Resnet-50 image classification : changing the architecture and technique of the model

Lee Wonbyung
Applied Artificial Intelligence, Sungkunkwan University

March 2024

1 Introduction

ResNet[1], or Residual Network, is a structure developed to solve the Vanishing Gradient problem that occurs when training deep neural networks. It was introduced by Kaiming He of Microsoft Research and his colleagues in 2015.

The core idea of ResNet is "Residual Learning." In residual learning, layers learn "residual" (differences) from input to output, instead of learning the input directly. This is done by adding direct "skip connections" or "shortcut connections" to the network. These connections reduce training problems that can arise when the network goes deeper, by skipping one or more layers and adding the input directly to the output.

For example, there are different versions of ResNet such as ResNet-50[1], ResNet-101, and ResNet-152, which means the number of layers each model has. ResNet is particularly effective in image recognition and classification tasks, and is used in many computer vision systems and applications.

In this work, the ResNet-50 model is used, and the image classification performance of this model is compared and analyzed.[2]

2 Dataset

2.1 Caltech-101

Caltech-101 is one of the widely used image datasets for computer vision research. The dataset was created by the California Institute of Technology and contains approximately 9,000 images, roughly classified into 101 different categories. Each category has between 40 and 800 images, and most classes consist of about 50 images.

The purpose of Caltech-101 is to evaluate the performance of object recognition and image classification algorithms. The different categories of the dataset contain images of different everyday objects, such as human faces, different an-

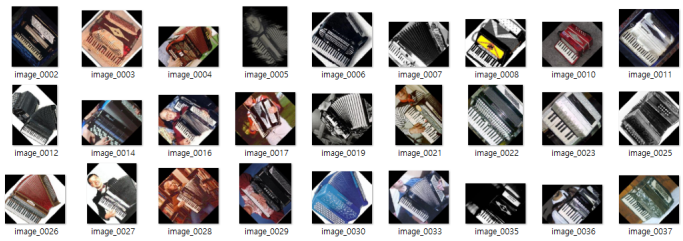


Figure 1: Caltech-101 dataset (accordian categories)

imals, plants, cars, and chairs. Each image mainly contains one target object at the center, making it suitable for testing object recognition techniques.

2.2 Data Setting

In this work, the Caltech-101 dataset is randomly divided into a training set and a validation set for model training. The ratio of training and validation is divided into 8:2, and images are randomly distributed by 101 categories in each dataset. Figure. 1 is an example of train set of Caltech-101 dataset.

3 Environment

The experimental environment is conducted in the anaconda virtual environment of the visual study code, and all experiments are conducted in the version of Python 3.11.5. The specifications of the computer are Intel Core i9-13900K. The batch size for each model and combination is unified to 64 and epochs to 10. If the batch size is set to 128 or more, it is set to 64 because data fit is not smoothly performed. It is expected to occur because the performance of the computer specification is poor or the amount of training data is relatively small.

4 Model Tweaks

4.1 Model Architecture

A model tweak is a minor adjustment to the network architecture, such as changing the stride of a particular convolution layer. Such a tweak often barely changes the computational complexity but might have a non-negligible effect on the model accuracy. In this section, we use ResNet as an example to investigate the effects of model tweaks. There are totally four models here. All of them require some modifications based on the ResNet-50 model. The model-A uses the original ResNet-50 model as it is. Figure. 2 shows the schematic of the ResNet-50 model.

Model-B: This tweak is proposed in Inception-v2[3] originally, and it can be found on the implementations of other models. This model switches the stride

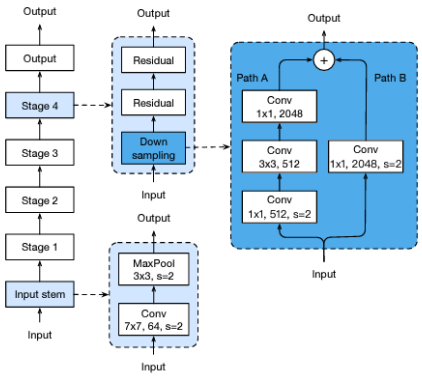


Figure 2: The architecture of ResNet-50.

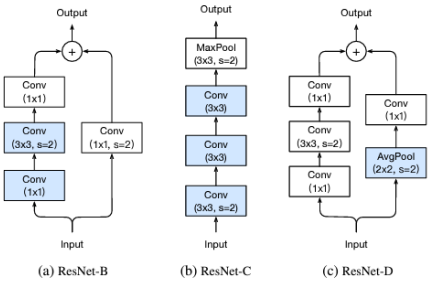


Figure 3: Three ResNet tweaks.

sizes of the first two convolutions in path A as shown in Figure. 3a, so no information is ignored. Because the second convolution has a kernel size of 3×3, the output shape of path A does not change.

