**Assignment:2**

**DEEP LEARNING**

**AO PENG**

## Part 1

In this part, we randomly created matrix A and B. Learning rate and epochs were defined as well which are 0.001 and 1000 respectively. The backpropagation process is in fitData function.

1. The back-propagation process is based on the chain-rule of derivative.

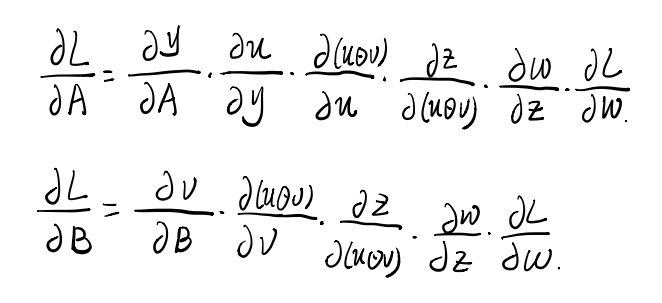


Figure.1 The derivative part

1. After getting the derivative of every step, we should transpose the derivative.

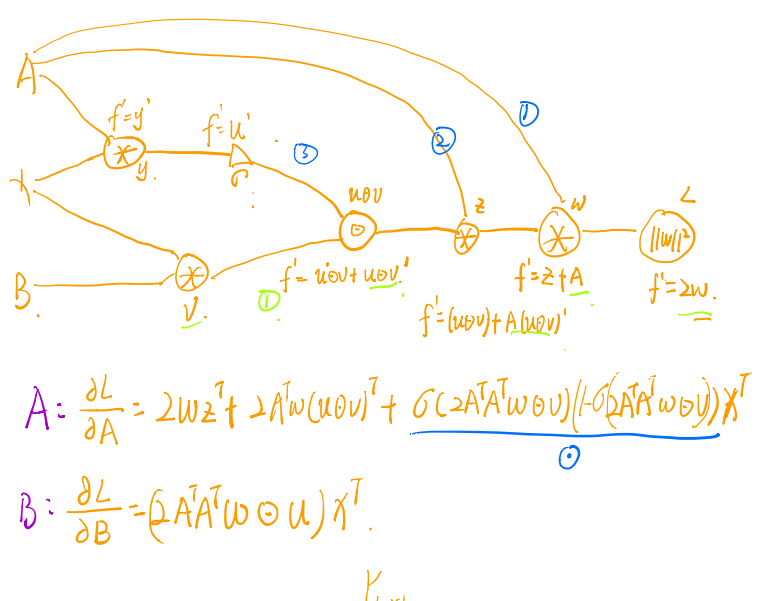


Figure. 2 back-propagation

1. The final step is to sum all the derivatives which are already transposed.

#### Result:

Learning rate is 0.001

1. K=2,

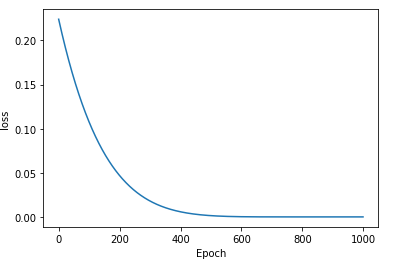


Figure. 3

1. K=3

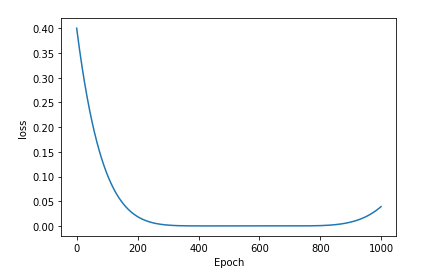


Figure. 4

1. K=10

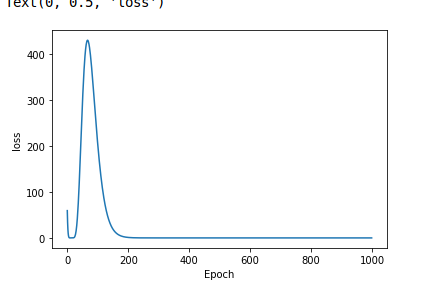


Figure. 5

Testing with Learning rate is 0.1

1. K=2

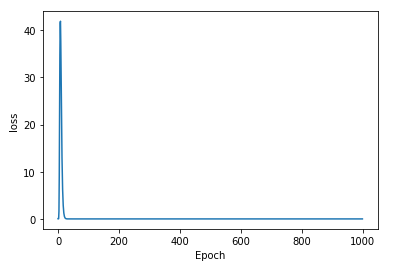


Figure. 6

1. K=3

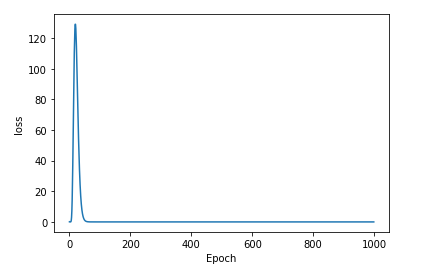


Figure. 7

1. K=10

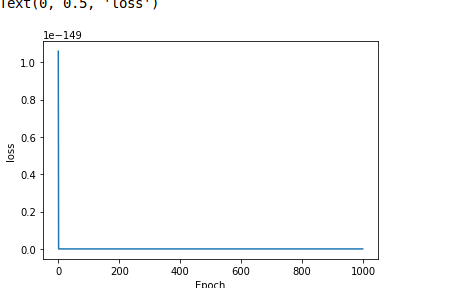


Figure. 8

### Conclusion

1. With the increase of K, the loss will increase rapidly and then drop quickly.
2. The learning rate also plays an important role in gradient descent. Big learning rate will result in that the model cannot converge. Too small learning rate will take much time for training.

### Problem

Our model is unstable, we cannot decrease the loss. We checked the derivative and our code and found nothing.

