Deeper Networks for Image Classification

Zhuo Liu

April 2019

1 Introduction

Convolutional neural network (CNN) is one of the most mainstream neural networks. Compared with other neural networks, CNN has obvious superiority in image information processing. Specifically, it has two advantages:

- Reduce the amount of image data and computation.
- Preserving image features

According to Guido Montufar‘s research (1). The Relu nonlinear activation function is mathematically equivalent to a piecewise linear function. The more linear regions, the better performance of the neural network.

Based on this, in order to get better performance in neural network, in 2015, Oxford’s Visual Geometry Group proposed a deeper CNN model, which is called VGG. And in 2016, a deep recurrent network (ResNet) which has performance better than VGG was also found. In the next part, we will introduce the VGG and ResNet model.

2 Critical Analysis / Related Work

2.1 Main Trend

VGGt(2) using 16 hidden layers proves that increasing the depth of the neural network can improve the final performance of the network to some extent. And in ResNet model a year later, the number of hidden layers is further added to 152, and this improvement also results in the better performance. Meanwhile, as the increasing number of hidden layers, the processing accuracy on ImageNet is proved to be higher and higher. Therefore, in image processing, the trend of improving neural network is to add more and more hidden layers to achieve higher performance requirements.

2.2 Key Ideas

The VGG16 model contains 16 hidden layers (13 convolutional layers and 3 fully connected layers). The structure of VGG19 is similar to the VGG16, but it has 19 hidden layers.

n VGG, three 3x3 convolution kernels are used instead of using 7x7 convolution kernels, and two 3x3 convolution kernels are used instead of using 5x5 convolution kernels. The main purpose of this change is to improve the depth of the network and the performance of neural network to some extent under the condition of ensuring the same perception field.

Deep residual network (ResNet) is one of the more classic CNN networks after VGG, refreshing the history of CNN model on ImageNet. It’s even more radical, in which 152 hidden layers are added to the network. Theoretically, when the number of layers is increased, the network can extract more complex feature patterns. However, according to the research of David Duvenaud(9) and Kaiming He(3), there is a Degradation problem in the deep network: when the network depth increases, the network accuracy becomes saturated or even decreases due to overfitting.

Therefore, Kaiming he proposes to solve this problem by residual learning. Therefore, ResNet uses short circuit connections for learning. ResNet network is a reference to the VGG19 network, residual cells are added to network through the short-circuit connection on the basis of modifying VGG19. The change is mainly reflected in the fact that ResNet directly uses the convolution of stride=2 to sample, and replaces the dense layer with the global average pool layer.

2.3 Solved and Improvements

The structure of VGGNet is very simple. The entire network uses the same convolution kernel size (3x3) and the same maximum pooling size (2x2). Due to deeper structure using a combination of several small filters (3x3) convolution layers improves the performance compared with a large filter (5x5 or 7x7) convolution layer.

ResNet solves the problem of degradation in deep network learning. In this model, the chain derivative is propagated forward and the back propagation can be carried out smoothly, which improve the classification accuracy. ResNet proves that the performance can be improved by deepening the network structure.

2.4 Problems

VGG also has its own shortcoming, that is, the number of parameter is very large. Therefore, it takes more computing resources. This is because VGG has three fully connected layers, and most of the parameters are from the first one. If the training data set is limited, it is easy to cause overfitting.

Although ResNet has a higher accuracy rate, it takes long time for training because of the large number of layers. For some models which require quick training, ResNet is hard to meet the requirements. Moreover, it needs a large number of image features, so if the image size is too small, overfitting is prone to occur.

2.5 Unsolved Problem

The problem I am most interested in is that so far, almost any deep network model requires enough training sets to ensure training the parameters successfully. For example, in image classification, one class may need thousands of images to train in order to achieve a relatively high accuracy.

Deep learning can't do the same as human yet, which can learn an object through a small training set. At present, the main direction of deep learning is still focus on improving speed and accuracy, rather than to achieve the same method with smaller training sets. It will be a significant challenge to study how to reduce the data needed for training under the premise of ensuring performance.

3 Method / Model Description

In this paper, I use various deeper networks for evaluating the effectiveness of deeper CNN models for image classification on MNIST and CIFAR-10.

