NYCU Visual Recognitionusing Deep Learning Homework 1 Report

Tsung-Wei Tsai 313553002

March 21, 2025

Abstract

The report describes the method how I finish the homework, including model selection and data processing. Also the result and ablation study.

1 Introduction

The objective of the homework is Image Classification with 100 class, use ResNet as backbone. To finish the task, I choose ResNet50 as the backbone considering the hardware limit and model size requirement, with some modification to change the original 1000 class classification to 100 class classification.

In order to avoid overfitting, I also use some data augmentation technology, providing great improvement to the model performance.

With the above method, the model has an accuracy of about 92.6%, which is beyond a strong baseline in the private leaderboard. The work can find here https://github.com/wwiigh/cvhw1.

2 Method

In this section, I will introduce the method about my work, including data pre-processing, model architecture, hyperparameter settings, and loss function.

2.1 Model architecture

In the homework, I use ResNet50[Pyt17a] as backbone, and use the pre-trained weight, IMAGENET1K_V2, which has 80.858% accuracy on ImageNet-1K TOP-1 score.

Because original ResNet50 is train for classification with 1000 class, but in this work we only need 100 class, so I remove the last fc layer in pytorch ResNet50, and add a two layer MLP to fit the 100 class.

Before the last fc layer in ResNet50, the output dimension is 2048, to change to 100 dimension, first use a fc layer change dimension from 2048 to 512, following Batch Normalization and SiLU, then second fc layer change dimension from 512 to 100.

2.2 Data pre-processing

2.2.1 Data Augmentation

In order to avoid overfitting, I use some data augmentation to make data more diversity. Bellow is the augmentation I use. Follow the order of augmentation.

- Horizontal Flip with probability 0.5
- Vertical Flip with probability 0.2
- Rotation with degrees 15
- RandomAffine with translate in (0.1, 0.1), scale in (0.9, 1.1)

Method	Acc
Whole method	92%
No MLP	91.2%
No Mixup	91%

Table 1: Model results on public score board.

- ColorJitter with brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1
- RandAugment with number of augmentation 2, magnitude 10
- Resize image to (224, 224),
- RandomErasing with probability 0.3, scale in (0.02, 0.1), ratio in (0.3, 3.3), and erasing each pixel with random values
- Normalize image with mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]

All of the augmentation above are use pytorch function, which can be find here[Pyt17b]. Also resize and normalize image is to fit the input format of pytorch ResNet50.

2.2.2 MixUp

After above setting, I found the performance is not good enough. Then I find in [ZCDLP18], they use mixup to improve the model performance. As proposed in [ZCDLP18], their formulation is given below:

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j$$

$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j$$

x_i, x_j are our original data, and y_i, y_j are corresponding original labels. λ is obtain by Beta distribution control by α.

After using method proposed in [ZCDLP18], the model performance get better, and achieve strong baseline.

2.3 Loss and experiment setting

The whole architecture is implement with pytorch. For loss function, I choose CrossEntropyLoss with label smoothing 0.1, to avoid overfitting. I try to add focal loss to improve performance but don't see significant improvement. Ohter settings are batch size 64, 100 epochs, learning rate 5e-4. Optimizer is AdamW with weight decay 5e-4. Scheduler is ReduceLROnPlateau with factor 0.5 and patience 3. α in mixup is 0.2. Also before the second fc layer in MLP, I add Dropout layer with probability 0.5.

Total training time is about 10 hours and the model parameters are 24609444. GPU is NVIDIA GeForce RTX 2070.

3 Results

In this section, first I analysis the model performance, and some ablation study. The result can see in Table.1.

3.1 Model performance

With the setting above, the model get 92% accuracy in public leaderboard, and 92.6% in private leaderboard. The training loss and validation accuracy can see in Figure.2 and Figure.3. Although I set the epoch to 100, it can find that the training loss and validation accuracy is about shock after epoch 40. And the training loss is about 1.34 and validation accuracy is about 86%.

In confusion matrix heatmap(Figure.1), it is too big and sparse for directly analysis, so I chose Top 10 error class to check model limitation. After calculation, the top 10 class that have less accuracy are 50, 83, 87, 88, 57, 18, 62, 35, 56, 26, sorting in highest error rate. Most of them are plant, which

means that the model has some space for a better result in the plant area. I think using other data augmentation methods can solve the problem, but I don't find the right way to get a better result.

3.2 Ablation study

In this section, I modify some component in my model to analysis the model design. Including MLP layer and mixup.