Model-C: This modification replaces the 7×7 convolution of the input stem with the three conservative 3×3 convolutions shown in Figure. 3b, with the first and second convolutions having 32 output channels and the last one using 64 output channels.

Model-D: Before convolution, we find the addition of a 2×2 average pooling layer with a stride of 2, whose stride works well in practice and has little effect on the computational cost.[2]

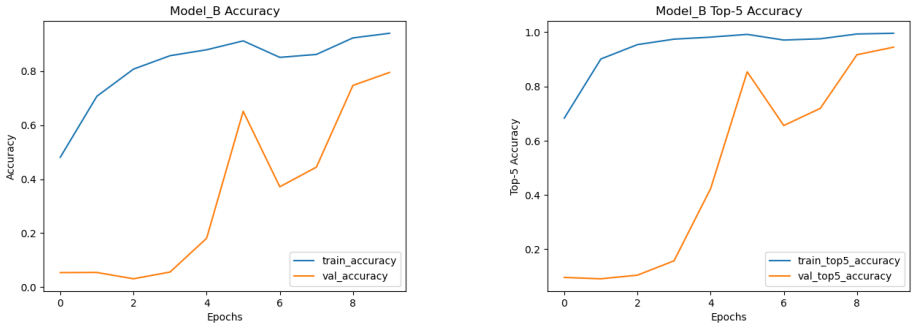
4.2 Experiment Results

We train and evaluate four ResNet models with Caltech-101 database. As an evaluation index, Top-1 accuracy and Top-5 accuracy are calculated and compared. Top-1 and top-5 accuracy are evaluated by the validation set previously separated from the Caltech-101 database. The result is in Table. 1.

The results show that the Top-1 accuracy and Top-5 accuracy of Model-

Model	Top-1 acc	Top-5 acc
Model-A	0.66	0.88
Model-B	**0.79**	**0.95**
Model-C	0.63	0.83
Model-D	0.63	0.84

Table 1: Compare tweaks models (BS=64, epochs=10)



(a) Model-B accuracy

(b) Model-B top-5 accuracy

Figure 4: Visualization Model-B's performance

B are the highest. In other words, in the ResNet-50 model, the best performance is shown by exchanging the strides of the first and second convolution of the downsampling block. Subsequently, the basic model, Model-A, appears to show the second highest accuracy. In Figure. 4, as epochs increase, the accuracy of validation increases. Therefore, we conduct some additional experiments on Model-A and Model-B, which shows good performance, and examine performance changes.

5 Training Refinements

In this section, we describe three training refinements that aim to further improve the model accuracy. In the existing paper, a knowledge distinction method is also conducted to learn by adding ResNet-152 as a teacher model. However, this method is excluded because the model itself is too large to be suitable for learning in the current environment.

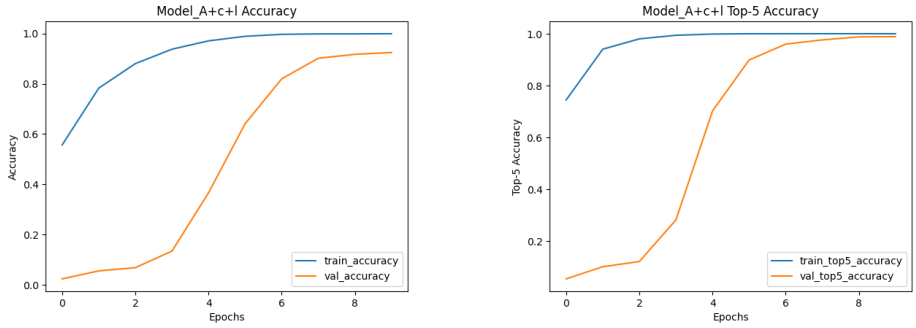
5.1 Cosine Learning Rate Decay

This refinement decreases the learning rate from the initial value to 0 by following the cosine function. This strategy is a cosine annealing strategy proposed by Loshchilov et al.[4]. The cosine decay starts to decay the learning since the beginning but remains large until step decay reduces the learning rate by 10x,

Model	Top-1 acc	Top-5 acc
A+c	0.9	0.98
A+l	0.82	0.95
A+m	0.83	0.95
A+c+l	**0.93**	**0.99**
A+c+m	0.92	0.98
A+l+m	0.81	0.94
A+c+l+m	0.91	0.98

Table 2: Training refinements with Model-A /c : cosine learning rate decay, l : label smoothing, m : mixup train

which potentially improves the training progress. In the experiment, we set the cosine decay learning rate to 0.001 and proceed by applying it to the optimizer.



