Name(s): Punit K. Jha
NetID(s): punit2
Team name on Kaggle leaderboard: PunitKJha

For each of the sections below, your reported test accuracy should approximately match the accuracy reported on Kaggle.

Briefly describe the hyperparameter tuning strategies you used in this assignment. Then record your optimal hyperparameters and test/val performance for the four different network types.

The different hyperparameter tuning strategies that I tried are listed and described below:

1. **Epochs**: To see the effect of number of epochs on the accuracy of the NNs I tried running the NN for the following values of epoch -- [100,150]
2. **Batch Size**: To see the effect of increasing the batch size in the stochastic gradient descent method on the accuracy of the NNs I tried the following list of values for the batch size parameter -- [50,200,400,1000]
3. **Hidden Layer Size**: To see the effect of the number of nodes on the accuracy of the NNs I tried the following list of values for the hidden size layer -- [20,40,80,120]
4. **Learning Rates-** To see the effect of learning rates on the accuracy of the NNs I tried the following list of values for learning rates -- [$1e^{-1}$,$1e^{-2}$,$1e^{-3}$]
5. **Regularization**: To increase or decrease the fitting of the training data set I tried the following list of values for the regularization parameter -- [0.95,0.4,0.05,0.005]
6. **Learning Decay rate-** I also went on to implement an learning decay algorithm in my codes such that the learning rate is decreased as the NNs approach the optimum parameters. Its shown below

```
learning_rate=learning_rate/(1+learning_rate_decay*epoch)
```

Two-layer Network Trained with SGD

Best hyperparameters (if you changed any of the other default hyperparameters like initialization method, etc. please note that as well):

Batch size:	200
Learning rate:	1e-3
Hidden layer size:	80
Regularization coefficient:	0.009

Record the results for your best hyperparameter setting below:

Validation accuracy:	`0.53222`
Test accuracy:	`0.5001`

Three-layer Network Trained with SGD

Best hyperparameters (if you changed any of the other default hyperparameters like initialization method, etc. please note that as well):

Batch size:	200
Learning rate:	1e-3
Hidden layer size:	120
Regularization coefficient:	0.0009

Record the results for your best hyperparameter setting below:

Validation accuracy:	`0.611`
Test accuracy:	`0.517`

Two-layer Network Trained with Adam

Best hyperparameters (if you changed any of the other default hyperparameters like initialization method, etc. please note that as well):

Batch size:	200
Learning rate:	1e-3
Hidden layer size:	100
Regularization coefficient:	0.00009
β_1	0.9
β_2	0.999

Record the results for your best hyperparameter setting below:

Validation accuracy:	`0.7696`
Test accuracy:	`0.5009`

Three-layer Network Trained with Adam

Best hyperparameters (if you changed any of the other default hyperparameters like initialization method, etc. please note that as well):

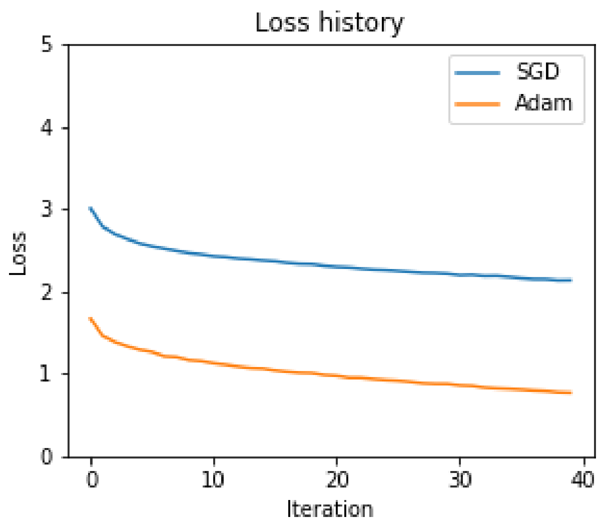
Batch size:	200
Learning rate:	1e-3
Hidden layer size:	100
Regularization coefficient:	0.00009
β_1	0.9
β_2	0.999

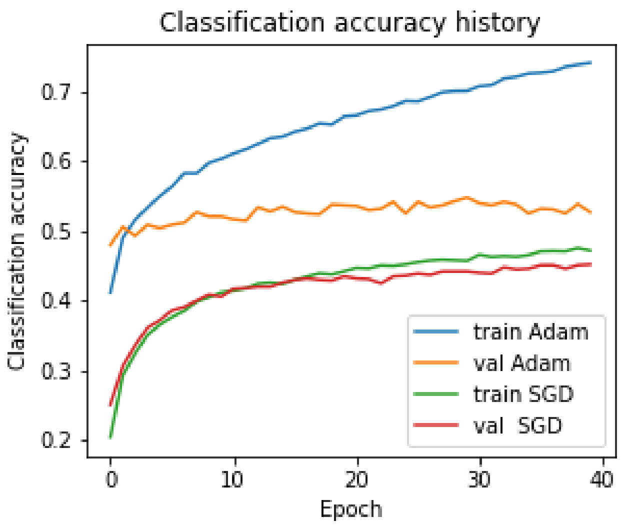
Record the results for your best hyperparameter setting below:

Validation accuracy:	0.7397
Test accuracy:	0.5158

Comparison of SGD and Adam

Attach two plots, one of the training loss for each epoch and one of the validation accuracy for each epoch. Both plots should have a line for SGD and Adam. Be sure to add a title, axis labels, and a legend.





For the Adam gradient updated 2 or 3-layer NN we observe that even for a very low value of regularization constant as the training error decreases with the increasing number of epochs the training accuracy keeps on increasing. However, the validation and testing accuracy increase up to a certain number of epochs and then start decreasing. This might be due to the fact that Adam gradient method causes faster convergence of results as compared to SDG and for lesser number of epochs we get the best Adam results given the parameters. However, if we don't stop the training after the optimum number of epochs, the model overlearns or overfits the data which increases training accuracy but decreases the validation and testing accuracy. So we do only 40 epochs for Adam gradient methods as compared to 100-150 for SGD.

Compare the performance of SGD and Adam on training times and convergence rates. Do you notice any difference? Note any other interesting behavior you observed as well.

- As is visible in the plots above, as compared to the SGD the Adam gradient update method converges faster, in only about 40 iterations on the given CIFAR-10 dataset while it required about 100-150 epochs for convergence using SGD dataset. This leads to enormous training time savings (almost 1/3 of the time required for SGD).
- We also note that the training error rate is lower in case of Adam and the rate of decrease of training error is faster.
- We also observe that Adam leads to higher training and prediction accuracy overall (if the number of epochs are kept constant).
- We also learn that the propensity of overfitting is higher for the Adam if the optimal number of epochs are not used. While the training accuracy keeps increasing for Adam as we increase the number of epochs the prediction accuracy starts to decrease.
- All in all, the Adam method is superior to the SGD method for gradient calucations.