# Image Classification Based on Transfer Learning

Siping Hu
sh542@duke.edu

## Abstract

Although convolutional neural network has a satisfying performance in image classification, the tendency in recent years is the network becoming deeper and deeper, which lead to a heavier and heavier computational workload. Transfer learning uses the weights and parameters pre-trained in huge image set to initialize the local network, which can bring convenience to individual classification tasks since it helps to save local training time. This project aims at doing research on the performance of transfer learning, including the loss, accuracy and time in training process. After applying transfer learning to ResNet50, the 10-class animal classifier in my experiment can achieve the highest validation accuracy 97.0% and the highest test accuracy 92.3%. The average time to train one epoch is 8.3541s.

## 1 Introduction

Because of the rapid development of convolutional neural networks (CNN) in the past decade, it has occupied an unshakable position in the field of image classification. However, with the requirements for the depth of neural networks become deeper and deeper, the huge number of parameters makes the training process of CNN takes a long time to complete the corresponding calculation.

The emergence of transfer learning, by directly using the pre-trained weights to initialize the network parameters, greatly reduces the local training time for networks. The main purpose of this project is to explore the performance and efficiency of transfer learning. The project selects ResNet as the research object and carries out transfer learning on it, to built an animal classification algorithm to verify the performance of transfer learning.

## 2 Convolutional Neural Network Overview

### 2.1 Concept of CNN

CNN is developed on the basis of biological visual cortex research results. It combines deep neural network algorithm with traditional image processing convolution operation, and is a feedforward neural network[1].

The basic structure of CNN is mainly composed of input layer, convolution layer, activation function, pooling layer, fully connected layer and output layer. The convolutional layer and the pooling layer are usually alternately arranged in the convolutional network, that is, a pooling layer is arranged behind a convolutional layer, and then a convolutional layer is arranged after the pooling layer[2]. An important feature of CNN is that each neuron in the convolutional layer is only connected to a part of the input, that is, the local connection. The output value of the neuron is a weighted sum of the connection weight and the local input plus an offset value. This operation is similar to the process of convolution, so this network structure is called a convolutional neural network, and its typical structure is shown in Figure 2-1.

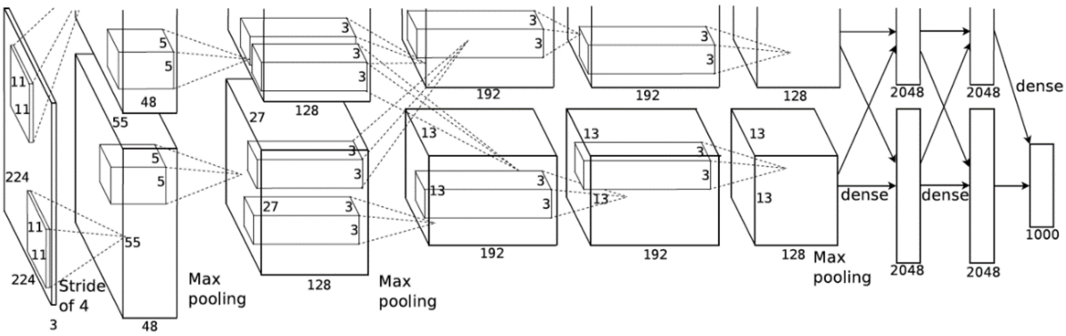


Fig 2-1. Typical structure of convolutional neural network

## 2.2 Residual Neural Network

In 2015, a 152-layer deep neural network---Residual Neural Network (ResNet), proposed by He Mingkai and four other Chinese people at Microsoft Research, won the ILSVRC2015 championship with a top5 3.57% error rate. At the same time, compared with VGGNet, the network has a lower number of parameters and excellent performance.

