Q1. Code in load\_data.py

Q2. Code in hw1.py

Q3.

**K = 1, N = 2**



As we can see above the test accuracy is plotted as a function of the steps. We see that after initially hovering around 50%, the accuracy increases to the 60-65% range and stays there for a few more iterations and then after around 27k steps the accuracy increases to above 80% where it remains until the final 50k steps, showing it has converged.

We can also see that the variance in the test accuracy, even after converging to the 80-90% range towards the end is quite high. This is probably due to the lower number of samples (K=1) in the one shot case.

**Note: Before mentioning observations, I would like to note that for all the below 3 configurations I had to use a lower learning rate of 0.0001 to achieve convergence. When I used 0.001 learning rate, the test accuracy would get stuck and not be able to converge. To compensate for the decrease in learning rate by a factor of 10, I increased the number of steps by 10 to 500k. However, as we will see below convergence is achieved earlier than that.**

**K = 1, N = 3**



As we can see above, after 170k steps with learning rate of 0.0001, we are able to on average achieve greater than 60% test accuracy. This is further improved when we train further and after 420k steps we see that we are comfortably able to achieve test accuracies of greater than 65% accuracy.

**K = 1, N = 4**



For this configuration, we see that we initially have test accuracy of 25% up to around 60k steps. Then we average around 40% until 250k steps after which we are consistently above 45% which is the test accuracy that we converge to.

**K = 5, N = 4**



Here we see that we get stuck in a local minima of 25% and are not able to get out of this. Given no improvement for 400k steps, I have terminated the training earlier than 500k in this case compared to the other two.

Q4.