APPLICATION OF RESNET50 ON ANIMAL CLASSFICATION

NAME: Yucong Feng NUID: 001586064

ABSTRACT

The paper aims to use Residual Network 50 (ResNet50) *[1]* model to classify 12 kinds of animals. The dataset is consisted of different images which are collected from internet. The results show that we can successfully classify these animals based on the animal dataset, which means this can bea useful tool for image recognition in many fields. The image dataset of animals that I used can be found on AIStudio: https://aistudio.baidu.com/aistudio/datasetdetail/68755.

1 INTRODUCTION

With the decline of traditional taxonomy research, there are fewer and fewer taxonomy experts, but there are still a large number of animal and plant specimens and photos that need to be quickly identified and identified; at the same time, field natural education is gradually rising, which needs to be able to quickly identify species and provide relevant tools. Some popular image recognition networks, such as AlexNet, VGGNet, ResNet, and InceptionV1-V4 network, are widely used in image recognition. Since I want our classified image to have higher precision, I chose the ResNet50 as the network I use.

Residual Network 50 *[2]* is a powerful tool in the state of image recognition. Many types of research have been conducted around this topic, and this mode advanced dramatically in recent years. This model was immensely successful, as can be ascertained from the fact that its ensemble won the top position at the ILSVRC 2015 classification competition with an error of only 3.57%. Additionally, it also came first in the ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation in the ILSVRC & COCO competitions of 2015. *[3]*

The dataset of animals is collected from the internet. Figure 1 shows a part of the dataset. This training dataset has a total of 12 different categories of animals, and each category has about 600 different images, a total of 7096 images.

Figure 1: *Sample from dataset*

After training, the model can predict some images with good accuracy. Figure 2 shows some predicted results.

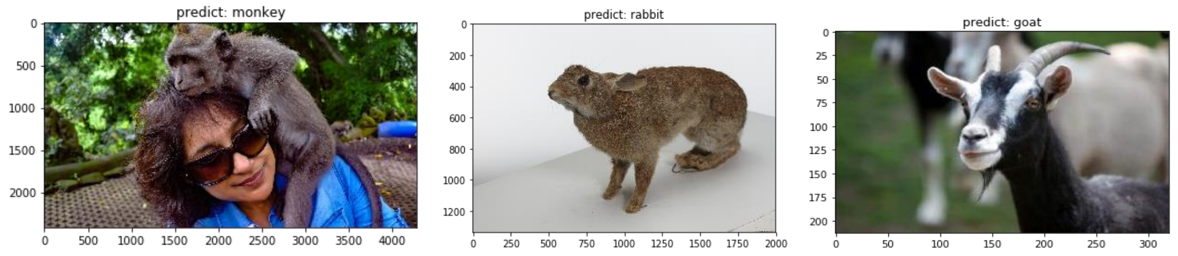


Figure 2: *Predicted results*

2 METHOD

2.1 RESNET50 ARICTECTURE

The original ResNet architecture was ResNet-34, which comprised 34 weighted layers. It provided a novel way to add more convolutional layers to a CNN, without running into the vanishing gradient problem, using the concept of shortcut connections *[4]*. A shortcut connection “skips over” some layers, converting a regular network to a residual network.

The regular network was based on the VGG neural networks (VGG-16 and VGG-19)—each convolutional network had a 3×3 filter. However, a ResNet has fewer filters and is less complex than a VGGNet. A 34-layer ResNet can achieve a performance of 3.6 billion FLOPs, and a smaller 18-layer ResNet can achieve 1.8 billion FLOPs, which is significantly faster than a VGG-19 Network with 19.6 billion FLOPs. Figure 3 shows the difference between these models.

The ResNet architecture follows two basic design rules: First, the number of filters in each layer is the same depending on the size of the output feature map. Second, if the feature map’s size is halved, it has double the number of filters to maintain the time complexity of each layer.

