CS087: Computer Science: Advanced Topics Course Project Report

Yuxuan Duan* 516030910573

Abstract

This report is based on the project of Team 5[1]: *How Does Regularization Influences the Performance of Networks with Projection Shortcuts*. In this project, we focused on various kinds of regularization terms and used them on the networks with projection shortcuts. Furthermore, we designed *mutual regularization*, which considers more than one weight matrix, as the feature of this project.

1 Introduction

1.1 Regularization

Regularization in machine learning can be seen as using prior knowledge to let the model be closer to what we expect. It is a method of controlling the model manually.

The famous L1-norm and L2-norm are adding penalties to the complexity of a model, in order to make the model simpler. It is a specific form of regularization according to the definition above. And it also embodies the principle of Occam's Razor.

The general form of adding regularization terms to the loss function is

$$Loss = D(\mathbf{x}, \mathbf{y}) + \lambda\Omega(\mathbf{W})$$

where $\mathbf{x}$ is the prediction vector, $\mathbf{y}$ is the ground truth vector, $D(\cdot)$ is the function which measures the difference between $\mathbf{x}$ and $\mathbf{y}$, $\Omega(\mathbf{W})$ denotes the regularization term and λ is the coefficient of it.

As for L1-norm,

$$\Omega(\mathbf{W}) = \sum_k \|W^k\|_1 = \sum_k \sum_i |W_i^k|$$

and for L2-norm,

$$\Omega(\mathbf{W}) = \sum_k \|W^k\|_2^2 = \sum_k \sum_i |W_i^k|^2.$$

1.2 Projection Shortcut

Using shortcuts in neural networks became well-known since the publishing of *Deep Residual Learning for Image Recognition*[2]. This paper introduced the residual block with identity shortcut, which is a way to prevent the network from being too deep, and to alleviate the problem of vanishing/exploding gradients.

As the authors of the paper pointed out, the residual block with identity shortcut is a specific form of a more general residual block: that with projection shortcut, showed in Figure 1. As we were interested in this generalized form, so in this project, we mainly focused on the residual blocks with projection shortcuts.

*Shanghai Jiao Tong University, sjtudyx2016@sjtu.edu.cn.

[1]With Lizhen Zhu.

[2]By K. He et al., CVPR 2016.

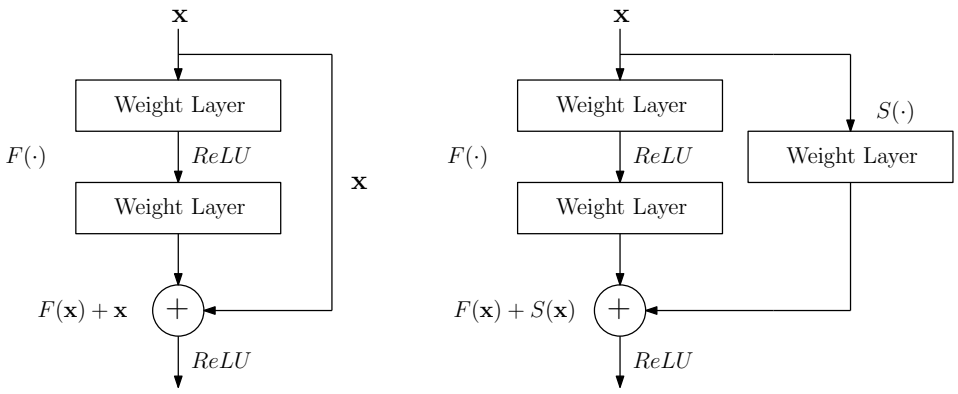

Figure 1: Residual block with identity shortcut (left) and projection shortcut (right).

2 Design and Analysis

2.1 Conventional Regularization

Neural networks use back propagation technique to update their parameters. Considering the regularization terms only, for a weight matrix W, we have

$$\Delta W = \frac{\partial \lambda \Omega(\mathbf{W})}{\partial W}.$$

2.1.1 L1-norm

$$\Delta W = \frac{\partial \lambda \sum_k \|W^k\|_1}{\partial W} = \lambda \cdot \mathrm{sgn}(W)$$

where $\mathrm{sgn}(W)$ is a matrix with the shape of W, and

$$\mathrm{sgn}(W)_{ij} = \begin{cases} 1 & (W_{ij} > 0) \\ 0 & (W_{ij} = 0) \\ -1 & (W_{ij} < 0). \end{cases}$$

So L1-norm will reduce the difference between each element of W and 0 by λ. We considered it as a more radical way of simplifying the model. Just as the outcome of our experiments showed later, L1-norm can significantly alleviate the overfitting, yet sacrifice the performance.