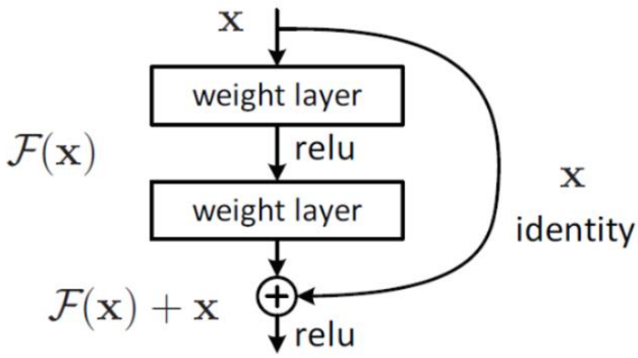


Fig 2-2. Residual unit

The innovation of ResNet lies in the proposed residual unit, as shown in Figure 2-2. Previous convolutional neural networks used a non-linear transformation of the input through the activation function, and ResNet added a direct connection channel to the residual unit, so that the network can retain a certain proportion of the output of the previous network layer. In other words, this layer of neural network does not need to learn the entire output, only the residual output from the previous network needs to be learned, so ResNet is also called a residual network[3].

The convolutional network or fully connected network before ResNet will lose some of the information when it transmits information, and at the same time, it will cause the gradient to disappear or the gradient to explode, resulting in a network that cannot be trained too deep. ResNet solves this problem to a certain extent by directly transferring input information to output to avoid loss of information. At the same time, because the entire network only needs to learn the difference between input and output, the difficulty of learning goals is reduced[4]. The structure of ResNet has the characteristics of rapidly improving the training speed of neural network training, and at the same time does not reduce the accuracy of the model.

# 3 Transfer Learning

Transfer learning, as the name implies, is to apply the knowledge that has been learned from other occasions. In the field of deep learning, transfer learning is to apply the knowledge learned from a task to another independent task[5].

Convolutional neural networks actually obtain image features through convolutional layers. These feature vectors will gradually become more complex as the depth of the convolutional network increases, which means that the features learned in the first few layers of the convolutional layer are relatively simple. , Basic, such as edges, contours, etc., these basic features are the characteristics that most pictures have, and are common. Therefore, when the sample is small, the accuracy of the model can be improved through transfer learning, and at the same time, the problem of overfitting can be effectively alleviated. There are usually two ways to transfer learning[6]:

1. Fine-tune

To fine-tune the network, we must first use the already trained network parameters to initialize the parameters of the new network as a pre-trained model. Then you can choose which layers of weights to fix and which layers of weights to train according to the size of the actual data set. Generally, when the data set is small, only the weights of the last two or three fully connected layers can be trained; when the data set is large enough, the weights of the entire network can be updated.

2. Feature extraction

Feature extraction requires removing the last soft-max layer of the convolutional neural network, and then treating the rest of the network as a fixed feature extractor, and then using a standard machine learning classification model (such as Logistic regression) Feature training, and finally get a classifier.

# 4 Experiment and Result

In the experiment, I select Resnet50 to build the model and load the pretrained parameters to initialize this network.

The dataset used in the project is Caltech256. I choose a subset of it which has 10 classes of animals: bear, chimp, giraffe, gorilla, llama, ostrich, porcupine, skunk, triceratops and zebra. The training set consists of 600 images (each class has 60 images); the validation set consists of 100 images (each class has 10 images), and the test set consists of 409 images.

To build the animal classifier, I replace the last linear layer of ResNet50 which has

256 outputs by a linear layer which has 10 outputs. Also, I fixed all the weights of layers but only train the last layer.

Fig 4-1 and fig 4-2 show the loss and accuracy of train set, validation set and test set separately. Fig 4-3 shows examples of the classifier's prediction to test images after the model is trained by 15 epochs.

I also measure the time to complete the training process. The average time to finish one epoch (including training, validating and testing) is 8.3541s.

