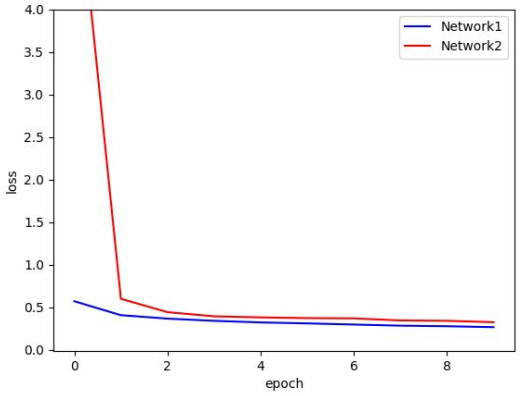
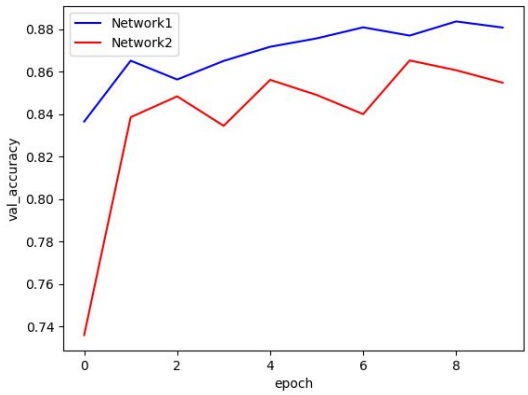
Deep Learning - Assignment 2

Veldurthy Sudhanyu – 15CS10050

Task (a)

- Parameters
  - Uniform initialization of weights in range [-0.1, 0.1]. This is to keep it different from other initializations in task (b). Except expt.1 in task (b), this initialization has been used.
  - Learning rate = 0.001. Faster convergence. Gets stuck in local minima for 0.01 or higher.
  - Epochs = 10. Validation accuracy saturates in approximately 5-6 epochs but used 10 for better plot. Training loss also reaches close to convergence in 10 epochs. This value has been fixed for all experiments in this assignment.
  - Optimizer = Adam. This has been used in all experiments unless mentioned otherwise.
- Network1 vs Network2
  - In loss vs epoch plot, we see that network 1 has a smaller loss at the beginning when compared to Network 2 but eventually both the losses converge to values which are very close.
  - In validation accuracy vs epoch plot, we observe that network 1 outperforms network 2. The higher depth of network 1 allows the features to become more abstract leading to a higher accuracy. Test accuracy reaffirms that network 1 performs better than network 2.



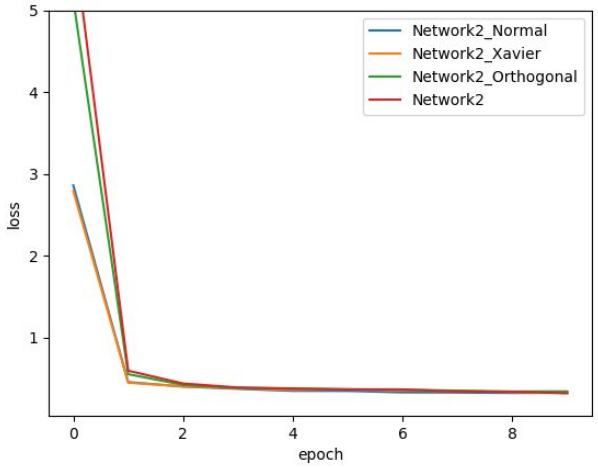


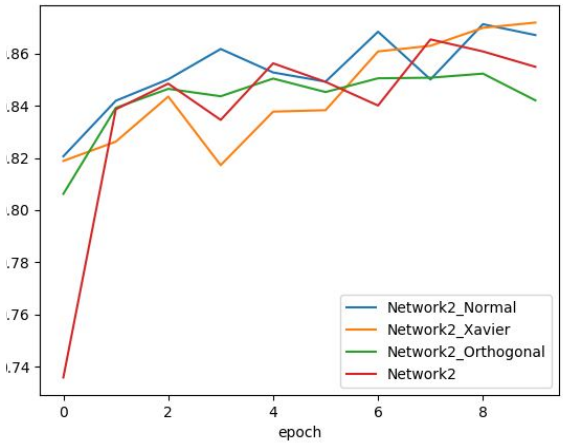
  - Test Accuracy of Network 1: 0.8729
  - Test Accuracy of Network 2: 0.8426

