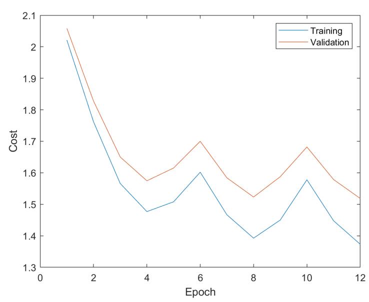
Deep Learning in Data Science (DD2424)
Report to Assignment 2 (Optional part)

Cong Gao

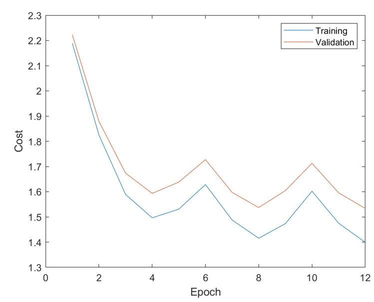
April 6, 2020

1 Optimize the performance

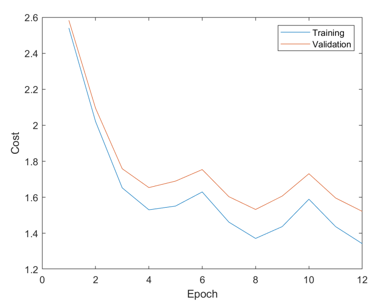
I tried the three following improving methods:
1. explore more hidden nodes. (The amount of regularization would have to increase with more hidden nodes)
I tried 80, 100, and 200 hidden nodes (all trained with three cycles). The results of classification accuracy and graphs of cost are as follows.



(a) 80 hidden nodes



(b) 100 hidden nodes



(c) 200 hidden nodes

Hidden nodes	80	100	200
λ	$5e-3$	$5.5e-3$	$6e-3$
Training accuracy	59.32%	58.99%	63.23%
Validation accuracy	53.80%	53.78%	55.288%
Test accuracy	53.24%	53.73%	54.28%

Table 1: Validation Accuracy as lambda varies

From above, we can see that with more hidden nodes the network gets better performance. After applying this improving method, the classification accuracy was enhanced by about 2%.
2. do a more exhaustive random search to find good values for the amount of regularization, the length of the cycles and number of cycles.
The found values: $hidden\_nodes = 200$, $\lambda = 5.5e-3$, $batch\_size = 200$, $eta\_min = 1e-5$, $eta\_max = 1e-1$, $n\_s = 4 * \frac{n\_samples}{batch\_size} = 900$, $n\_cycles = 5$.

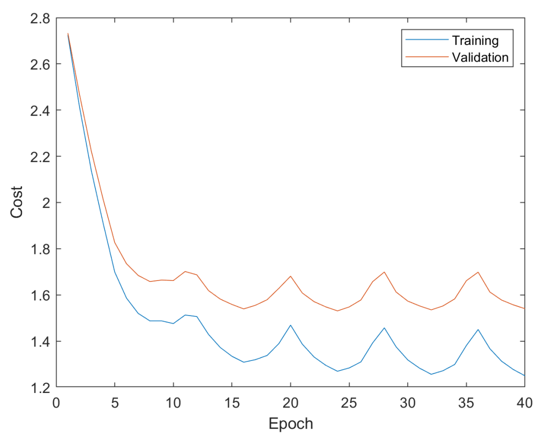


Figure 2: Graph of cost on training and validation set

	Training data	Validation data	Test data
Accuracy	68.64%	56.60%	55.69%

Table 2: Classification Accuracy

After more exhaustive search for values of parameters, the classification accuracy was enhanced about 1.5%.
3. apply dropout to the training.
According to the results above, we can see that the training accuracy is 68.64% and test accuracy is 55.69%, so the network needs more regularization. And also the network has a high number of hidden nodes, so I applied dropout to

the training process. The code of dropout is shown as follows. I set $p = 0.9$. And when at test time there is no need to compensate. I didn't change any other parameter.

```
s1 = W1*X + b1*ones(1, size(X,2));
u1 = (rand(size(s1))<p)/p;
s1 = u1.*s1;
h1 = s1;
h1(find(s1<0)) = 0;
s2 = W2*h1 + b2*ones(1, size(X,2));
u2 = (rand(size(s2))<p)/p;
s2 = u2.*s2;
```

Figure 3: Dropout code

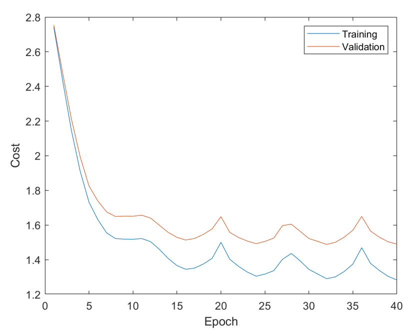


Figure 4: Cost plot after applying dropout

	Training data	Validation data	Test data
Accuracy	64.19%	55.96%	55.29%

Table 3: Classification Accuracy after applying dropout

.