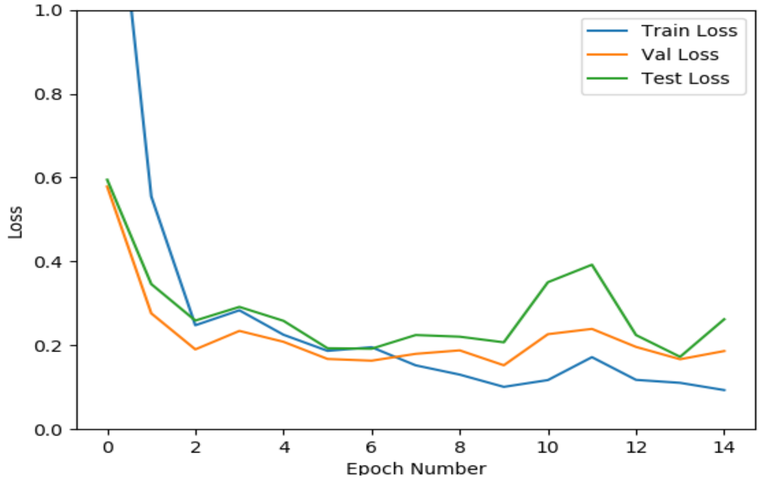


Fig 4-1. Loss of the animal classifier in 15 epochs

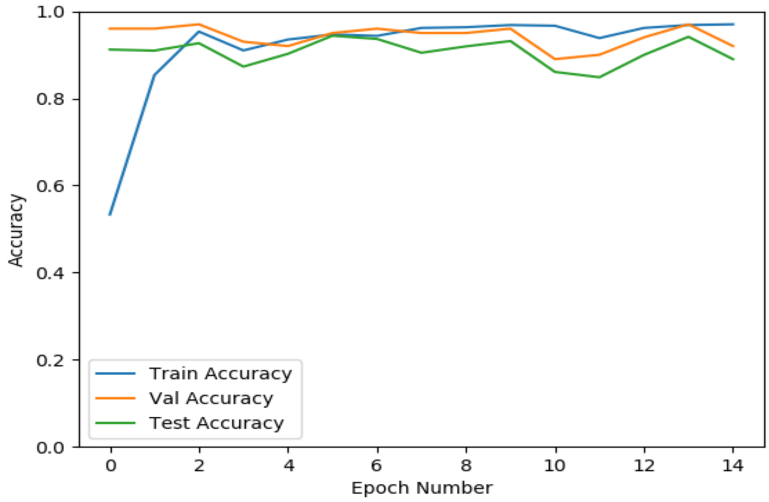


Fig 4-2. Accuracy of the animal classifier in 15 epochs

# 5 Conclusion

The results provided above obviously show the performance of the animal classifier based on transfer learning:

- From fig 4-1 and fig 4-2, we can find that the classifier can have an ideal accuracy only after 3 epochs, which means transfer learning is an useful and convenient way to complete the classification task.


Fig 4-3. Examples of the prediction result of the classifier after 15 epochs' training

- When training the model, I only trained the last linear layer and fixed all other layers. This method reduces the computation for undating weights in the whole training process. As the report mentioned above, the average time to train one epoch only costs around 8 seconds, which is really efficient.
- Fig 4-1 presents the trend of loss in train, val and test sets. We can see that the loss of test set can not be lower than 0.2. This maybe caused by the fixed weights of former layers. For further stpes, I'll try to train more layers (e.g. the last 3 layers), and figure out whether this emplement can decrease the loss and imporve the accuracy of the layer.

To conclude, in this project, I study the performance of transfer learning and find this method really has a satisfying performance. Applying transfer learning to CNN can not only save a bunch of time of training, but also avoid the tremendous computational work which always need to be completed by equipment with high quality. Last but not least, transfer learning can guarantee high accuracies in test images, which is an ideal method for classification tasks.

## References

[1] T. N. Sainath, A. Mohamed, B. Kingsbury and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, 2013, pp. 8614-8618.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks[C], " *Conference and Workshop on Neural Information Processing Systems, Lake Tahoe*, 2012:1097-1105.

[3] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778.

[4] Simonyan K , Zisserman A . "Very Deep Convolutional Networks for Large-Scale Image

Recognition[J], " *Computer Science*, 2014.

[5] Niculescu-Mizil, Alexandru; Caruana, Rich (March 21–24, 2007), "Inductive Transfer for Bayesian Network Structure Learning" , *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, retrieved 2007-08-05.

[6] L. Yang, L. Jing, J. Yu and M. K. Ng, "Learning Transferred Weights From Co-Occurrence Data for Heterogeneous Transfer Learning," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 11, pp. 2187-2200, Nov. 2016.