(a) Model-A+c+l accuracy

(b) Model-A+c+l top-5 accuracy

Figure 5: Visualization Model-A+c+l's performance

5.2 Label Smoothing

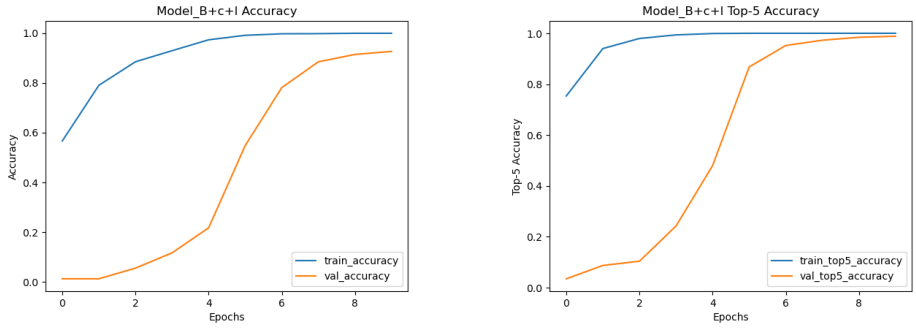
Label Smoothing is one of the data regularization techniques that has gained attention by increasing the generalization performance of the model while being a simple method. Label smoothing is a technique proposed by Szegedy et al.[3]. With label smoothing, the distribution centers at the theoretical value and has fewer extreme values. We set the label smoothing to 0.1 in model experiments.

5.3 Mixup Train

In mixup train[5], each time we randomly sample two examples. Then we form a new example by a weighted linear interpolation of these two examples. In mixed training, only new examples (ˆ x, ˆ y) are used.

Model	Top-1 acc	Top-5 acc
B+c	0.9	0.98
B+l	0.84	0.97
B+m	0.78	0.91
B+c+l	**0.93**	**0.99**
B+c+m	0.91	0.98
B+l+m	0.79	0.94
B+c+l+m	0.92	**0.99**

Table 3: Training refinements with Model-B /c : cosine learning rate decay, l : label smoothing, m : mixup train



(a) Model-B+c+l accuracy

(b) Model-B+c+l top-5 accuracy

Figure 6: Visualization Model-B+c+l's performance

5.4 Experiment Results

The results of this experiment look at the performance change when training refinements of the model are applied. In the previous model tweaks experiment, the experiment is conducted on model-B, which performed the best, and model-A, which was the next best. All possible combinations are tested using the three refinements, and the results of a total of seven types of models will be compared and analyzed.

First, the experimental results for model-A are shown in Table. 2. Looking at the results, the performance of model-A+c+l is the highest. That is, it can be seen that both Top-1 and Top-5 accuracy of the model applied by combining cosine decay and label smoothing with model-A are the highest. In Figure. 5, as the epoch increases, the difference between training accuracy and validation accuracy decreases significantly. When reaching epoch 10, there is little difference between the two accuracy. This means that the method of applying cosine decay and label smoothing together is very suitable for this model.

Next, the same refinements are applied to model-B. Model training & evaluation results are shown in Table. 3 and visualization is shown in Figure. 6. The results show that, like model-A, the accuracy of the model with cosine de-

cay and label smoothing applied was the highest at 0.93 and Top-5 accuracy at 0.99. Also, looking at the cases where refines were applied only one by one, the performance of the model with cosine decay applied seems to be good. The learning results of model-A show a similar trend. In addition, considering that the Top-5 accuracy of model-A was 0.88 at the time of the first experiment, the performance of model-A is significantly increased. Of course, the methods applied to model-B also increase in performance, but are less than the increase rate of model-A.

6 Conclusion

In these experiments, we try various changes based on the resnet-50 model. Layers are added in the original architecture or techniques are added in the training process. What is meaningful is that the performance has increased due to changes in the model configuration. In addition, only learning about image classification is conducted here. However, performance increases can be expected in other fields where deep learning models are used, such as object detection and segmentation.

I am interested in pose estimation, a part of vision. Among them, I am particularly considering the use of vision in combination with sports. Sports have traditionally prioritized the intuition and decision of coaching steps in the field. However, artificial intelligence technology has developed tremendously, and there is a good possibility that techniques such as vision will be used in the sports field. For example, vision models can be helpful in real-life sports such as fitness and yoga[6]. Vision technology can be applied as well to soccer[7] and baseball[8], which are major sports events. As the first goal of sports is victory, the artificial intelligence model can be the target if it helps the game win.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.

[5] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017.

[6] Debanjan Borthakur, Arindam Paul, Dev Kapil, and Manob Jyoti Saikia. Yoga pose estimation using angle-based feature extraction. *Healthcare*, 11(24), 2023.

[7] Yutao Cui, Chenkai Zeng, Xiaoyu Zhao, Yichun Yang, Gangshan Wu, and Limin Wang. Sportsmot: A large multi-object tracking dataset in multiple sports scenes. *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9921–9931, 2023.

[8] Yung-Che Li, Ching-Tang Chang, Chin-Chang Cheng, and Yu-Len Huang. Baseball swing pose estimation using openpose. *2021 IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI)*, pages 6–9, 2021.