2.1.2 L2-norm

$$\Delta W = \frac{\partial \lambda \sum_k \|W^k\|_2^2}{\partial W} = 2\lambda W.$$

So L2-norm is a more conservative regularization method, comparing to L1-norm. It reduces the difference between each element of W and 0 proportionally. As our experiments showed, L2-norm is less effective in preventing overfitting, but keeps the performance.

2.2 Mutual Regularization

As the feature of our project, two kinds of mutual regularization were designed, which considers more than one weight matrix at a time. For the networks with projection shortcuts, a natural idea is to encourage the shortcut to be as different as possible with the original path. In this way, we can assure that adding shortcuts is meaningful to the network.

2.2.1 Orthogonal Regularization

Now consider the fully-connected residual block, in which the three are all fully-connected ones. We use W_{F1}, W_{F2} to denote the weight matrices of the two layers on the original path, and W_S to denote the weight matrix of the layer on the shortcut. Suggest this residual block has an input of a dimensions, b dimensions for the halfway result and c dimensions for the output, then W_{F1} is of size $a \times b$, W_{F_2} is of size $b \times c$ and W_S is of size $a \times c$.

Take $a = 3$, $b = 4$, $c = 5$ as an example.

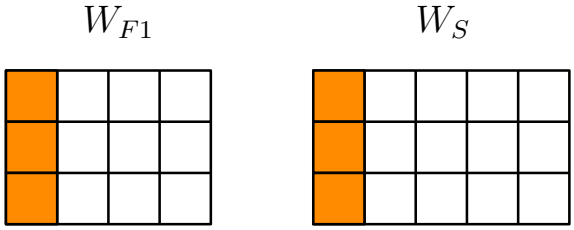

Figure 2: Two weight matrices of the example

We noticed that every column of W_{F1} and W_S is a weight vector when the two matrices are multiplied with the input. If we calculate $W_{F1}^T \cdot W_S$, then every element of the new matrix will be the inner product between a column vector in W_{F1} and column vector in W_S.

Calculating the inner product is a method to compare two vectors. And if the two vectors are orthogonal (i.e. differ a lot), the inner product will be 0. Therefore, if we want the columns of W_{F1} and W_S to be different, we can add L2-norm to the product of them. That is

$$\Omega(\mathbf{W}) = \lambda \|W_{F1}^T \cdot W_S\|_2^2.$$

When doing back propagation, take W_S as an example,

$$\Delta W_S = \frac{\partial \lambda \|W_{F1}^T \cdot W_S\|_2^2}{\partial W_S} = 2\lambda W_{F1} \cdot (W_{F1}^T \cdot W_S)$$

where $(W_{F1}^T \cdot W_S)$ measures how much the two matrices differ, and multiplying with W_{F1} means W_S will be subtracted by a certain degree of W_{F1}. So this regularization can make W_S and W_{F1} different.

2.2.2 Kernel Difference Regularization

In convolutional operations, a kernel can extract a certain type of local feature. So obviously we do not want the kernels of the original path to be similar with those of the shortcut. Therefore, we designed a new regularization which measures the similarity between two convolutional kernels.

$$\Omega(\mathbf{W}) = \lambda \|W_F - W_S\|_2^2$$

where W_F, W_S is respectively a kernel of the original path and the shortcut.

When doing back propagation, take W_S as an example,

$$\Delta W_S = \frac{\partial \lambda \|W_F - W_S\|_2^2}{\partial W_S} = 2\lambda(W_S - W_F).$$

This regularization can also make the two kernels different. If we take λ a negative value and $W_S = W_F + \varepsilon$, then $\Delta W_S = 2\lambda\varepsilon$ will be added to W_S, which enlarges the difference between the two kernels.

There was still a minor strategy when utilizing this regularization. To simplify the calculation and shorten the training time, we used grouped convolution and only calculated pairwise regularization if W_F and W_S are from the same channel or to the same channel. Figure 3 and Figure 4 shows an example in which this convolutional residual block has 2 in-channels, 4 mid-channels and 8 outchannels.

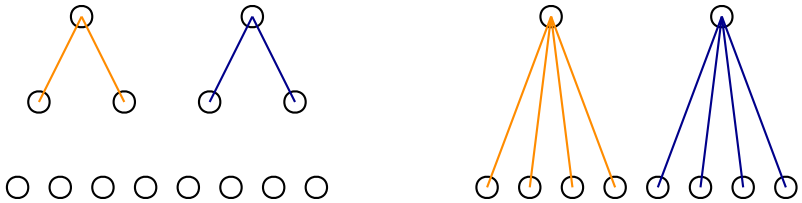

Figure 3: The kernels with the same input will be calculated their difference

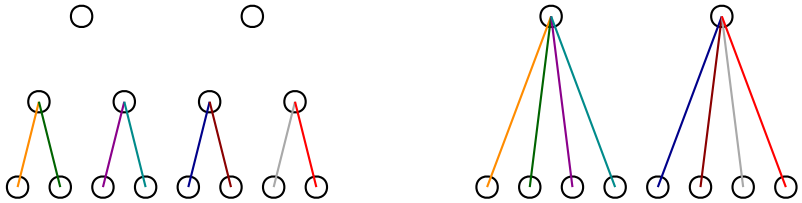

Figure 4: The kernels with the same output will be calculated their difference

In the example above, first we will choose one orange kernel from the original path (left) and one orange kernel from the shortcut (right) and calculate their difference. In this way we obtain $2 \times 4 = 8$ differences. Then for the blue ones we can also obtain 8 differences. Finally, we can obtain another 8 differences with regard to the same output channels in Figure 4.

