acc: 0.761100 , Top5_Acc: 0.983600
---------- epoch 20 ----------
Finish Training 20 epoch, Loss: 0.628349, Acc.: 0.778280
Test Loss: 0.678558, Top1_Acc: 0.767600 , Top5_Acc: 0.982200
---------- epoch 21 ----------
Finish Training 21 epoch, Loss: 0.613202, Acc.: 0.784780
Test Loss: 0.659631, Top1_Acc: 0.775800 , Top5_Acc: 0.984500
---------- epoch 22 ----------
Finish Training 22 epoch, Loss: 0.597814, Acc.: 0.791740
Test Loss: 0.667225, Top1_Acc: 0.775600 , Top5_Acc: 0.985300
---------- epoch 23 ----------
Finish Training 23 epoch, Loss: 0.585383, Acc.: 0.796080
Test Loss: 0.654250, Top1_Acc: 0.778600 , Top5_Acc: 0.983300
---------- epoch 24 ----------
Finish Training 24 epoch, Loss: 0.565759, Acc.: 0.801660
Test Loss: 0.666518, Top1_Acc: 0.778100 , Top5_Acc: 0.986100
---------- epoch 25 ----------
Finish Training 25 epoch, Loss: 0.558224, Acc.: 0.803360
Test Loss: 0.646179, Top1_Acc: 0.784700 , Top5_Acc: 0.985700
End time: 2022-05-12 19:31:41.627229
```