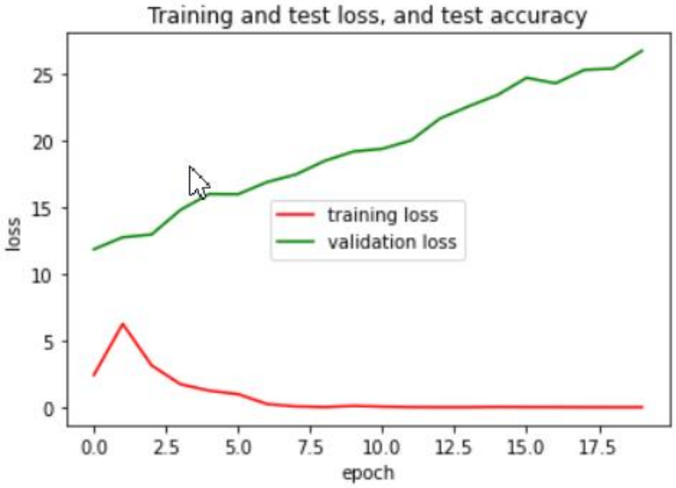
# COMP5623 Coursework on Image Classification and Visualizations with Convolutional Neural Networks – ImageNet10

| | |
|---|---|
| Name | **Kunhao Liang** |
| Student username & ID | **mm20kl / 201469850** |

## QUESTION I [55 marks]

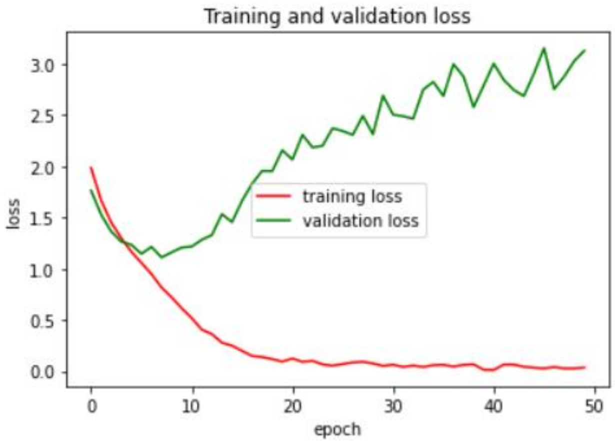
## 1.1 Single-batch training [16 marks]

| |
|---|
| 1.1.1. Display graph 1.1.1 (training & validation loss over training epochs) and briefly explain what is happening and why. [4 marks] |



The training loss decreases over epochs and down to zero at the end of the training,  while the validation loss keep increasing in the whole training process. It shows that the model is overfitting because it can achieve good performance at training set but fails to fit unseen data. This is because a single fully-connected layer is unable to learn the complicated features of image to obtain a good generalization.

1.1.2 Display graph 1.1.2 (training & validation loss over training epochs, with modified architecture) and explain how and why it shows that the model is overfitting the training batch. [8 marks]



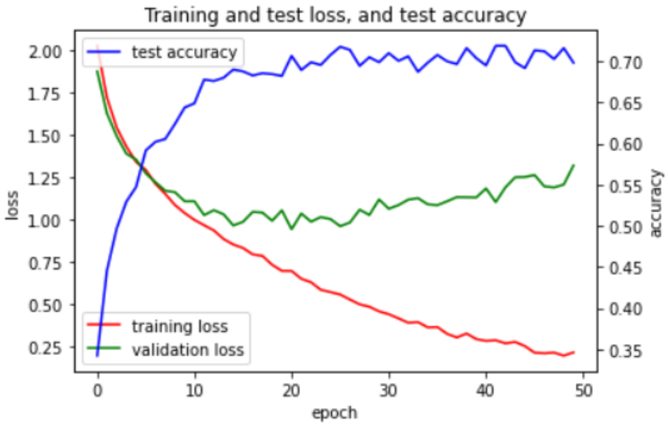
The training loss observes a downward trend, decreasing gradually over epochs, and it is asymptotic to zero at the end of training. The validation loss also drops at the first 10 epochs, but increases stably after that. It indicates that the model can perform well on the training set, but it has poor generalization to other data. Therefore, the model is overfitting.

