# Assignment 1 CNN for image recognition

Zheyu Lin

## 2 Create your own ResNet work shown in Table 1 (19p)

### 2.2 Create this model in Table 1 (8p)

See my model in MyResnet.py

### 2.4 Use the downloaded training dataset (see Section 1) to train your created model

Since the size of images from CIFAR-10 is 32x32, I resized the size of images to 224x224.
See the details in test.py.

### 2.5 Report the final accuracy (10,000 steps) of training and testing for the CIFAR-10 dataset (2x1=2p).

The final accuracy of training is 100%, and the final accuracy of testing is 86.33%.
The results are based on 9765 iterations (25 epoch times, 128 batch sizes).
See the details in test.py.

### 2.6 Visualize the learning progress. Accuracy (1p) and Cross_entropy (or loss function) (1p).

See the details in test.py. and Myvisdom.py.


Figure.6.1 Validation Accuracy

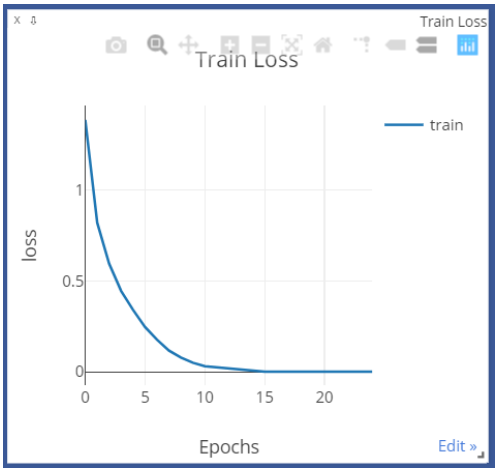


Figure.6.2 Train Loss

**2.7 Filter visualization (3p).**

See the details in test_filter.py

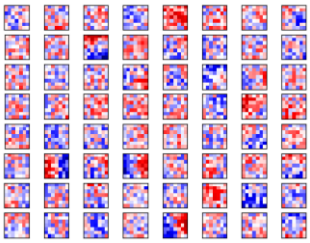

**Figure.7.1. Channel_0**

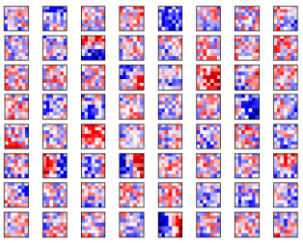

**Figure.7.2. Channel_1**

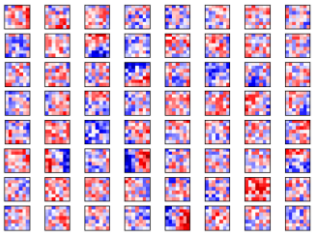

**Figure.7.3 Channel_2**

**2.8 Feature mapping visualization (4p).**

**Figure.8.1. An image from validation set**

Firstly, I choosed an image(Figure.8.1.) from the validation set, and get the feature maps of the layer 'Conv1'(Figure.8.2.) and the last layer in the block Conv5_x(Figure.8.3.).
See the details in test_reply.py.

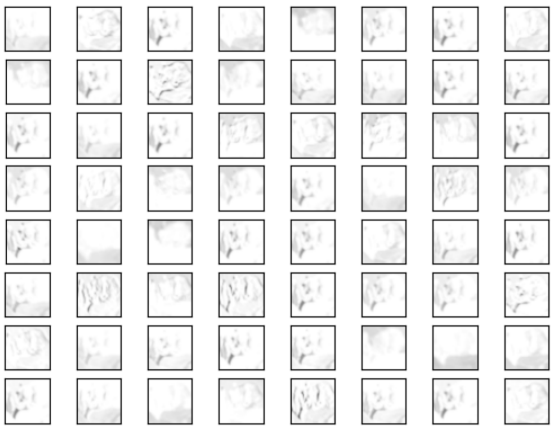

**Figure.8.2. Feature maps of the layer 'Con1'**

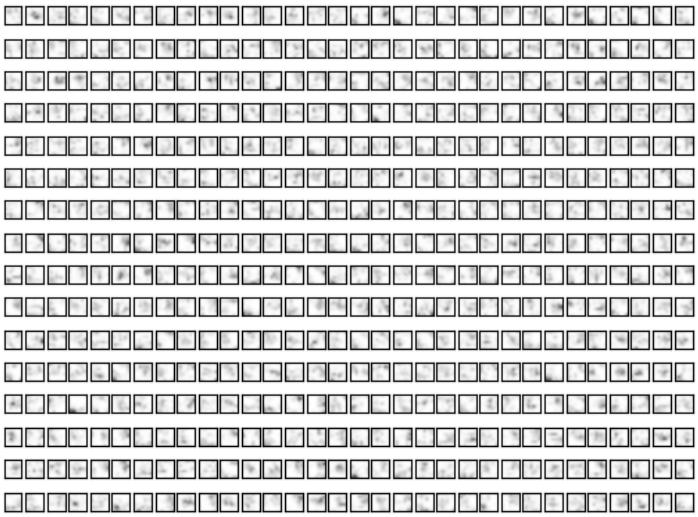

**Figure.8.3. Feature maps of the layer 'Conv5_x'**

In order to choose a feature map which had the highest activations in each layer, I showed activations of each layer through column charts(Figure.8.4.1. and Figure.8.4.2.).
See the details in [test_image.py](test_image.py).