3 Experiments [3]

3.1 Data

We chose CIFAR-10 dataset, which consists of 60000 pictures with the size of 32×32 and 3 layers (RGB). Each picture has a label indicating its category. There are 10 categories in total. To shorten the training time in order to do more experiments, we only use 20000 pictures as our training dataset, and another 1000 pictures as our validation dataset. Both dataset are chosen randomly and divided into mini-batches sized 64.

3.2 Network Structure

The structure of our network is showed as an appendix.

3.3 Training Method

- Code Framework: PyTorch.
- Optimizer: Adam[4].
- Criterion ($D(\cdot)$ function): Cross Entropy Loss.

[3] Full code can be found at *git@github.com:Ldhlwh/RegularShortcut*.
[4] From *Adam: A Method for Stochastic Optimization* by D. Kingma et al., ICLR 2015.

- Learning Rate: 0.0001.

3.4 Evaluation Method

For each time, we trained the model for 50 epochs to reach a steady situation. Then we calculated the average validation accuracy (ValAcc) and the average difference (Delta) between the training accuracy (TrainAcc) and the validation accuracy of the last 10 epochs. We took the latter as a metric of overfitting.

3.5 Result

The visualized outcomes are showed in Figure 5 and Figure 6.

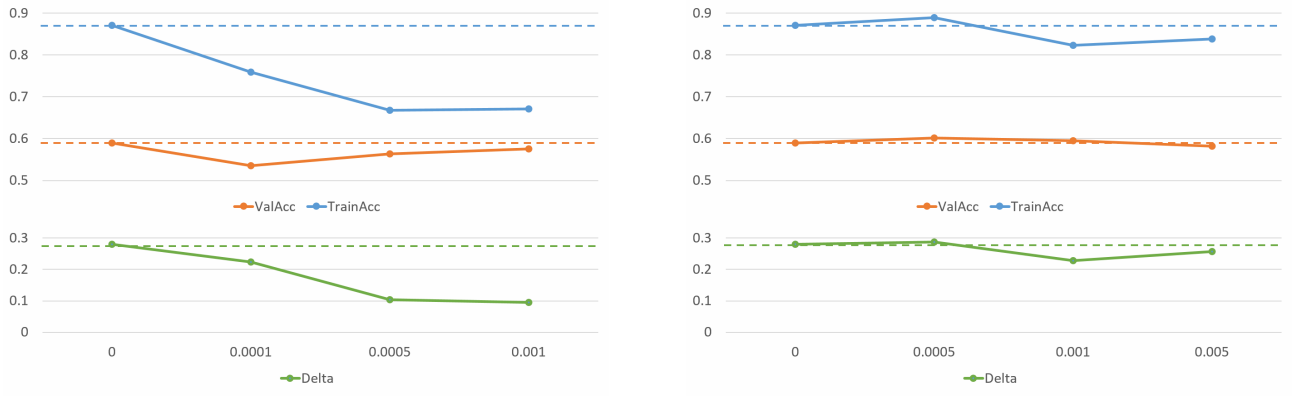

Figure 5: Left: L1-norm; Right: L2-norm

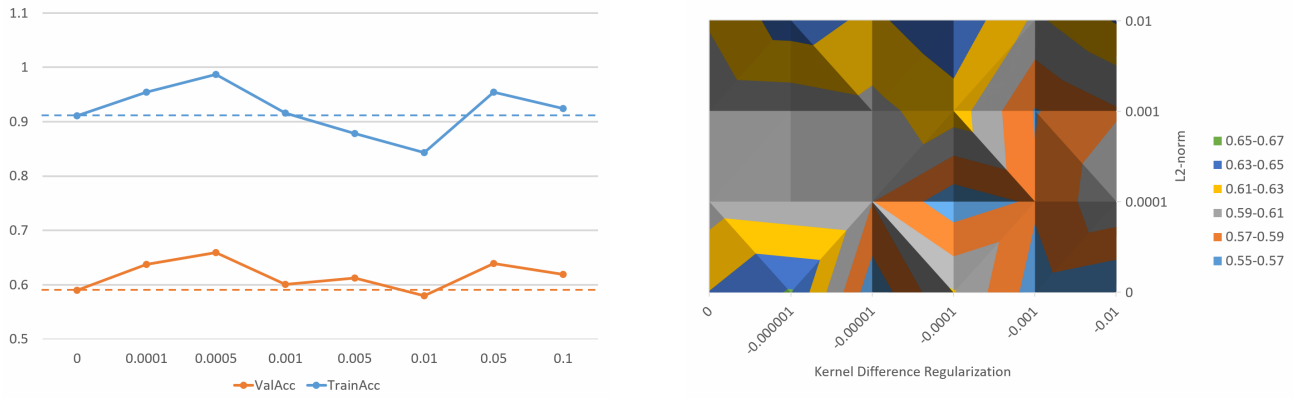

Figure 6: Left: Orthogonal Regularization; Right: Kernel Difference Regularization (ValAcc)

3.5.1 L1-norm

As the λ of L1-norm increases, the Delta decreases fast. So our assertion that L1-norm is more radical in preventing overfitting is verified.

3.5.2 L2-norm

Comparing with L1-norm, L2-norm is less effective in preventing overfitting, which we considered as more conservative. However, L2-norm improves the performance a little if λ is well chosen.

3.5.3 Orthogonal Regularization

