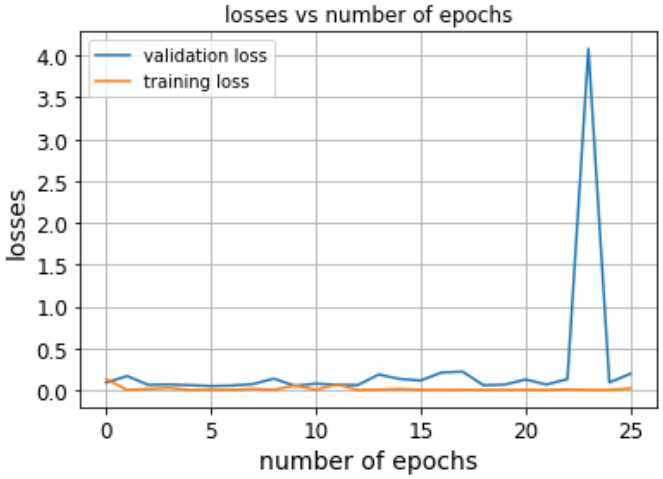
ECE 1504: Assignment 2
Neural Networks

Lei Hua Huang 1001538204
Yizhou Liu 1001139822
November 23, 2019

1 Deep Learning

1.2.2 Training

After training the designed neural network in Part 1.2.1on reduced version of MNIST, we get 98.87% accuracy on the test set, which is not bad. It takes 25 iterations to converge.



1.2.3 Tuning Hyperparameters

After training the neural network in part 1.2.1 with two given set of hyperparameters, we get 98.81% accuracy on the test set for set 1, and 97.88% accuracy on the test set for set 2. The corresponding iterations they take to converge is 33 and 111 respectively.

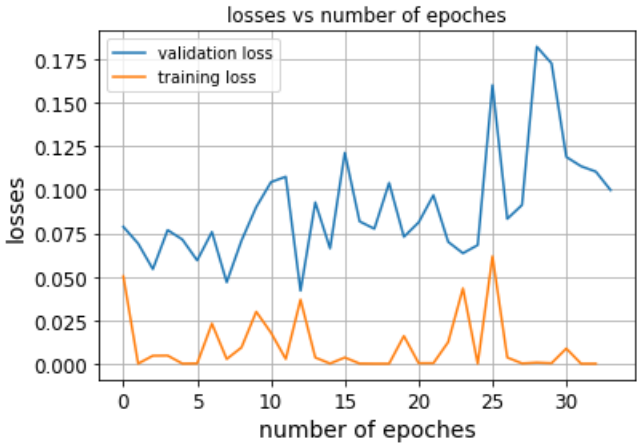


Figure1.2.3a loss functions vs number of epochs for set 1

1.2.4 Batch Normalization

Momentum in batch normalization helps reducing the noise in gradient update term and therefore helps to improve the speed of convergence to the desired optimal value. The momentum updates mean and variance by using an exponential moving average with smoothing factor of momentum.

After training different neural networks with momentum of { 0.85, 0.9, 0.95, 0.99}, the best accuracy we can achieve is 99.36% when momentum = 0.9 and 0.99. By observing the result in the table below, we may conclude that the model with momentum = 0.9 performs better in terms of precision on the training set and validation set since the number of iterations it takes is less than for momentum = 0.99.

Momentum	Accuracy	Iterations
0.85	99.32%	33
0.9	99.36%	33
0.95	99.32%	36
0.99	99.36%	36

1.2.5 Dropout

Adding dropout to the model in 1.2.3 results in the output in the table 1.2.5a below. In terms of cross entropy, the model with dropout rate=0.1 performs better, while the model with dropout rate=0.3 performs better in terms of the precision on training and validation set. The model in 1.2.4 with the momentum whose test accuracy is the best is when momentum = 0.9. Then we may add dropout to this model. According to the tables of result below, we can see the accuracy scores of different dropout rates with momentum happen to be the same. In this case, we may conclude that adding the momentum to the models with dropout rates does not make any sufficient changes on accuracy. However, it makes the speed of convergence increase by a significant percentage. Therefore, the model with dropout rate=0.1 with momentum=0.9 has better overall performance in terms of both precision on the training set and validation set and number of iterations. And the best accuracy we can achieve on the test set is 99.03%.