Task (b)

Experiment 1 - Different initializations of weights

- Parameters
    - For Normal initialization, a sigma value of 0.05 has been used as it gave faster convergence.
    - For Xavier and Orthogonal initializations, default values have been used as they were performing good.
- Comparison with Network2
    - From the plots, we can observe that all three initializations are faster at convergence than uniform initialization (not very significant). Even though all three initializations lead to very close loss and accuracy values, Xavier initialization works the best for this particular data and network. This is also evident from the test accuracy values.

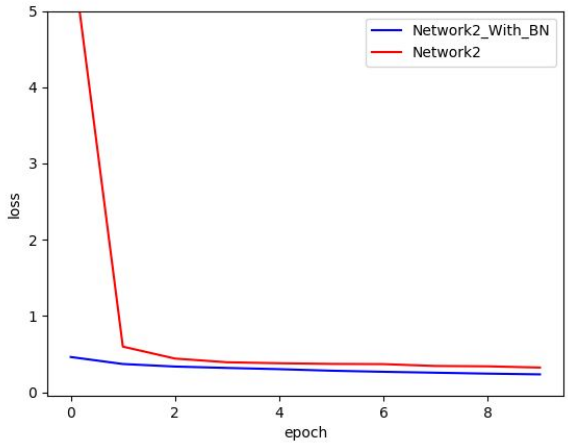


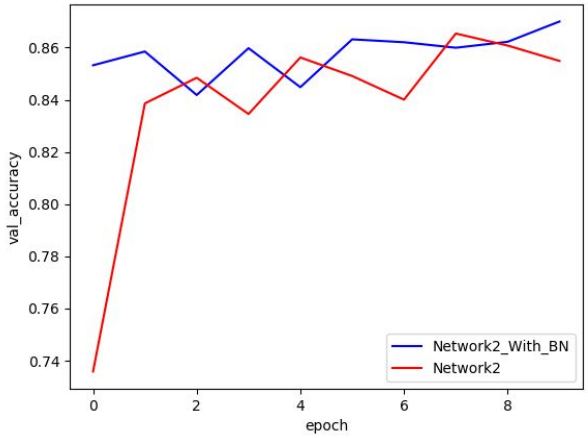


    - Test Accuracy of Network 2 - Normal Initialization: 0.8524
      Test Accuracy of Network 2 - Xavier Initialization: 0.8616
      Test Accuracy of Network 2 - Orthogonal Initialization: 0.8302
      Test Accuracy of Network 2: 0.8426

Experiment 2 - Using Batch Normalization

- Architecture
    - Three batch normalization layers have been added to the network, one after each of the fully connected layers.
- Comparison with Network2
    - Batch normalization takes care of exploding gradients in this case.
    - From the plot, we can see that the loss in the first epoch is very high without batch normalization layers. This is because batch normalization layers limit the outputs of the layers to a certain range thus also limiting the gradients. This also leads to faster convergence.