On the contrary, orthogonal regularization does not focus on preventing overfitting, so the Delta did not decrease significantly. However, the performance improved a lot when λ was around 0.0005, and it was better than L2-norm.

3.5.4 Kernel Difference Regularization

The λ of kernel difference regularization should be chosen as a negative value. To prevent the loss to be negative, we combined this regularization with L2-norm. The figure has different colors for the performance. For example, the green area is the best (around L2-norm: 0, Kernel Difference Regularization: -1×10^{-6}), and then comes the dark blue area, the yellow area and the grey area and so on. The validation accuracy with no regularization is around 0.59, so the colors mentioned just now are all better results.

Furthermore, when considering the Delta as well (not showed in the figure), we found that when λ of L2-norm is around 0.01, the overfitting problem alleviated much (with average Delta around 0.16). Yet the performance still remained good and steady.

4 Conclusion

During this project, we had a new insight into the essence of regularization. Also, we learned that different kinds of performance-enhancing regularization could be designed for specific machine learning models and tasks. As the saying goes, there might be routines in machine learning, yet the real-world tasks don't. Only when we adapt to those specific tasks will we achieve our goals.

5 Acknowledgment

- It was a pleasure to do research with my teammate Lizhen Zhu. She contributed a lot to this project, with two well-prepared presentations. I would never finished this project without her work.
- I am also appreciated for Yuda Fan's comprehension on regularization, Yutong Xie's advice of measuring the difference of two vectors with orthogonal method and all the suggestions from other classmates.

6 Appendix

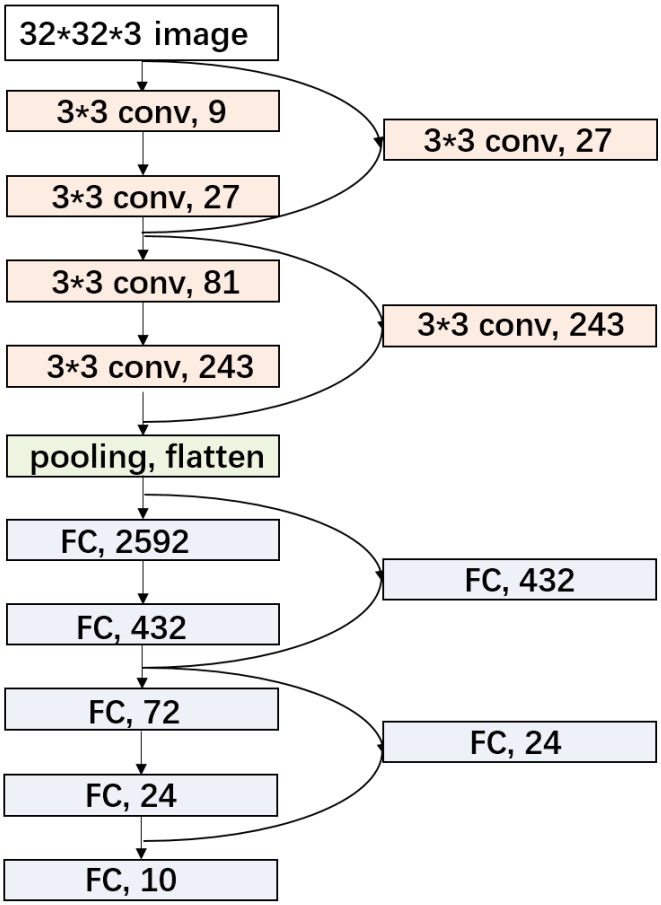

Figure 7: The structure of our network