## Part 2

In these two models, we can see the shape of matrix W and matrix (A+B)C are the same. From the purpose of softmax function which is to choose the highest probability of classes, the final results of these two models are same which are a group of probability for different classes and the sum of them equal to 1. That means the distribution of the 2 models are the same as well.

In this part, we use an example to further prove that this assumption is correct with random matrix W and (A+B)C. Finally, we find the means and variances of them are almost same which proves that they are the same models.

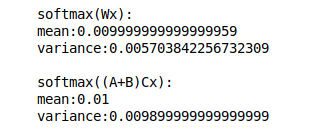


Figure. 9 example of h1 and h2

## Part 3

## In this part, we create 3 different type model based on softmax, MLP and CNN. For each kind of model, we compare the performance of the model with dropout and batch normalization or not.

### 1. Softmax

For softmax part, we use 7 examples to get more details on how the number of filter, dropout and batch normalization effect our model.

The first one(with 6 filters) is the most simple model in Question 3 part so that the performance is not good enough and there is an overfitting problem.

The second one(with 128 filters) which we changed the number of filter in Dense layer is worse than the former one. The curve shows there is a fluctuation which means this model does not have a good generalization ability. In my opinion, too many filters would result in the missing of features of data. That is the reason why the accuracy of that is lower.

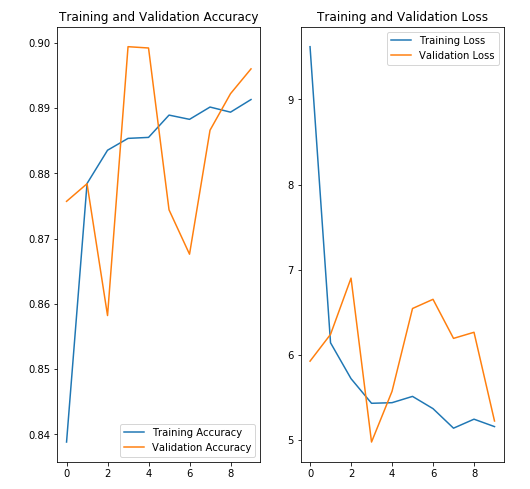
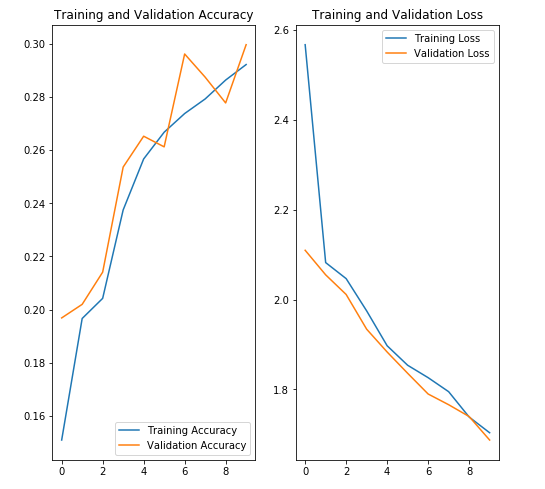