After applying dropout, test accuracy is now 55.29% which is not different from previous result. Therefore in this case, dropout doesn't help improve test accuracy, but help alleviate overfitting.

4. build an ensemble of classifiers by saving the network parameter values at the end of each cycle and use this ensemble to classify.

I saved the network parameter values at the end of each cycle and used them to classify images in test set. The classification at end of each cycle is shown as follows.

	Second cycle	Third cycle	Fourth cycle	Fifth cycle
Accuracy	54.79%	55.21%	55.59	55.89%

Table 4: Classification Accuracy after applying dropout

.

The ensemble classification accuracy on test set is 55.37%, therefore in this case this method doesn't help improve the performance.

2 Set better eta_min and eta_max

In this case, I set $\lambda = 5.5e{-3}$, $hidden\_nodes = 200$, $batch\_size = 200$, $n\_epoch = 8$ At first I set $eta\_min = 1e-5$ and $eat\_ = 0.15$, $n\_s = max\_iter = n\_epoch * \frac{n\_samples}{batch\_size} = 1800$, that is, during the training process the learning rate will just increase linearly from the minimum value to the maximum value. I plot the accuracy versus learning rate which is shown as follows.

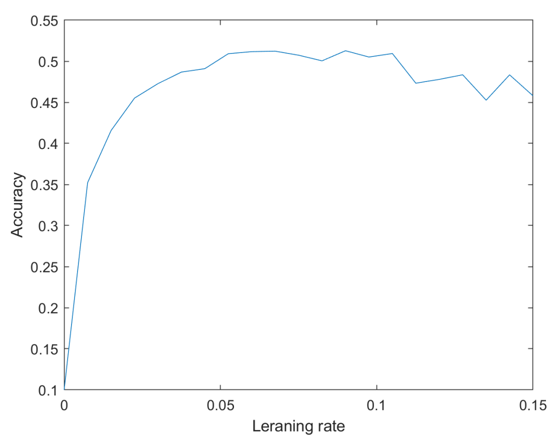


Figure 5: Classification accuracy as a function of increasing learning rate for 8 epochs (Learning rate test)

As we can see from the figure above, the network begins toc converge right away, so it is reasonable to set $eta\_min = 1e{-5}$. Furthermore, the classification accuracy riase gets rough and eventually drop above a learning rate of 6e-2, so I set $eta\_max = 0.05$. Other parameters are $\lambda = 6.5e-5$, $batch\_size = 200$, $n\_s = 4 * \frac{n\_samples}{batch\_size} = 900$, $hidden\_nodes = 200$, $n\_cycles = 5$, so the training cost plot and classification for the final network I trained with new found settings are as follows.

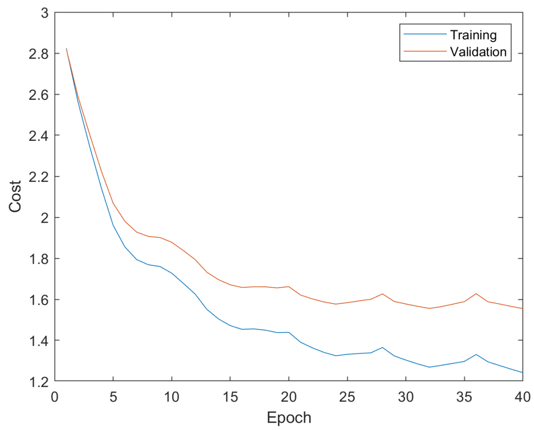


Figure 6: Classification accuracy as a function of increasing learning rate for 8 epochs (Learning rate test)

	Training data	Validation data	Test data
Accuracy	69.34%	56.02%	55.50%

Table 5: Classification Accuracy after applying dropout

.

Because *eta_max* is relatively small, it is likely that we can't see how many cycles in the training process from the cost/loss plot. That is to say, if *eta_max* is small enough (but reasonable), the cost/loss curve will keep decreasing rather than fluctuating. And with small *eta_max*, the training process will need more cycles (not considering cycle length there). The test accuracy of the final network is 55.50% after finding *eta_min* and *eta_max* for the network with a different number of hidden nodes and regularization.

3 References

[Smith, 2015] Smith, L. N. (2015). Cyclical learning rates for training neural networks. arXiv:1506.01186 [cs.CV].