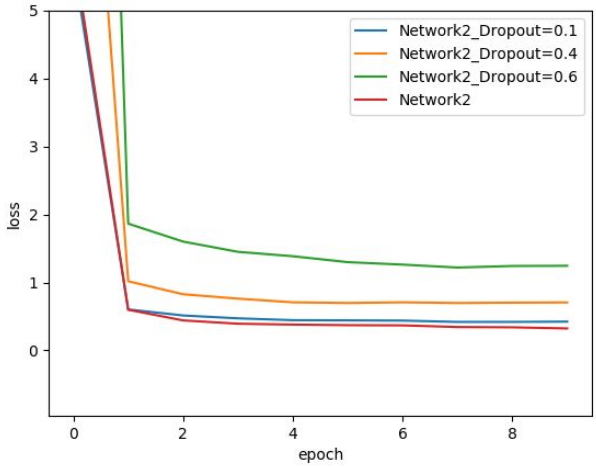


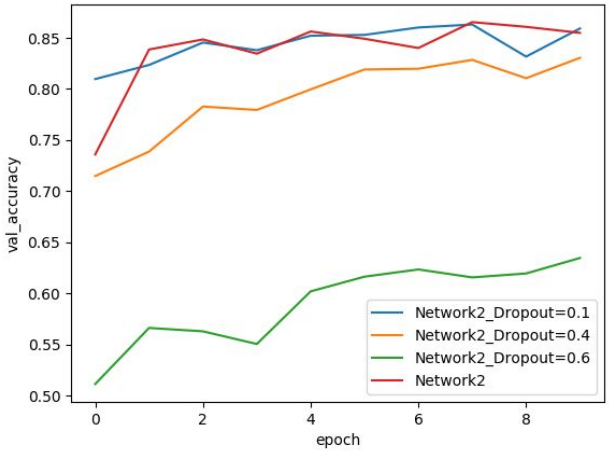


    - From the test accuracy values, we see that batch normalization also increases the accuracy marginally.
    - Test Accuracy of Network 2 - With Batch Normalization: 0.8623
      Test Accuracy of Network 2: 0.8426

Experiment 3 - Using Dropout (0.1, 0.4, 0.6)

- Architecture
    - For each of the values of dropout (0.1, 0.4, 0.6), one layer of dropout with specified drop probability is added after each fully connected layer.
- Comparison with Network2
    - Dropout is a regularization technique used to tackle the problem of overfitting. But in our case, since there is no overfitting taking place, dropout does not lead to a lesser loss or better accuracy.
    - Dropout of 0.6 probabilistically makes the network lose many nodes in each epoch leading to underfitting and hence a lower accuracy value (all train, validation and test accuracies).
    - Dropout of 0.4 also leads to underfitting leading to slightly lesser accuracy values and higher loss.

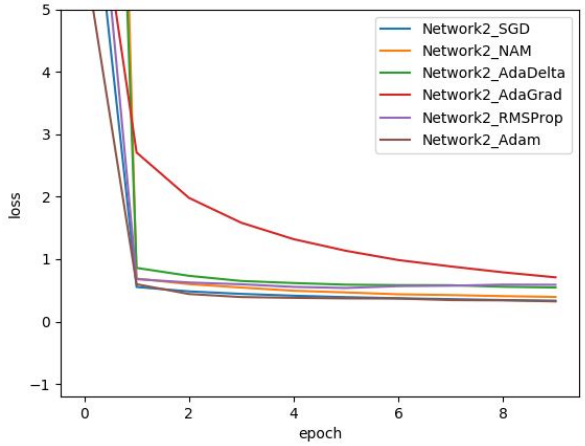


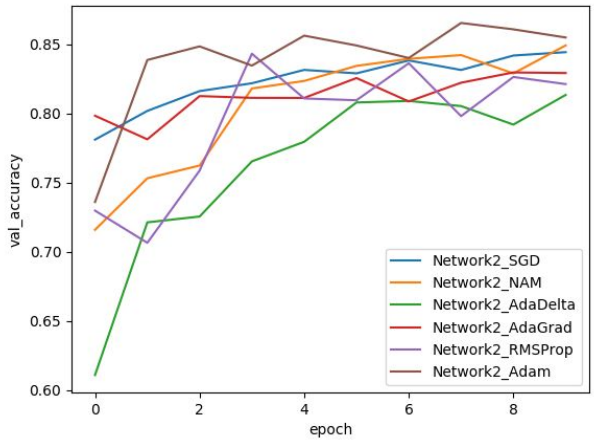


- Test Accuracy of Network 2 - Dropout = 0.1: 0.8527
  Test Accuracy of Network 2 - Dropout = 0.4: 0.8228
  Test Accuracy of Network 2 - Dropout = 0.6: 0.6268
  Test Accuracy of Network 2: 0.8426