Figure. 10 6 filters(left) and 128 filters(right)

#### 1.1 With Dropout

We add dropout layer for third model with 6 filters which is same as the first one, which is good for comparing them. It has no good effect on the model. However, after I changed the value of dropout from 0.2 to 0.1, the overfitting problem disappeared, although the performance of generalization was not good either.We then increased the value to 0.5. The final results became worse.



Figure. 11 without dropout(left) and with dropout(right)

#### 1.2 With Batch Normalization

Based on the purpose of BN which is it can mitigate the problem of internal covariate shift , we use a worse model which has 128 filters in dense layer for testing. Actually, if we just add BN in our model, it cannot get an improvement instead get a worse result. However, if we combine BN and dropout, the performance of our model will be better.

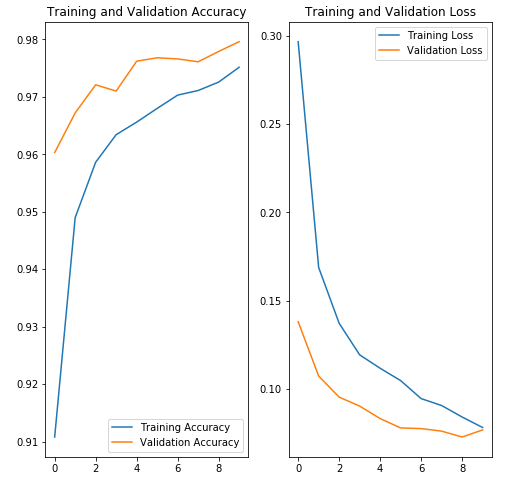
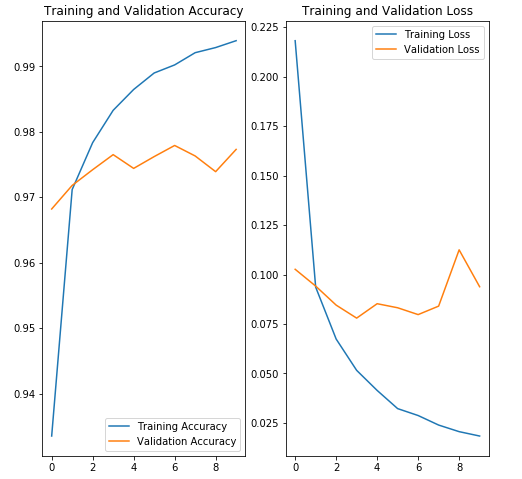


Figure. 12

#### 1.3 Summary

We have tested 3 sizes of filter in using different techniques and different parameters and draw the testing loss and accuracy tables. So it is clear to see that BN plays an important role for softmax model and all these parameters can affect the performance on the model.

**Tabel. 1 Testing loss**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Initial | With Dropout(0.2) | With Dropout(0.1) | With Dropout(0.5) | With BN |
| 6 | 1.6875 | 1.8070 | 1.7809 | 2.1415 | 0.3083 |
| 128 | 5.2239 | 0.2721 | 0.2926 | 0.3996 | 0.0991 |
| 28 | 0.3527 | 0.3794 | 0.3497 | 1.2859 | 0.1213 |

**Tabel. 2 Testing accuracy**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Initial | With Dropout(0.2) | With Dropout(0.1) | With Dropout(0.5) | With BN |
| 6 | 0.2996 | 0.2812 | 0.2483 | 0.1817 | 0.9124 |
| 128 | 0.8960 | 0.9480 | 0.9489 | 0.9148 | 0.9756 |
| 28 | 0.9201 | 0.9046 | 0.9118 | 0.5135 | 0.9648 |

### 2. MLP

MLP is a way based on softmax. It can get better results than softmax one if the both model have the same number of filter.If the number of dense layer is proper, the more layers,the better model. If the number is too big, there will be overfitting problem.