3.1 Model Architecture

- VGG (For VGG16, input (224x224 RGB image))

  $Input \longrightarrow conv3-64 \longrightarrow conv3-64 \longrightarrow maxpool \longrightarrow conv3-128 \longrightarrow conv3-128 \longrightarrow maxpool \longrightarrow conv3-256 \longrightarrow conv3-256 \longrightarrow conv3-256 \longrightarrow maxpool \longrightarrow conv3-512 \longrightarrow conv3-512 \longrightarrow conv3-512 \longrightarrow maxpool \longrightarrow conv3-512 \longrightarrow conv3-512 \longrightarrow conv3-512 \longrightarrow maxpool \longrightarrow FC4096 \longrightarrow FC4096 \longrightarrow FC4096 \longrightarrow softmax$

- ResNet (For ResNet34, input (224x224 RGB image))

  $Input \longrightarrow conv7-64 \longrightarrow maxpool \longrightarrow (RL) \longrightarrow conv3-64 \longrightarrow conv3-64 \longrightarrow (RL) \longrightarrow conv3-64 \longrightarrow conv3-64 \longrightarrow (RL) \longrightarrow conv3-64 \longrightarrow conv3-64$

  Here RL is Residual learning Fig.1 (4) shows it.

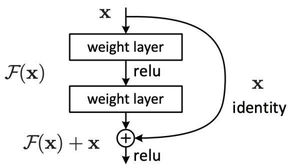

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1: Residual learning

4 Experiments

4.1 MNIST Dataset

The MNIST database(5) of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. Every image is 28x28 pixel. The MNIST examples show by Fig.2.

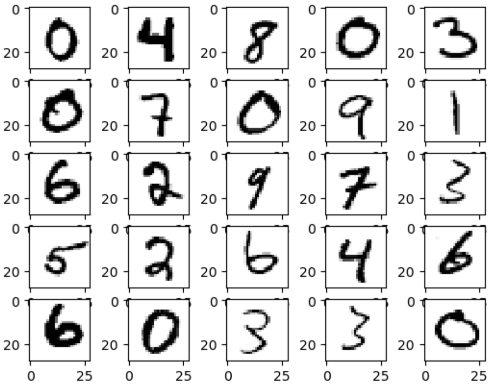


Figure 2: MNIST Examples

4.2 Environment/Data Preprocessing

We use the version 1.10.0 of tensorflow-gpu, and the Keras that with Tensorflow. The Keras version is 2.1.6. Using GPU can improve the computation speed.

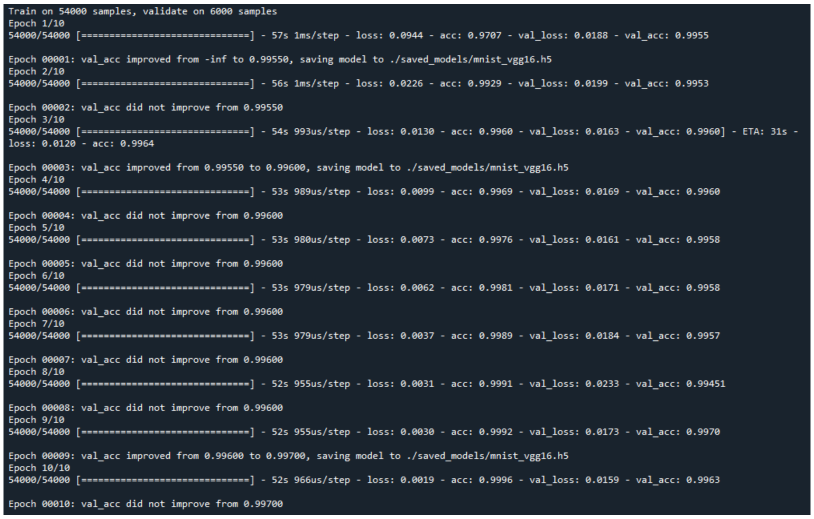
Keras Applications is a powerful CNN model library, and we found errors by importing the VGG16 model. Images need to be at least 48 by 48 to import. Because VGG and ResNet were designed for RGB diagrams, the input parameter is usually 224 * 224 * 3. The MNIST data set clearly does not fit this characteristic. The image of MNIST is a grayscale image of 28*28*1, so we changed the data set into a 48*48*3 data set through the tool in keras, so that the import was successful.

We took 90% of the data (54000) as the training set and 10% as the validation set (6000). For testing, 10,000 images are used.

4.3 Train/Test

We have done 10 epoch of training on VGG16 and ResNet20 respectively. The training process is shown in Figure 3, and the training results are shown in Figure 4 and Figure 5.