1.1.3 Fill in table 1.1.3 (your adjusted architecture after single-batch training), adding rows and columns as necessary. [4 marks]

| Input channels | Output channels | Layer type | Kernel size | Padding | Stride |
|---|---|---|---|---|---|
| 3*128*128 | 32*128*128 | convolutional layer | 3 x 3 | 1 | 1 |
| 32*128*128 | 32*64*64 | pooling layer | 2 x 2 | / | 2 |
| 32*64*64 | 64*64*64 | convolutional layer | 3 x 3 | 1 | 1 |
| 64*64*64 | 64*32*32 | pooling layer | 2 x 2 | / | 2 |
| 64*32*32 | 128*32*32 | convolutional layer | 3 x 3 | 1 | 1 |
| 128*32*32 | 128*16*16 | Pooling layer | 2 x 2 | / | 2 |
| 128*16*16 | 256*8*8 | Convolutional layer | 3 x 3 | 1 | 1 |
| 256*8*8 | 256*8*8 | Pooling layer | 2 x 2 | / | 2 |
| 256*8*8 | 256 | Fully-connected layer | / | / | / |
| 256 | 64 | Fully-connected layer | / | / | / |
| 64 | 10 | Fully-connected layer | / | / | / |

## 1.2 Fine-tuning on full dataset [18 marks]

1.2.1 Display graph 1.2.1 and indicate what the optimal number of training epochs is and why. [4 marks]



From my perspective, the optimal number of training epochs is between 20-30. As it can be seen, after training 30 epochs, the training loss still keep decreasing, but the validation loss increases with test accuracy fluctuated at 70% until the end of training. This shows that the model is becoming more overfitting. Hence, in order to obtain a relatively good generalization of model, we should stop training at about 30 epochs.

1.2.2 Describe in detail your fine-tuning process on the complete dataset, including any adjustments you made to the network or training process to increase prediction accuracy. Explain why these adjustments increased accuracy. [10 marks]

Adjustment:

- Regularization(Batch Normalization, Dropout)
- Using data augmentation

Batch normalization is added to normalize the convolved features outputted by convolutional layers to take on a similar range of values before being sent to the activation function. It has the effect of regularization, which helps prevent overfitting and improve the generalization of the model. Therefore, the model can have a higher accuracy on the test set. Dropout is also used in the experiment. It randomly eliminates a specific number of neurons of network during training, making a layer with different structures in each training. Therefore, it can improve the robustness of model, further increasing model accuracy. In my experiment, these two regularization methods are both added in the order shown below:

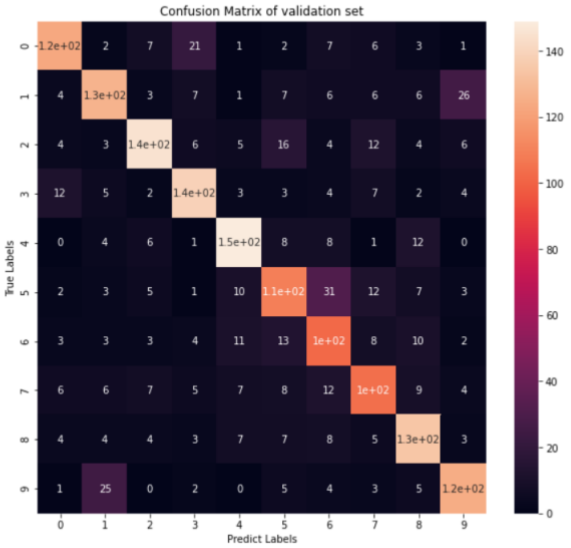
Conv -> Batch Normalization -> Activation Function(ReLU) -> Dropout -> Pooling layer

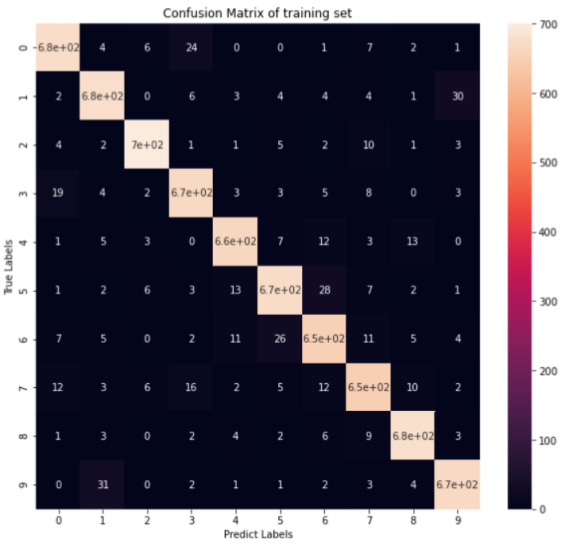
Linear -> Dropout -> Batch Normalization -> Activation Function(ReLU)

Then we can propose data augmentation which includes horizontal flip of image with a given probability and rotation of image with specific angle. Finally, by applying regularization methods and data augmentation mentioned above, the average test accuracy increases by 7% from 63% to 70% after training enough epochs.