3.2.1 MLP layer

In my design, I use two MLP layer to replace origin 2048 to 1000 fc layer in ResNet50. Because ResNet50 is for 1000 class classification, I think if direct change 1000 class to 100 class, the feature dimension is drop from 2048 to 100, which may cause model difficult to learn the relationship between feature and class. So to make the dimension decrease more reasonably and learn the relationship, I design the two layer MLP as describe in section2.1.

In Table.1, we can see if we don't use MLP and simply change to 100 class, the performance is about 0.08 % worsen then use MLP, which imply that model can learn better relationship between feature and class with more layer if we set dimension decrease reasonably.

3.2.2 Without Mixup

In [ZCDLP18], they find use mixup can improve the model performance, avoid overfitting. To verify their finding and improve accuracy, I also use mixup in this work and compare with no mixup version to see the difference. In Table.1 we can see if use mixup, the model can have better performance and improve the accuracy about 1%. This imply using mixup does help the model have better generalization, and get the better performance.

3.3 Discussion

Although the above three method can get over 90% in testing pictures, but if we check the Figure.3, we can find none of these method can exceed 90%. The reason I think is because we only have 300 pictures for validation, which is much smaller than 2344 testing pictures. So the accuracy here will have some difference when testing.

In Figure.2, we can find if we don't add mixup, the loss is lowest and smooth then the other two. I think this is because mixup will make dataset more diversity. The best direction to optimize parameter in this epoch may be wrong direction in next epoch. So the loss is hard to decrease and look unstable. If we don't use mixup, the data we see in this epoch is about the same in next epoch, make the update more stable and easy to decrease.

4 Conclusion

In this work, I use ResNet50 as backbone, with two layer small MLP for classification. After using the data augmentation, mixup, the model performance can get 92% in public leaderboard and 92.6% in private leaderboard, which is over strong baseline. Also do some ablation study to verify my model design, with discussion on loss curve and validation accuracy.

References

[Pyt17a] Pytorch. resnet50, 2017. [Online; accessed 18-March-2025].

[Pyt17b] Pytorch. Transforming and augmenting images, 2017. [Online; accessed 18-March-2025].

[ZCDLP18] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018.

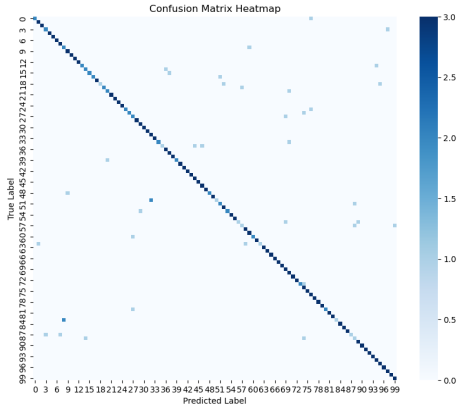

Figure 1: Confusion Matrix Heatmap. It can find that most of the value are on the diagonal, which means model have great accuracy on most case, except some sparse points indicate failure case.

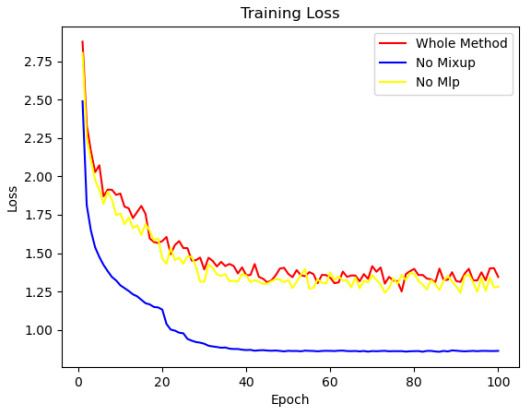

Figure 2: Training Loss. It can see that if we don't have mixup, the curve is smooth and lower than other two methods. The behavior of whole method and no mlp are about the same.

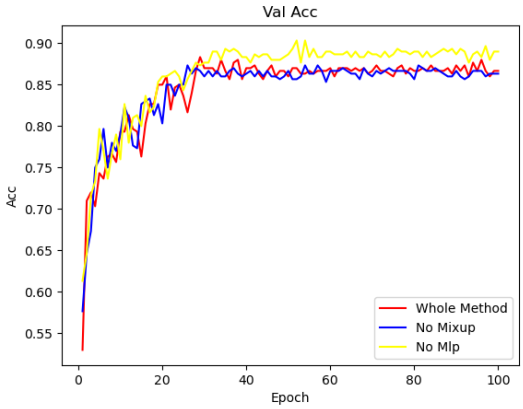

Figure 3: Val Acc. All of the method have about the same performance on validation set.