We are surprised to observe that the accuracy of VGG16 model training set reaches more than 90% after only one epoch of training. This sufficiently indicates that deeper neural networks have better prediction results.



```
Train on 54000 samples, validate on 6000 samples
Epoch 1/10
54000/54000 [==============================] - 57s 1ms/step - loss: 0.0944 - acc: 0.9707 - val_loss: 0.0188 - val_acc: 0.9955

Epoch 00001: val_acc improved from -inf to 0.99550, saving model to ./saved_models/mnist_vgg16.h5
Epoch 2/10
54000/54000 [==============================] - 56s 1ms/step - loss: 0.0226 - acc: 0.9929 - val_loss: 0.0199 - val_acc: 0.9953

Epoch 00002: val_acc did not improve from 0.99550
Epoch 3/10
54000/54000 [==============================] - 54s 993us/step - loss: 0.0130 - acc: 0.9960 - val_loss: 0.0163 - val_acc: 0.9960] - ETA: 31s -
loss: 0.0120 - acc: 0.9964

Epoch 00003: val_acc improved from 0.99550 to 0.99600, saving model to ./saved_models/mnist_vgg16.h5
Epoch 4/10
54000/54000 [==============================] - 53s 989us/step - loss: 0.0099 - acc: 0.9969 - val_loss: 0.0169 - val_acc: 0.9960

Epoch 00004: val_acc did not improve from 0.99600
Epoch 5/10
54000/54000 [==============================] - 53s 980us/step - loss: 0.0073 - acc: 0.9976 - val_loss: 0.0161 - val_acc: 0.9958

Epoch 00005: val_acc did not improve from 0.99600
Epoch 6/10
54000/54000 [==============================] - 53s 979us/step - loss: 0.0062 - acc: 0.9981 - val_loss: 0.0171 - val_acc: 0.9958

Epoch 00006: val_acc did not improve from 0.99600
Epoch 7/10
54000/54000 [==============================] - 53s 979us/step - loss: 0.0037 - acc: 0.9989 - val_loss: 0.0184 - val_acc: 0.9957

Epoch 00007: val_acc did not improve from 0.99600
Epoch 8/10
54000/54000 [==============================] - 52s 955us/step - loss: 0.0031 - acc: 0.9991 - val_loss: 0.0233 - val_acc: 0.99451

Epoch 00008: val_acc did not improve from 0.99600
Epoch 9/10
54000/54000 [==============================] - 52s 955us/step - loss: 0.0030 - acc: 0.9992 - val_loss: 0.0173 - val_acc: 0.9970

Epoch 00009: val_acc improved from 0.99600 to 0.99700, saving model to ./saved_models/mnist_vgg16.h5
Epoch 10/10
54000/54000 [==============================] - 52s 966us/step - loss: 0.0019 - acc: 0.9996 - val_loss: 0.0159 - val_acc: 0.9963

Epoch 00010: val_acc did not improve from 0.99700
```

Figure 3: VGG Train Process

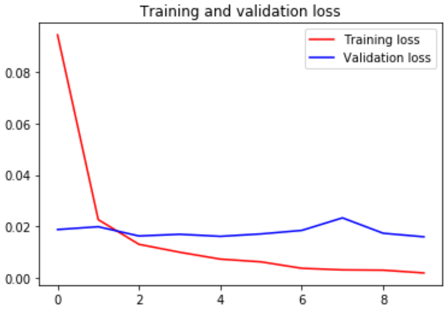


Figure 4: VGG Train Loss

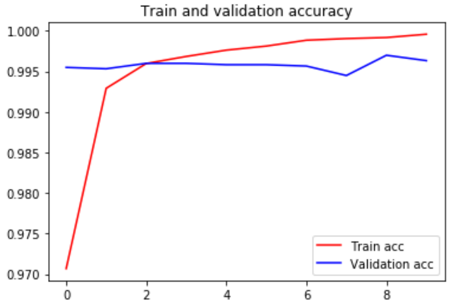


Figure 5: VGG Train Acc

Next, we have tried to train using ResNet. The source codes of building ResNet was from the keras official documentation(6), and modifications have done based on these codes.

4.4 Testing Results

After 10 epochs of training, we have selected the best model for prediction through change the settings and the results are shown in Table 1 and Figure 6.

Table 1: Testing Results

Model	Testing Results
VGG16	99.64%
ResNet	98.19%

It can be seen that both VGG16 and ResNet have achieved really good results with high accuracy. VGG16's results were even better than ResNet's. The predicted results are enough to show these classifiers are designed to be successful.