We also use the 6 filters in Dense layer in the first MLP model as an example. In the first model we just add 2 dense layers where we just got a not good result and we then add 4 dense layers, the performance becomes better. With the proper increase of number of filters, the performance is better.

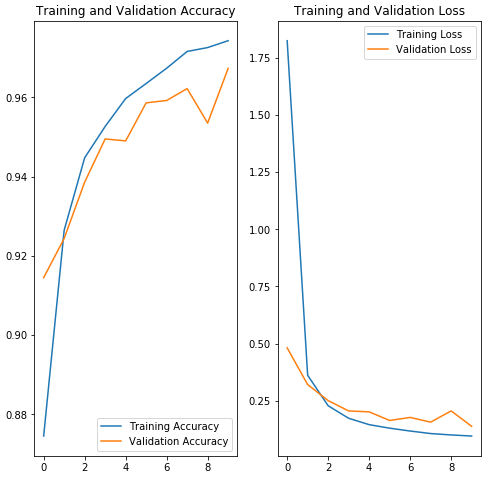
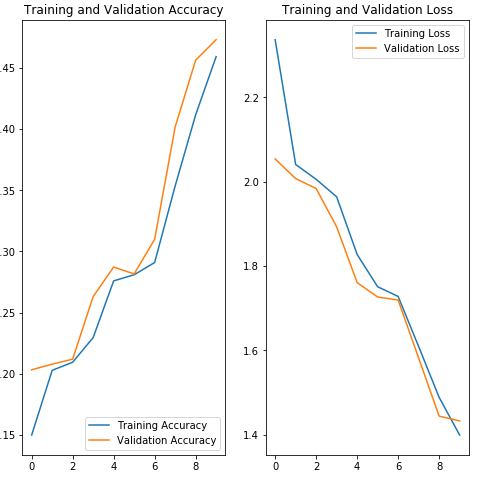


Figure.13. 6 filters(left) and 128 filters (right)

#### 2.1 With Dropout

In this part, we use a 4-layer structure network for comparison. We can find for every model, adding dropout layer makes the model worse than the model without dropout. This proves that dropout cannot still work as it is a way by randomly dropping out data from output unit to decrease the probability of overfitting. So if we have scarce data and we use dropout, it will cause a bad model. From the figure ,there is no overfitting problem so it can prevent overfitting.

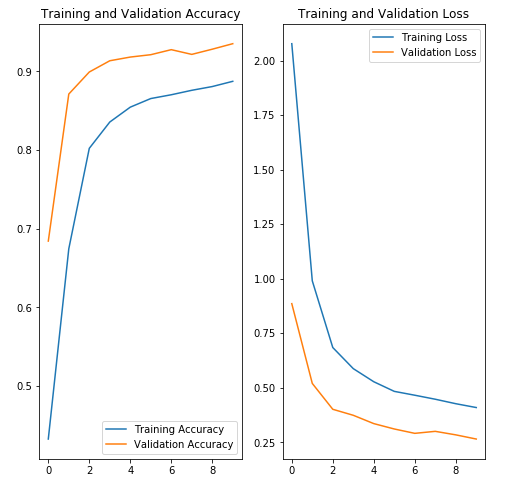


Figure. 14 56 filter(left) and 128 filters(right)

#### 2.2 With Batch Normalization

For every model, after adding BN layer, the performance has a great improvement. BN can also decrease the number of training which means it can converge faster. There is little difference among these three models. In addition, all models have the overfitting problem.