Dropout rate(without momentum)	Accuracy score	Number of Iterations
0.1	0.9903	52
0.3	0.9837	29

Table 1.2.5a result of adding dropout for the model in 1.2.2

Dropout rate(with momentum)	Accuracy score	Number of Iterations
0.1	0.9837	29
0.3	0.9837	29

Table 1.2.5b result of adding dropout for the model in 1.2.4

1.2.6 Impact of Regularization Methods

By observing these approaches for regularization, we may conclude that adding a momentum is the most helpful approach. It would sufficiently reduce the noise thereby

improve the speed of convergence and achieve higher accuracy. By doing this, we may achieve an accuracy of 99.36%.

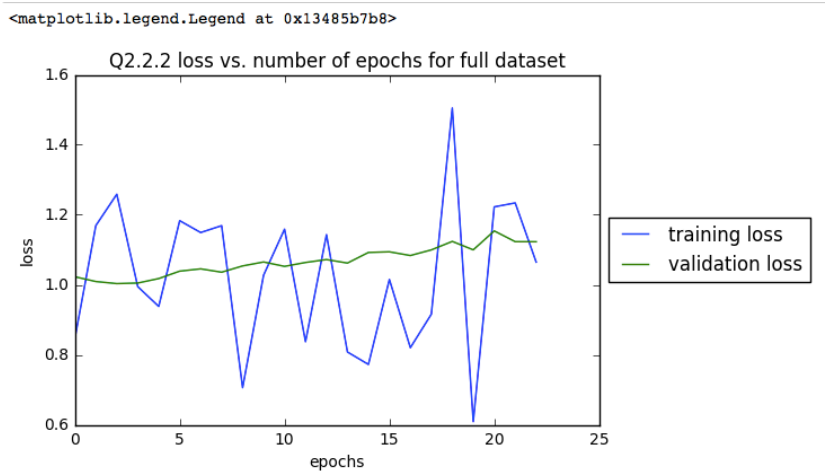
In comparison with the results with part 1.2.2 where no regularization are applied, we may find that regularization helps improving the accuracy. The accuracy scores for the models with regularization are slightly better than the model without regularization. However, it takes fewer iterations to achieve the best accuracy than the models with regularization, which indicates the model complexity for the regularized models may be higher than the model in 1.2.2.

2. Transfer Learning

2.2.1 Reusing the model
Create a new DNN that reuses all the pretrained hidden layers of the architecture in part 1.2.1, freezes them, and replaces the softmax output layer with a new one.

We restore the best model's graph from part1.

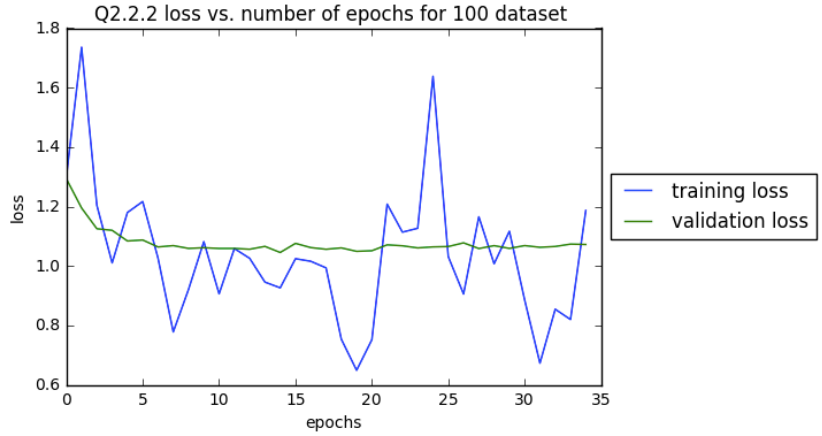
2.2.2 Training
Train this new DNN on digits 5 to 9. Plot the training and validation classification error and loss vs. the number of epochs. What precision can you achieve over the test set? How many iterations does it take to converge?



