HW4 Experiments

Team Members: **Gaurav Gupta (011445863)**
Ujjawal Garg (011917334)

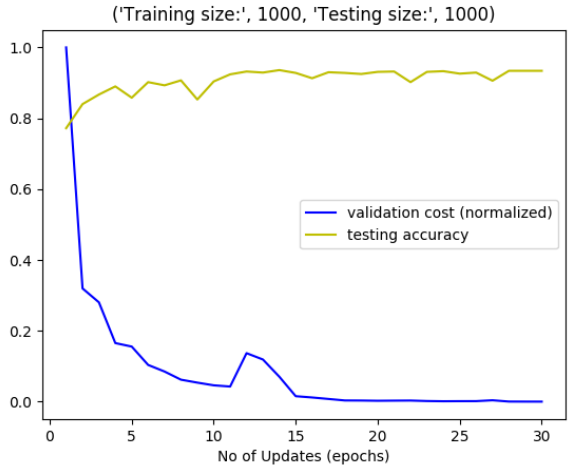
Note: Each experiment conducted has the corresponding result along with the its readme.txt. Also, the graph could be generated with those results using the plotter.py python file. Plotter.py has the parameter where it needs to specified the name of the results file for which the graph should be generated. Also, the graph displayed below uses the same plotter.py with the results stored in the corresponding trials.

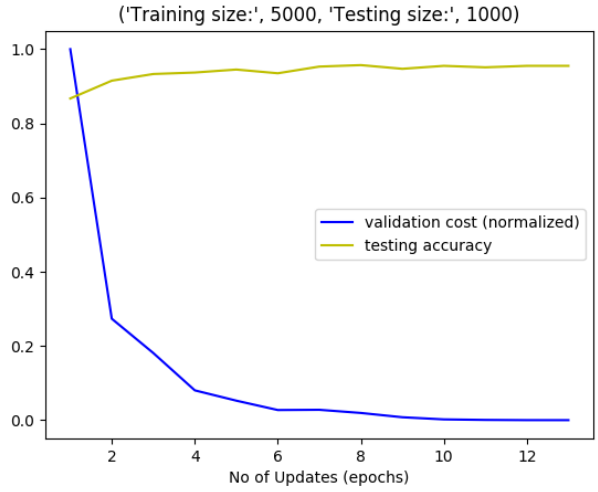
Experiment 1:

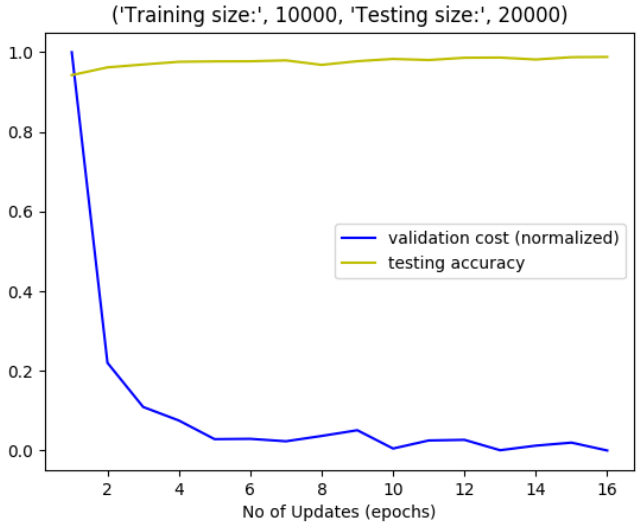
Hypothesis 1: Testing accuracy increases as the Validation cost decreases.

Description: For this hypothesis, we ran trials on different sizes of training and testing data sets. LeNet5 configuration without regularisation was used for training. Training was done for letter 'C'. Further details regarding this experiment can be found in the ReadMe.txt file in the Experiment 1 folder. The results of each experiment are stored in a txt file which can be fed to plotter.py to generate the graph.

Results:







Observations:

1. In each trial, we were able reach 100% training accuracy, and 0 final validation cost.
2. There are multiple local minimums for validation cost.
3. Max accuracy is observed at the end of training, when final validation cost is 0.
4. An increase in validation cost after an epoch does not necessarily mean decrease in accuracy. Rather, we see this as a general trend that accuracy and validation cost are inversely proportional.

Based on above observations, we can conclude that out hypothesis is correct.

Experiment 2

Hypothesis 2: Without any regularisation, model would converge faster, but we would get less accuracy on unseen testing data, as our model be overfitting. Using L1 regularisation should reduce overfitting and make the convergence slower. Using L2 should further reduce the overfitting and make the convergence even slower so that accuracy of unseen testing data has more chance to increase.

Description: For this hypothesis, we ran trials on different values of regularisation alpha. LeNet5 configuration was used for training. Training was done for letter 'C'. Further details regarding this experiment can be found in the ReadMe.txt file in the Experiment 2 folder. The results of each experiment are stored in a text file which can be fed to plotter.py to generate the graph.