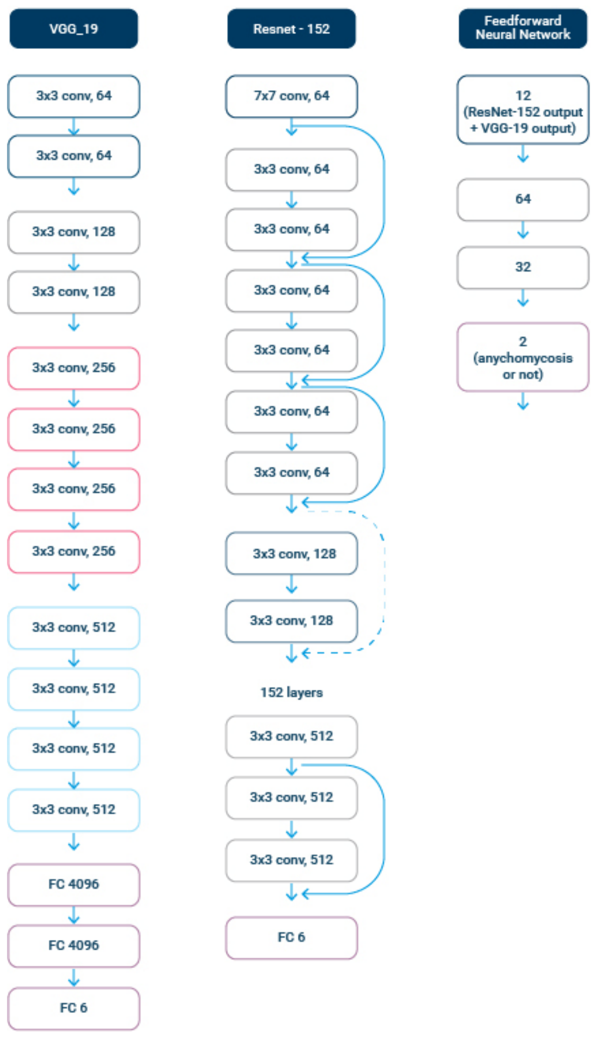


Figure 3: *Differences between architectures*

2.2 The Special Characteristics Of ResNet50

ResNet-50 is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer). Residual neural networks are a type of artificial neural network (ANN) that forms networks by stacking residual blocks. Figure 4 shows the architecture of ResNet50 model.

ResNet-50 has an architecture based on the model depicted above, but with one important difference. The 50-layer ResNet uses a bottleneck design for the building block. A bottleneck residual block uses 1×1 convolutions, known as a “bottleneck”, which reduces the number of parameters and matrix multiplications. This enables much faster training of each layer. It uses a stack of three layers rather than two layers.

The 50-layer ResNet architecture includes the following elements, as shown in the figure 5 below: A 7×7 kernel convolution alongside 64 other kernels with a 2-sized stride; A max pooling layer with a 2-sized stride; 9 more layers—3×3,64 kernel convolution, another with 1×1,64 kernels, and a third with 1×1,256 kernels. These 3 layers are repeated 3 times; 12 more layers with 1×1,128 kernels, 3×3,128 kernels, and 1×1,512 kernels, iterated 4 times; 18 more layers with 1×1,256 cores, and 2 cores 3×3,256 and 1×1,1024, iterated 6 times; 9 more layers with 1×1,512 cores, 3×3,512 cores, and 1×1,2048 cores iterated 3 times. *[5]*

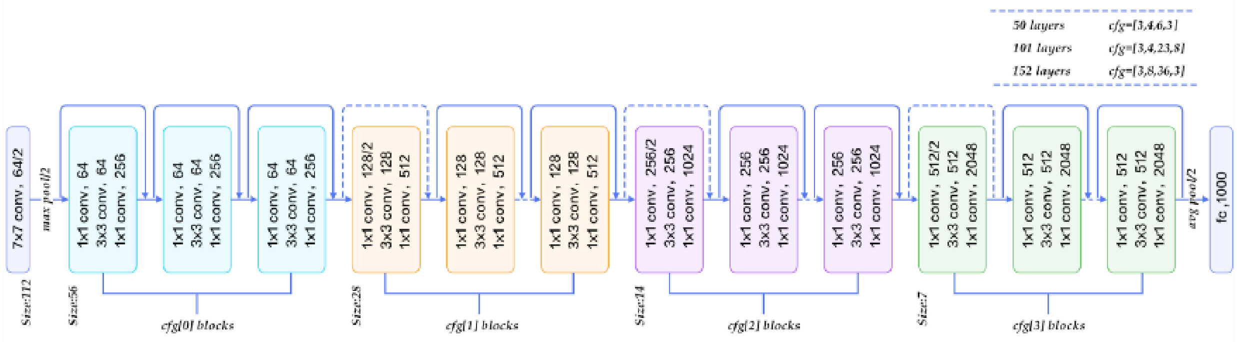

50 layers cfg=[3,4,6,3]
101 layers cfg=[3,4,23,8]
152 layers cfg=[3,8,36,3]
7x7 conv, 64/2
max pool/2
1x1 conv, 64 3x3 conv, 64 1x1 conv, 256
1x1 conv, 64 3x3 conv, 64 1x1 conv, 256
1x1 conv, 64 3x3 conv, 64 1x1 conv, 256
1x1 conv, 128/2 3x3 conv, 128 1x1 conv, 512
1x1 conv, 128 3x3 conv, 128 1x1 conv, 512
1x1 conv, 128 3x3 conv, 128 1x1 conv, 512
1x1 conv, 256/2 3x3 conv, 256 1x1 conv, 1024
1x1 conv, 256 3x3 conv, 256 1x1 conv, 1024
1x1 conv, 256 3x3 conv, 256 1x1 conv, 1024
1x1 conv, 512/2 3x3 conv, 512 1x1 conv, 2048
1x1 conv, 512 3x3 conv, 512 1x1 conv, 2048
1x1 conv, 512 3x3 conv, 512 1x1 conv, 2048
avg pool/2
fc ,1000
Size:112
Size:56
cfg[0] blocks
Size:28
cfg[1] blocks
Size:14
cfg[2] blocks
Size:7
cfg[3] blocks