The test set precision should be 0.6194199.
Since the number of epochs is 22, the number of iterations should be (26962//20)*22 = 29656.
Therefore, it takes 29656 iterations to converge.

Train again the DNN using 100 images per digit and compare the performance with the case that the DNN is trained over full training set.

```
<matplotlib.legend.Legend at 0x131734f60>
```

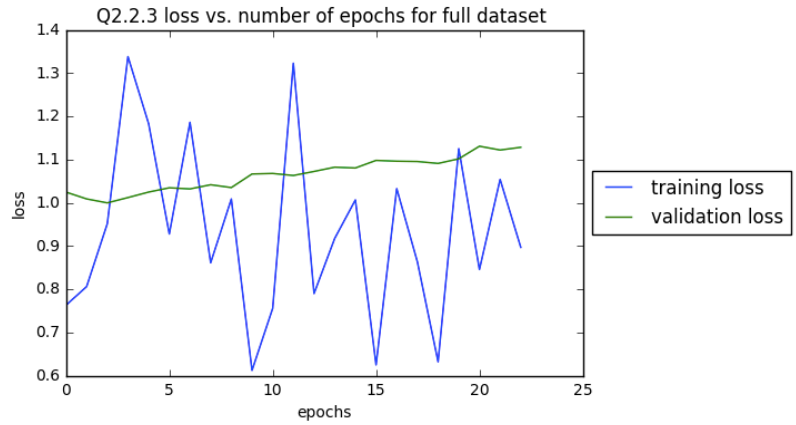


Compare the precision over the test set with that of part 2.2.2, which is 0.6194199, the precision value would increase as we removed the fifth hidden layer. The test set precision should be 0.60645956. Since the number of epochs is 34, the number of iterations should be (100//20)*34 = 170. Thus, our 100 training dataset model takes 170 iterations to converge. Comparing to the performance with the case that the DNN is trained over full training set, the test accuracy is lower and the number of iterations is much smaller. The reason is obvious that the training set is too small.

2.2.3 Removing the last hidden layer

Remove the fifth hidden layer, and train this new DNN. Plot the training and validation classification error and loss vs. the number of epochs. What precision can you achieve over the test set?

```
<matplotlib.legend.Legend at 0x14f879358>
```



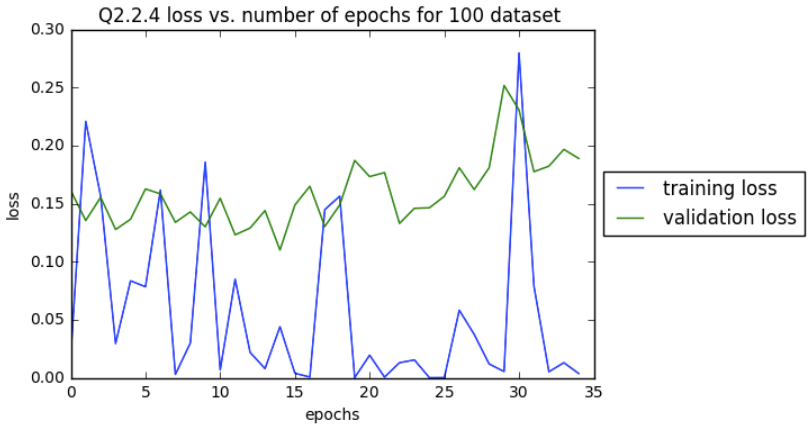
In this case, the precision over test set would be 0.6262086.

Compare the precision over the test set with that of part 2.2.2, which is 0.6194199, the precision value would increase as we removed the fifth hidden layer.

2.2.4 Unfreezing hidden layers 3 and 4

In this part, unfreeze weights of layers 3, 4, and the output and train them with the training dataset. What precision can you achieve over the test set?

```
<matplotlib.legend.Legend at 0x13164f2b0>
```



In this case, the precision over test set would be 0.89240897.

Compare the precision over test set with that of part 2.2.2 and 2.2.3, the test precision would apparently increase after we removing the fifth hidden layer and unfreezing the weights of layers 3, 4 and output. Both training loss and validation loss are small, and the training loss is below the validation loss. It means that the model is good fitted.