Results:

Trial 1: Training Size: 10000, Testing Size: 20000 images

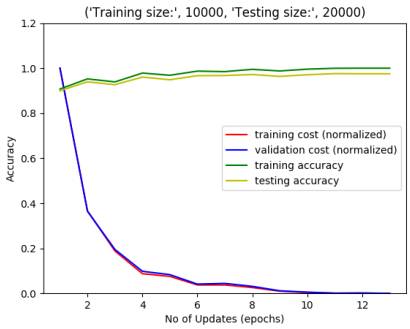


Fig 1. L1 alpha=0.01

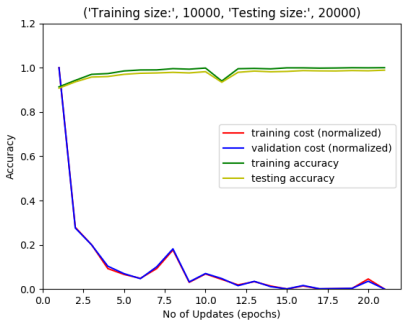


Fig 2. L1 alpha=0.5

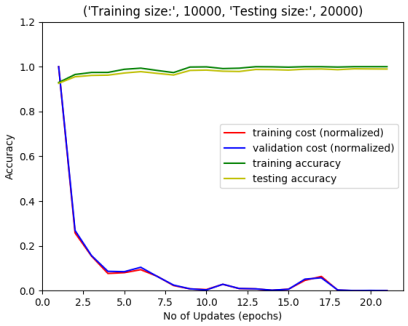


Fig 3. L1 alpha=1.0

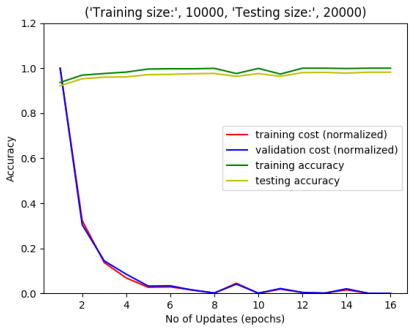


Fig 4. L2 alpha=1.0

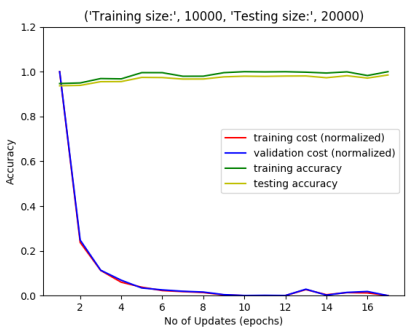


Fig 5. L2 alpha=0.5

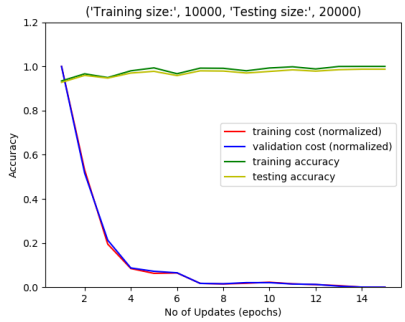


Fig 6. L2 alpha=0.01

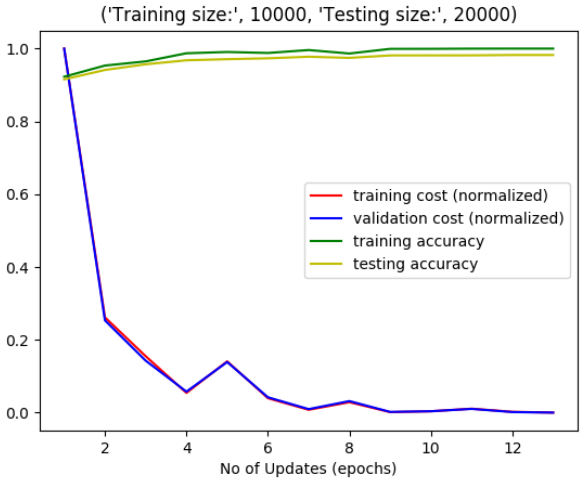


Fig 7. No Regularisation

Trial 2: Training Size: 1000, Testing Size: 5000 images

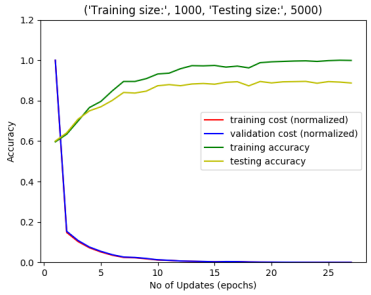


Fig 8. L2 alpha=0.5

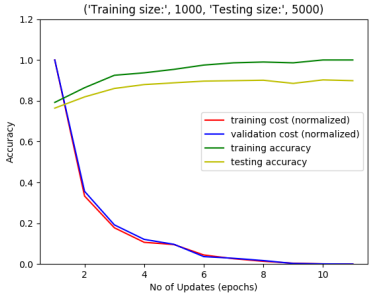


Fig 9. No Regularisation

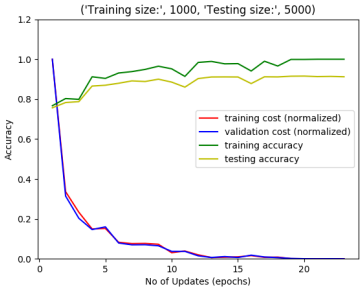


Fig 10. L1 alpha=0.5

Trial 3: Training Size: 5000, Testing Size: 1000 images

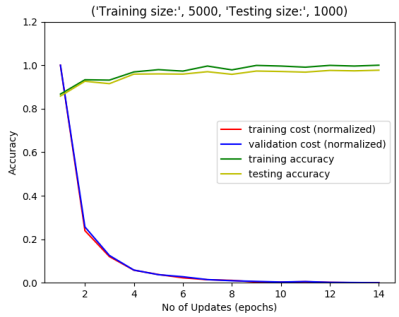


Fig 11. L1 alpha=0.5

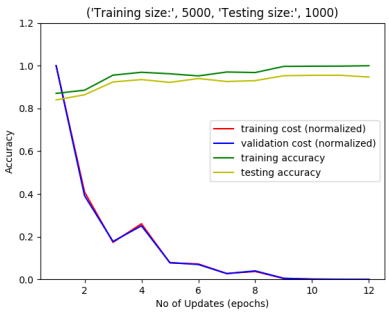


Fig 12. L2 alpha=0.5

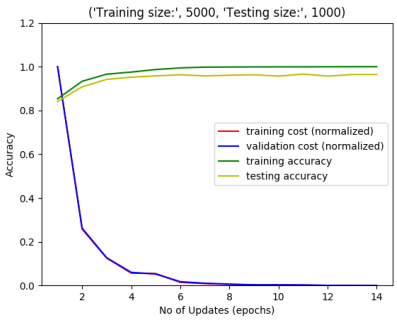


Fig 13. No regularisation

Observations:

1. In each of the trials, adding regularisation resulted in bumpy accuracy curves. This is expected as regularisations will add some penalty loss as we over fit the data, and accuracy will decrease for training set.
2. In each trial, training without regularisation resulted in lesser number of epochs.
3. In the trial with big training data set (10000 images), effect of regularisation was lesser. This is also expected as using large training data already reduces the chances of overfitting.

Based on these observations, we conclude that our hypothesis is correct.

Experiment 3:

Description: We conducted 6 Trials using the network example provided in HW description as our starting point. Other two Trials were conducted on LeNet5 network as per the lecture slides. Results of these trials are shown in Table below:

The same testing and training data set is provided for all the network configurations.

Trial Number	**Network Architecture File Name**	**Network Description**	**Epoch's**	**Testing Accuracy**
1	Le_net_5.txt	Le_Net_5 configuration according to slides	30	97.7%
2	Le_net_5.txt	Le_Net_5 configuration according to slides "**without max pooling**".	8	94.09%
3	Network_desc.txt	Example provided in the description of the homework 4, Also taken this as the base normal network description.	50	97.7%
4	Network_desc.txt	Network configuration used is same as provided in the homework description but the **max pooling layer skipped.**	11	92.4%