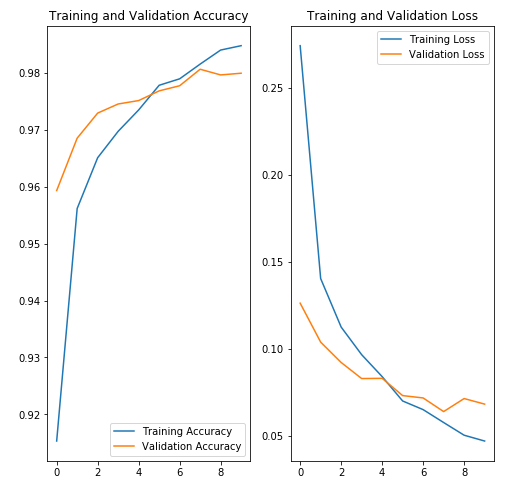
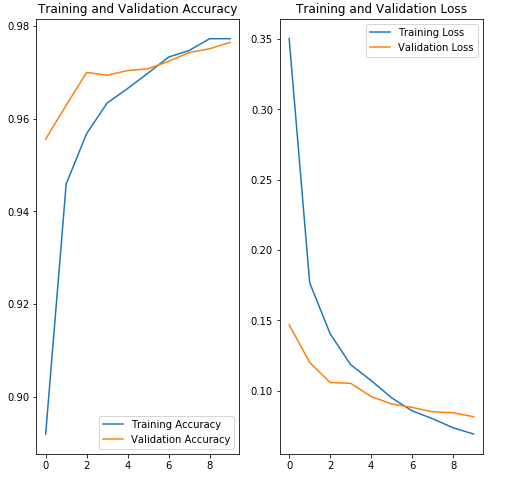


Figure. 15 56 filters(left) and 128 filters(right)

#### 2.3 Summary

The data in the tables are collected from 4 dense layers model.

We also tested an extra sample with 4 dense layers with 56 filters. In this model, we used both BN and dropout techniques. The final accuracy is not good as the model without combining these two ways, while there is no overfitting in this model. So the dropout definitely can deal well with overfitting.

**Tabel. 3 Testing loss**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial | With Dropout | With BN |
| 6 | 1.4325 | 1.5616 | 0.3530 |
| 128 | 0.1145 | 0.1856 | 0.0684 |
| 56 | 0.1272 | 0.2650 | 0.0816 |

**Tabel. 4 Testing accuracy**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial | With Dropout | With BN |
| 6 | 0.6573 | 0.3844 | 0.8967 |
| 128 | 0.9701 | 0.9544 | 0.9799 |
| 56 | 0.9673 | 0.9352 | 0.9765 |

### 3. CNN

At first, we just use 512 filters in convolutional layers, while the results are terrible. The model cannot fit data and converge. We realize too many filters lead to the missing of features. So we change the number of filters to be smaller and the new model could fit data perfectly.

From the Figure, we can see the performance of CNN model is good as CNN can collect the features of data efficiently. However, it is easier to result in overfitting for the model has more filters.

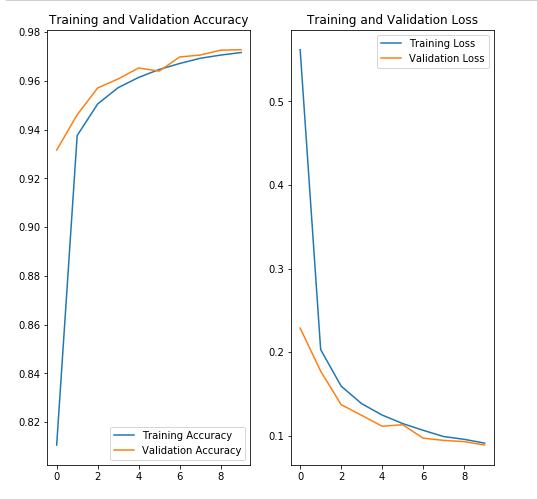


Figure. 16 2\*2\*6 kernel size(left) and 2\*2\*32 kernel size(right)

#### 3.1 With Dropout

From the above part, we know the bigger kernel can result in overfitting easier. So we add dropout layer for that model. From the figure, we can see it has a good effect on decrease overfitting. The accuracy of model with dropout is higher than that without dropout as well.

In addition, we change the value of dropout. It still workes and the performance of them are similar, which are 0.9890 with 0.1 and 0.9882 with 0.2.



Figure. 17 Without dropout(left) and with dropout(right)

As we learned in the lecture, dropout definitely can help to solve overfitting problem while if there is no that problem and we need enough data for training, it will result in bad performance as dropout process reduces randomly a part of output data. The value of dropout means the percentage of output unit which would be killed.