Figure 4: *ResNet50 structure*

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

Figure 5: *Special characteristics of ResNet-50*

2.3 Skip Connections

In this part, we explore the skip connections in ResNet50. The ResNet paper popularized the approach of using Skip Connections. Residual blocks are the essential building blocks of ResNet networks. To make very deep convolutional structures possible, ResNet adds intermediate inputs to the output of a group of convolution blocks. This is also called skip connections, identity mapping, and "residual connections.

The objective of skip connections is to allow smoother gradient flow, and ensure important features carry until the final layers. They do not add computational load to the network.

The following diagram illustrates a residual block, where x is the input to the ResNet block—output from previous layers, and F(x) is a small neural network with several convolution blocks

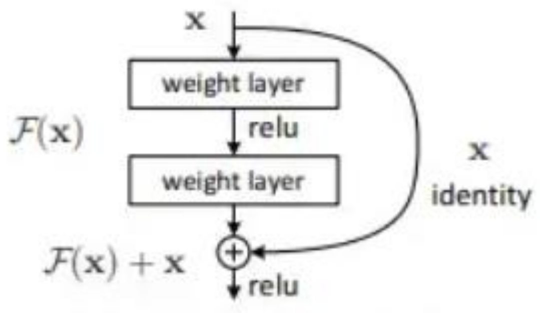

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 6: *GAN with convolution feature*

Figure 6 illustrates a residual block, where:

1. x is the input to the ResNet block—output from previous layers
2. F(x) is a small neural network with several convolution blocks

In mathematical terms, it would mean y= x + F(x) where y is the final output of the layer.

2.4 Model Optimization

I use cosine annealing as the method of model optimization. Cosine annealing is a type of learning rate schedule that has the effect of starting with a large learning rate that is relatively rapidly decreased to a minimum value before being increased rapidly again. The resetting of the learning rate acts like a simulated restart of the learning process and the re-use of good weights as the starting point of the restart is referred to as a "warm restart" in contrast to a "cold restart" where a new set of small random numbers may be used as a starting point. *[6]*

Set the learning rate of each parameter group using a cosine annealing schedule, where η_{max} is set to the initial lr and T_{cur} is the number of epochs since the last restart in SGDR:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + \cos\left(\frac{T_{cur}}{T_{max}}\pi\right)\right), \quad T_{cur} \neq (2k+1)T_{max};$$
$$\eta_{t+1} = \eta_t + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 - \cos\left(\frac{1}{T_{max}}\pi\right)\right), \quad T_{cur} = (2k+1)T_{max}.$$

When last_epoch=-1, sets initial lr as lr. Notice that because the schedule is defined recursively, the learning rate can be simultaneously modified outside this scheduler by other operators. If the learning rate is set solely by this scheduler, the learning rate at each step becomes:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + \cos\left(\frac{T_{cur}}{T_{max}}\pi\right)\right)$$

3 EXPERIMENTAL RESULTS

3.1 EVALUATION METHOD

To evaluate the result, we use an evaluation method: Random Sampling, to see the performance of ResNet50.

Random Sampling: In a simple random sample (SRS) of a given size, all subsets of a sampling frame have an equal probability of being selected. Each element of the frame thus has an equal probability of selection: the frame is not subdivided or partitioned. Furthermore, any given pair of elements has the same chance of selection as any other such pair. This minimizes bias and simplifies analysis of results. In particular, the variance between individual results within the sample is a good indicator of variance in the overall population, which makes it relatively easy to estimate the accuracy of results.

3.2 EVALUATION RESULT

The result is shown in the below table:

Table 1: *Evaluation Result*

ID: 2	**ID: 38**	**ID: 56**	**ID: 100**	**ID: 303**
Label: rat	**Label: rat**	**Label: pig**	**Label: pig**	**Label: tiger**

ID: 2	**ID: 38**	**ID: 56**	**ID: 100**	**ID: 303**
Predict: rat	**Predict: rat**	**Predict: pig**	**Predict: pig**	**Predict: tiger**

By reviewing the result, it can be believed that ResNet50 can make great classification.