Experiment 4 - Using Different Optimizers

- Parameters
    - Manually defined batch SGD and Nesterov's Accelerated Gradient Descent have been used.
    - For AdaDelta, AdaGrad, RMSProp and Adam, optimizers provided by Mxnet Gluon have been used with default parameters.
- Comparison with Network2
    - From the plot of training loss vs epochs, we see that AdaGrad is the slowest to converge and all other optimizers have similar convergence rate.
    - From accuracy plot, we see that Adam performs the best followed closely by NAM and SGD. AdaGrad, RMSProp and AdaDelta perform poorly for this network architecture and data.

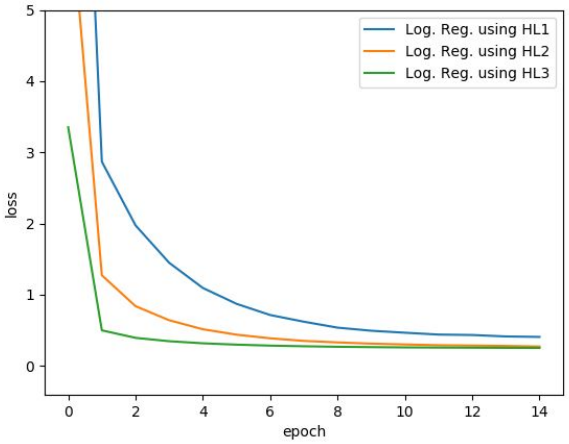


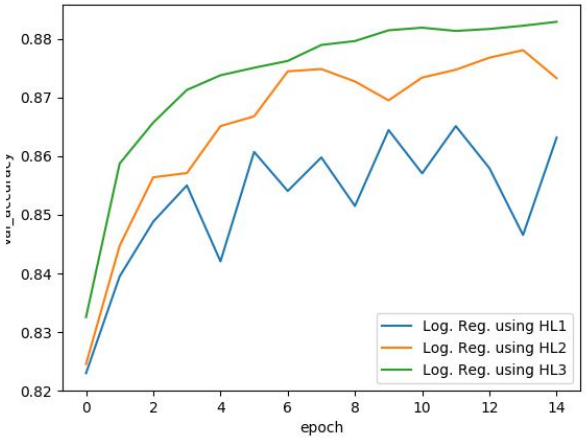


    - Test Accuracy of Network 2 - SGD: 0.837
      Test Accuracy of Network 2 - Nesterov's Accelerated Momentum: 0.8395
      Test Accuracy of Network 2 - AdaDelta: 0.8026
      Test Accuracy of Network 2 - AdaGrad: 0.8194
      Test Accuracy of Network 2 - RMSProp: 0.8199
      Test Accuracy of Network 2 - Adam: 0.8426

Task (c)

- Architecture
    - Logistic Regression classifier has been implemented using Mxnet Gluon. There are three classifiers which take the outputs of different hidden layers of network 2 as the inputs.
    - Learning rate of 0.0003 has been used for a smoother loss curve as convergence is fast.
    - Xavier initialization has been used.
    - 15 epochs were used for classifier using hidden layer 1 to fully converge.
- Comparison among the three classifiers
    - From the training loss plot, we see that the classifier using hidden layer 3 (HL3) output as input is the fastest to converge followed by HL2 and HL1.
    - Also we observe that HL1 converges to a higher loss value indicating underfitting.
    - From accuracy plot, we see that HL3 performs the best followed by HL2 and HL1. The reason for this could be that the features got from the outputs of hidden layer 1 were not good enough for the less powerful logistic regression classifier whereas the output of hidden layer 2 and hidden layer 3 gives very good abstract features leading to a high accuracy.





    - Test Accuracy of Logistic Regression Classifier - Using Output of HL1: 0.8583
      Test Accuracy of Logistic Regression Classifier - Using Output of HL2: 0.8676
      Test Accuracy of Logistic Regression Classifier - Using Output of HL3: 0.8755