In my opinion, I think that’s the reason why different values of dropout which can result different performance on our dataset. Convolutional layer will lose the features of data, so if the kernel of this layer is big, it will lead to the Incompleteness of model, that is underfitting. Model doesn’t learn the feature which will result in poor performance on training and testing process.

To figure out the reason of poor performance of the model, I reviewed the lecture and rebuild the model in the convolutional layer. I decreased the numbers of kernel in that layer in which the new numbers are 6, 12 and 12 respectively, as too many convolutional processing will result in the missing of features of data, and so does the processing of dropout. The new model’s performance is good enough and the loss is 0.0501 and the accuracy is 0.9829 in testing process.

From figures, we can see the CNN models with drop out can also get good performance while it needs more number of iterations required to converge compared to the way without dropout.

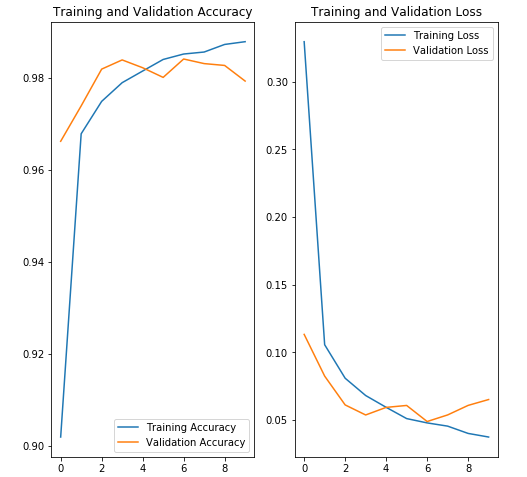
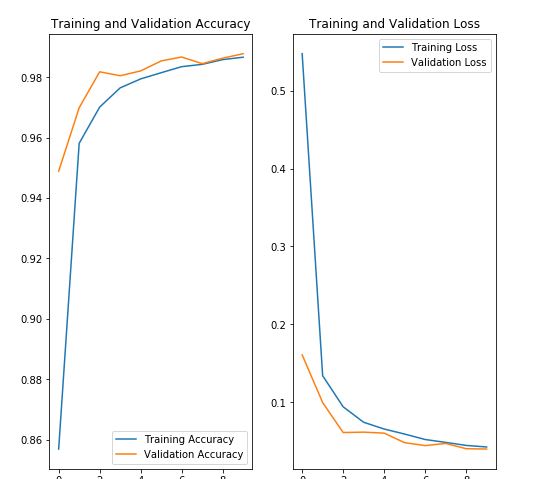


Figure. 19 With dropout and without dropout

#### 3.2 With Batch Normalization

If we just add BN layer, the performance is not good enough. So we also combine BN and dropout in CNN model to compare that without dropout. We use 2\*2\*32 size kernel to test it.

We can know the CNN model with the combination of BN and dropout would get a good generalization ability.



Figure. 20 Without dropout(left) and with dropout(BN)

#### 3.3 Summary

The data in the tables are collected from 4 dense layers model.

We also tested an extra sample with 4 dense layers with 56 filters. In this model, we used both BN and dropout techniques. The final accuracy is not good as the model without combining these two ways, while there is no overfitting in this model. So the dropout definitely can deal well with overfitting.

In addition, we find a problem that we cannot use 4\*4 size kernel in CNN model.

**Table. 5 Testing loss**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial | With Dropout | With BN |
| 2\*2\*6 | 0.0888 | 0.2254 | 0.0824 |
| 2\*2\*32 | 0.0486 | 0.0368 | 0.0411 |
| 3\*3\*6 | 0.0880 | 0.3392 | 0.0693 |

**Table. 6 Testing accuracy**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial | With Dropout | With BN |
| 2\*2\*6 | 0.9728 | 0.9508 | 0.9747 |
| 2\*2\*32 | 0.9862 | 0.9882 | 0.9910 |
| 3\*3\*6 | 0.9727 | 0.9318 | 0.9792 |

### 4. Conclusion

1. For every type of model, batch normalization technique is the most efficient way to increase the generalization ability of model among these 3 techniques
2. Dropout definitely can decrease the overfitting problem, but it is limited. If the data set is scarce or the original model is good enough, we do not need to use dropout.
3. Among these 3 different models, the performance of CNN is best which means it has a good generalization ability.