1.2.3 Display two confusion matrices 1.2.3 (one each for complete validation set and complete training set) for your final trained model and interpret what is shown. [4 marks]

The confusion matrix for validation set(left) and training set(right) are presented below:





As we can see from validation set, the model is not good at distinguishing objects of label 1 from label 9, as 35 images whose true label is 1 are misclassified as 9. A similar patterns also can be seen between label 5 and label 6. For the confusion matrix of training set, most objects are classified correctly, since the model is overfitting and it has good performance on training set.

## 1.3 Evaluation and code [21 marks]

1.3.1 Please include *[my_student_username]_test_preds.csv* with your final submission. [8 marks]

1.3.2 Please submit all relevant code you wrote for Question I in Python file *[my_student_username]_q1.py*. No need to include the config or ImageNet10 files. [13 marks]

*No response needed here.*

## QUESTION II [45 marks]

## 2.1 Preparing the pre-trained network [20 marks]

2.1.1 Read through the provided template code for the AlexNet model *alexnet.py*. What exactly is being loaded in line 59? [2 marks]

A: In line 59, a pre-trained AlexNet model is loaded.

2.1.2 Write the code in *explore.py* after line 50 to read in the image specified in the variable `args.image_path` and pass it through a single forward pass of the pre-trained AlexNet model. [5 marks]

2.1.3 Fill in function `extract_filter()` after line 84 extracting the filters from a given layer of the pre-trained AlexNet. [4 marks]

2.1.4 Fill in function `extract_feature_maps()` after line 105 extracting the feature maps from the convolutional layers of the pre-trained AlexNet. [6 marks]

Please submit all your Question II code in a Python file *[my_student_username]_explore.py*.

*No response needed here.*

2.1.5 Describe in words, not code, how you ensure that your filters and feature maps are pairs; that the feature maps you extract correspond to the given filter. [3 marks]

In my code, I plotted one filter at each channel, and plotted features maps of all channels. Then, the number of feature maps is the same as the number of filters, and they are shown with the same order. Therefore, each feature map and filter can be paired easily.

## 2.2 Visualizations [25 marks]

2.2.1 For three input images of different classes, show three pairs of filters and corresponding feature maps, each from a different layer in AlexNet. Indicate which layers you chose. For each pair, briefly explain what the filter is doing (for example: horizontal edge detection*)* which should be confirmed by the corresponding feature map. [15 marks]

Image #1, class: ___Fish_______

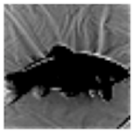
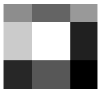
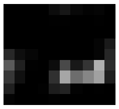
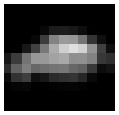
| | Filter | Feature map | Brief explanation |
|---|---|---|---|
| Early layer 0 | |  | Sharpen the image |
| Intermediate layer 6 |  |  | Detect the features at the center of image |
| Deep layer 10 | |  | Detect the features at the center of image |

Image #2, class: ___Bird________

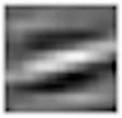
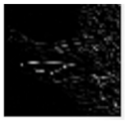
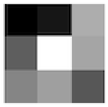
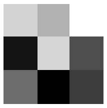
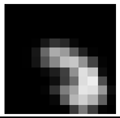
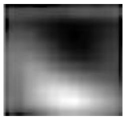
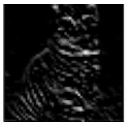
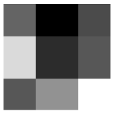
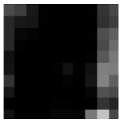
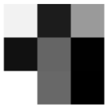
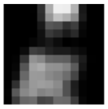
| | Filter | Feature map | Brief explanation |
|---|---|---|---|
| Early layer 0 |  |  | Horizontal edge detection |
| Intermediate layer 6 |  | | Detect the features at the center of image |
| Deep layer 10 |  |  | Detect the features at the right bottom corner |

Image #3, class: ___Cat_______

| | Filter | Feature map | Brief explanation |
|---|---|---|---|
| Early layer 0 |  |  | Horizontal edge detection |
| Intermediate layer 6 |  |  | Detect the features at the right side of image |
| Deep layer 10 |  |  | Detect the vertical features of image, and the shape of cat is shown |

2.2.2 Comment on how the filters and feature maps change with depth into the network. [5 marks]

The filters at the start of network are basic and used to detect edges. As it goes deeper, the size of filters shrinks, and filters become more complex which enables them to detect more sophisticated features. This pattern can be also observed on feature maps. At the early layer of network, the feature maps can present the outline of images which is informative, while the feature maps outputted by filters at deep layer are too abstract to understand.

Marks reserved for overall quality of report. [5 marks]

*No response needed here.*