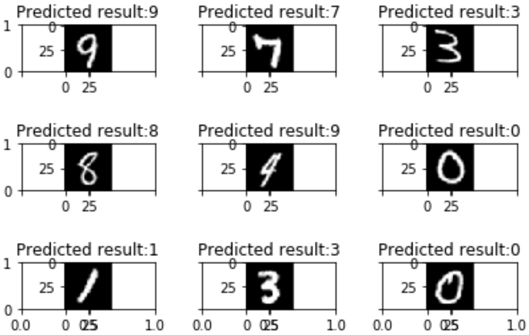


Figure 6: Prediction Visualization

4.5 The CIFAR-10 dataset

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(8).

We have also applied the same model to the CIFAR-10 dataset. After 10 epochs of training CIFAR-10 with the same parameter setting as MNIST, we have found that VGG16 has already converged and it is enough to finish training. But for deeper structure in ResNet, 10 epochs is not enough for sufficient training, thus 20 epochs of training were performed. The training results are shown in figure 7 and 8. The test results are shown in table 2.

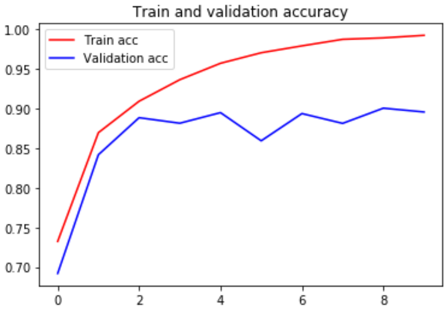


Figure 7: CIFAR-10 VGG Train Acc

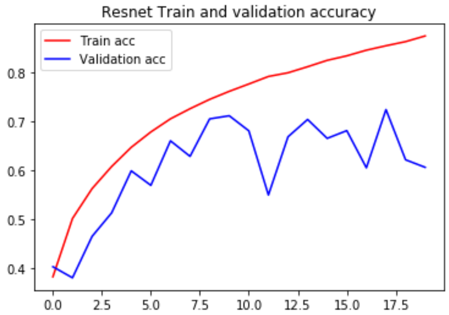


Figure 8: CIFAR-10 ResNet Train Acc

Table 2: CIFAR-10 Testing Results

Model	Testing Results
VGG16	88.7%
ResNet	38.47%

5 Conclusion

Experiments can demonstrate that the deeper neural network has better accuracy in image classification. However, we have also found that increasing the number of neural network layers continuously does not always lead to better results. In the experiments, the accuracy of VGG was better than that of ResNet on both MNIST and CIFAR-10 datasets. The low accuracy of CIFAR-10 can be attributed to the following reasons:

- The image size is too small (i.e. only 48*48*3) thus has insufficient features, which is fatal for performance of deep ResNet due to the risk of overfitting.
- Too few layers of ResNet (i.e. only 20 layers) are designed. Accuracy can be improved by using ResNet50.
- The training epochs are not enough. The accuracy can be improved if more epochs are used, but the training speed will become lower.

It can be concluded from the experiment that the appropriate model should be selected according to different kind of the training set. For training sets with small image size, VGG may be more appropriate to get fast training and can avoid overfitting effectively. For a training set with large image size which has enough features, ResNet is better for sufficient training than VGG because of its deeper and more complex structure.

References

[1] Guido Montufar; Razvan Pascanu; Kyunghyun Cho; Yoshua Bengio; *On the Number of Linear Regions of Deep Neural Networks,arXiv:1402.1869v2,2014.*

[2] Karen Simonyan ; Andrew Zisserman; *VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION, arXiv:1409.1556,2015.*

[3] Kaiming He; Xiangyu Zhang; Shaoqing Ren; Jian Sun; *Figure 1, Deep Residual Learning for Image Recognition,arXiv:1512.03385v1,2015.*

[4] Kaiming He; Xiangyu Zhang; Shaoqing Ren; Jian Sun; *Figure 2, Deep Residual Learning for Image Recognition,arXiv:1512.03385v1,2015.*

[5] Yann LeCun; Corinna Cortes; Christopher J.C. Burges; *THE MNIST DATABASE, http://yann.lecun.com/exdb/mnist/.*

[6] François Chollet; *https://keras.io/.*

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

[8] Alex Krizhevsky; *Learning Multiple Layers of Features from Tiny Images,2009.*

[9] David Duvenaud, Oren Rippel, Ryan P. Adams, Zoubin Ghahramani *Avoiding pathologies in very deep networks,2016.*

[10] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, Kaiming He *Aggregated Residual Transformations for Deep Neural Networks ,2016.*

[11] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R. Manmatha, Mu Li, and Alexander Smola *ResNeSt: Split-Attention Networks,2020.*