4 CONCLUSION

In this experiment, it demonstrates that ResNet can make great classification. Fortunately, the result shows a good demonstration of accuracy. However, I believe there should be a more detailed way to quantify the result comprehensively, maybe next time I should try to find some other evaluation metrics to measure the performance of this model. For example, we use Manhattan distance to measure the accuracy of K-means or Fréchet inception distance (FID) in generative model.

To sum up, residual network or ResNet was a major innovation that has changed the training of deep convolutional neural networks for tasks related to computer vision. While the original Resnet had 34 layers and used 2-layer blocks, other advanced variants such as the Resnet50 made the use of 3-layer bottleneck blocks to ensure improved accuracy and lesser training time.

REFERENCES

[1] Li, H.-T., Yang, & Hochreiter, S. (2017). Residual Network training for image recognition. arXiv preprint arXiv:1809.11096.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2015). Deep Residual Learning for Image Recognition.

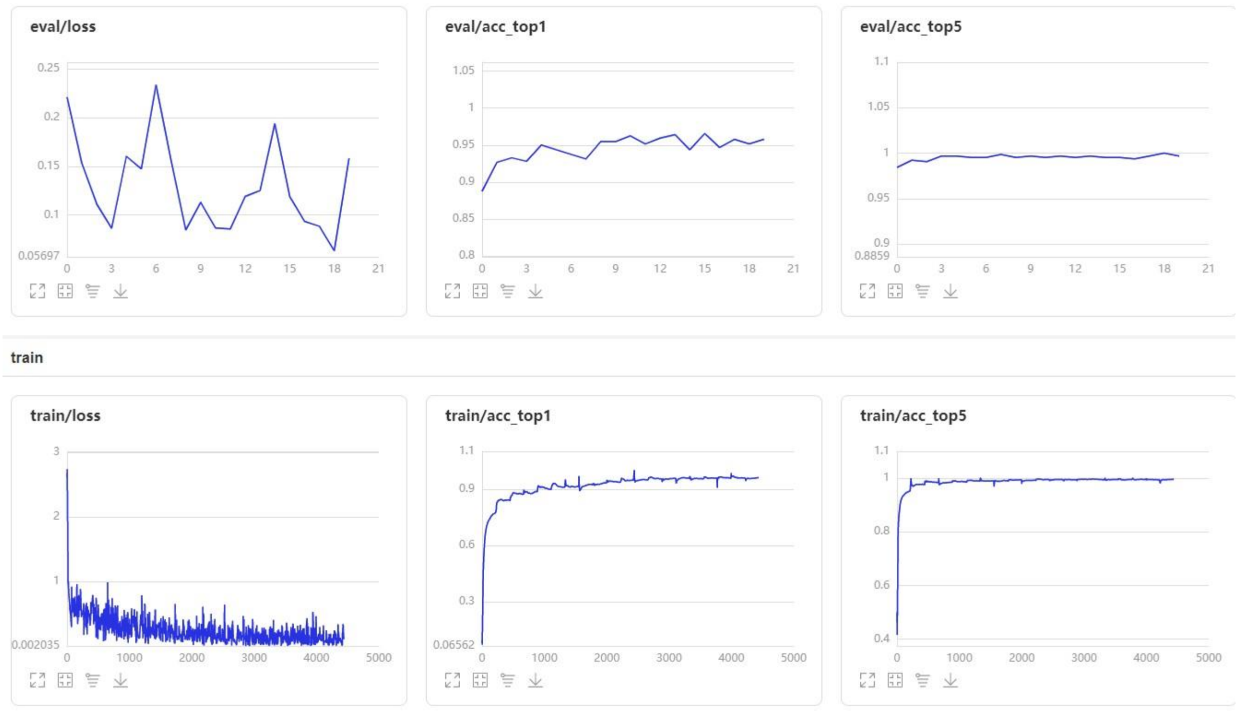
[3] Gaudenz, & Boesch. (2019). Deep residual network. Advances in neural information processing systems, 28.

[4] Miraclebiu (2016) TFFRCN_resnet50 - Faster rcnn based on tensorflow and resnet50 [Source code]. https://github.com/miraclebiu/TFFRCN_resnet50/tree/master/faster_rcnn

[5] Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., & Welling, M. (2016). Detailed techniques to implement ResNets. Advances in neural information processing systems, 26.

[6] Alekseys, V., Bilofur, A., & Zidane, Y. (2016). Cosine annealed warm restart learning schedulers

APPENDIX IMAGE CLASSIFICATION FITTING RESULT


eval/loss
eval/acc_top1
eval/acc_top5
train
train/loss
train/acc_top1
train/acc_top5