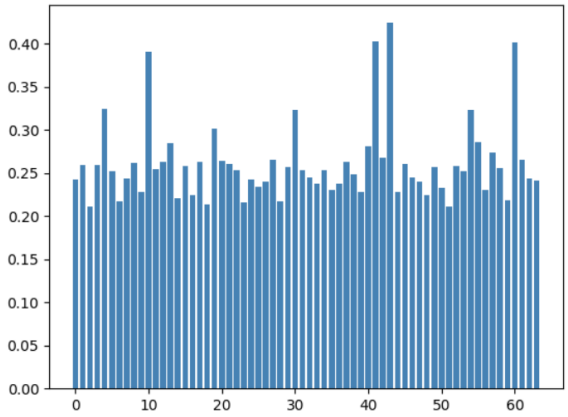


**Figure.8.4.1. Column chart of the layer 'Con1'**

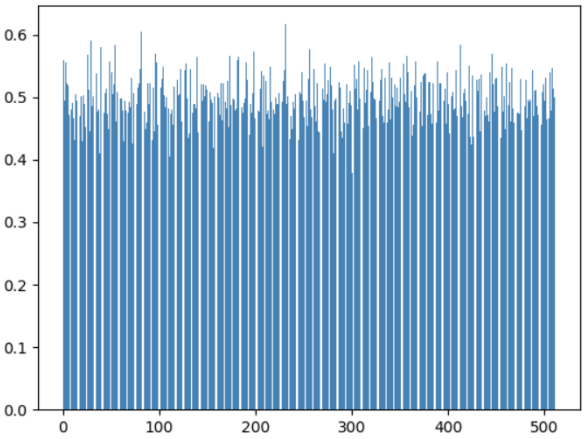


**Figure.8.4.2. Column chart of the layer 'Conv5_x'**

Since the Resnet has the shortcut operation, it`s hard to get reconstructed patterns by deconvolution for a feature map. Essentially, the deconvolutioin operation is a part of the processes of backpropagation. Therefore, the patterns can be reconstructed directly by backpropagation.

Finally, I choosed the 43th feature map in 'Conv1' layer and the 231th feature map in the last layer and get the reconstructed patterns of both feature maps(Figure.8.5.1. and Figure.8.5.2.).

See the details in test_reconimage_conv1.py and test_reconimage_conv5.py.

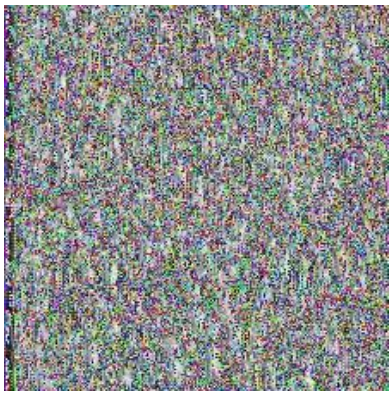

**Figure.8.5.1 The reconstructed patterns of the layer 'Con1'(the 43th feature map)**

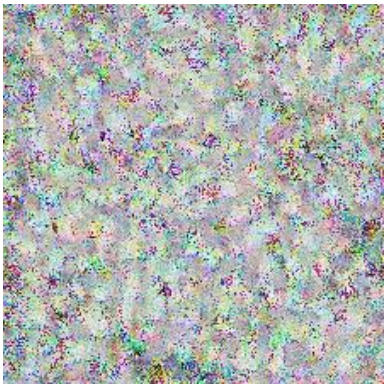

**Figure.8.5.2 The reconstructed patterns of the layer 'Conv5_x'(the 231th feature map)**

## 3 Feedback (1p)

- I spent more than 48 hours for this assignment.
- This course is pretty cool, but I am not so familiar with deep learning. So it will take me much time to preview some basic knowledge.
- Emmm...this assignment really helped me develop some skills of deep learning. Since I am not familiar with some basic knowledge of image processing, it really took me a lot of time to learn some basic skills of image processing.
- Well, maybe we should pay more attention on some basic principles of AI. And I`d also like to hear about the history of AI and what we can learn from it.

## Reference

MyResnet.py and test.py
[1] https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py
[2] https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

test_filter.py and test_reply.py
[3] https://github.com/grishasergei/conviz

test_reconimage_conv1.py and test_reconimage_conv5.py
[4] https://github.com/utkuozbulak/pytorch-cnn-visualizations
[5] https://mathpretty.com/10475.html