5	Network_desc_double.txt	The number of features used are **doubled** as compared to normal network configuration used.	22	96.7%
6	Network_desc_half.txt	The number of features used are **halved** as compared to normal network configuration used.	46	93.8%
7	Network_desc_additionalLayer.txt	The number of features in additional layer and rest of the layer are in the same ratio as that of normal configuration file.	7	90.3%
8	Network_desc_additionalDoubleLayer.txt	The number of features in additional layer and rest of the layer are **doubled** as compared to the normal network configuration used	29	98.1%

Observation:

1. From the experiments conducted as shown in the table above, the best accuracy achieved is **98.1%** in **Trial 8**. The accuracy is achieved by increasing the number of layers in our base configuration by one and doubling the number of features in each layer. Also, the number of epochs to reach the minimum validation cost is low when accuracy and epoch are both taken into consideration. The network configuration for Network_desc_additionalDoubleLayer.txt file is as follows:

```
5  8
6  16
6  32
8  64
128
```

2. Network configuration without using the max pool decreases the accuracy. It could be observed from the Trial 2 and Trial 4. In the Trial-1 and Trial-3, same network configuration with same training and test dataset is used but still there is a drop-in accuracy on an average by 4%.
3. Time per epoch in case of additional layer is increased significantly but the total number of epochs to converge is decreased. This is expected because in each epoch the amount of variables that must be computed are also increased.

Experiment 4:

The best accuracy achieved with SVM was 59% with 700 training samples and about 30000 updates. Whereas, with CNN models, even the lowest accuracy after first epoch was 75%. And if we let CNN train till 0.1 epsilon, we were able to achieve more than 95% accuracy in almost every trial.
This is expected, since using cross-validation results in better generalization of our model. Using L2 regularization further enhances the results of our model.

Also, each epoch of CNN will take more time then each iteration of SVM. Since, in each epoch we are considering the cross validation of the mini-batch certain number of times. Further it is required to compute regularization on the results of cross validation. Also, Max pooling adds a certain unit of time. However, the number of updates required in each iteration depends on the number of support vectors of SVM which would arbitrary, but still would be computed faster for one single iteration.