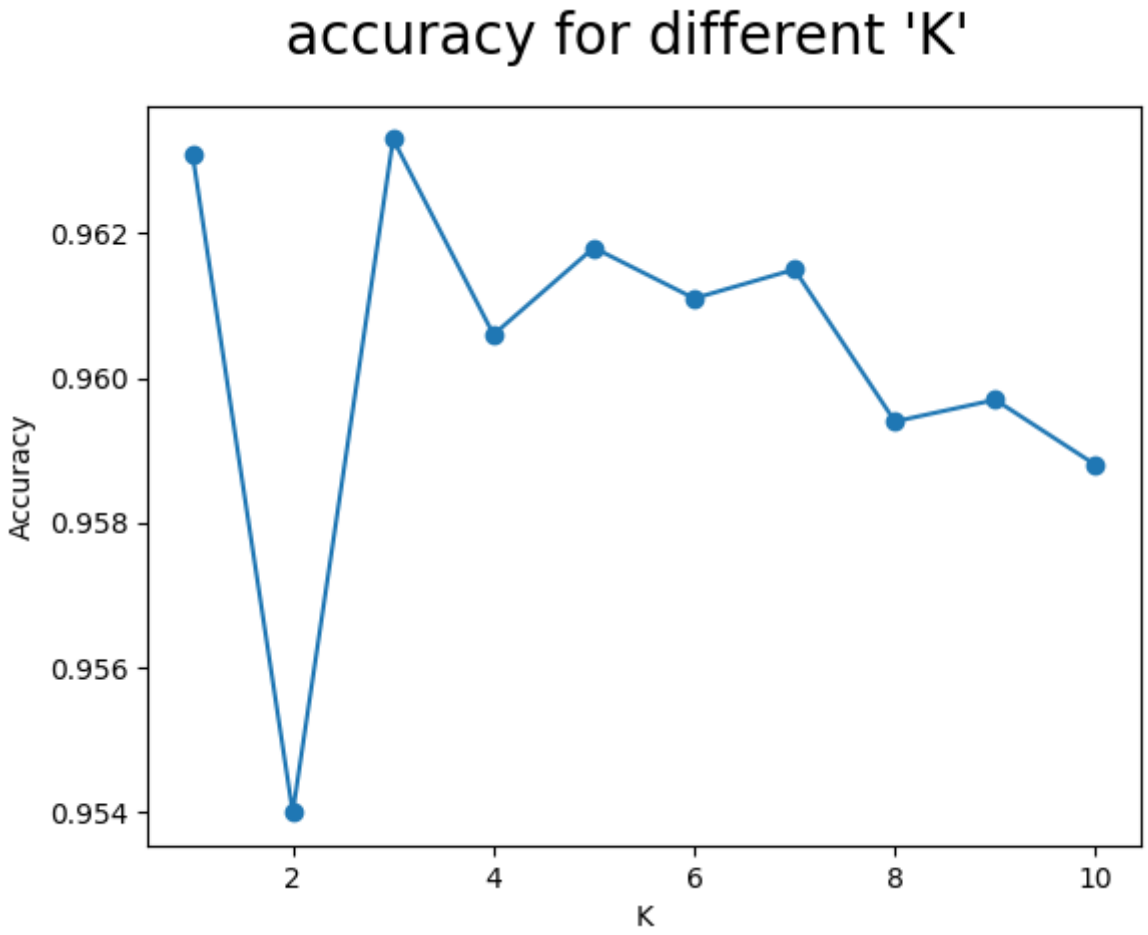
COMP5214 Assignment 1 Report

Note that except for KNN, during the trainning process of all MLP, CNN, and CAN, I randomenized the order and images for each batch. This means that each time the code is run, the model may be trained in a different way. Therefore, the testing results for each run may have slight differences. This slight difference shall not affect the comparison in terms of overall accuracy and efficiency between different models.

K Nearest Neighbors (KNN)

For the given `MNIST` Dataset, and the use of the `KNeighborsClassifier` from `sklearn.neighbors`, we obtained the following results:

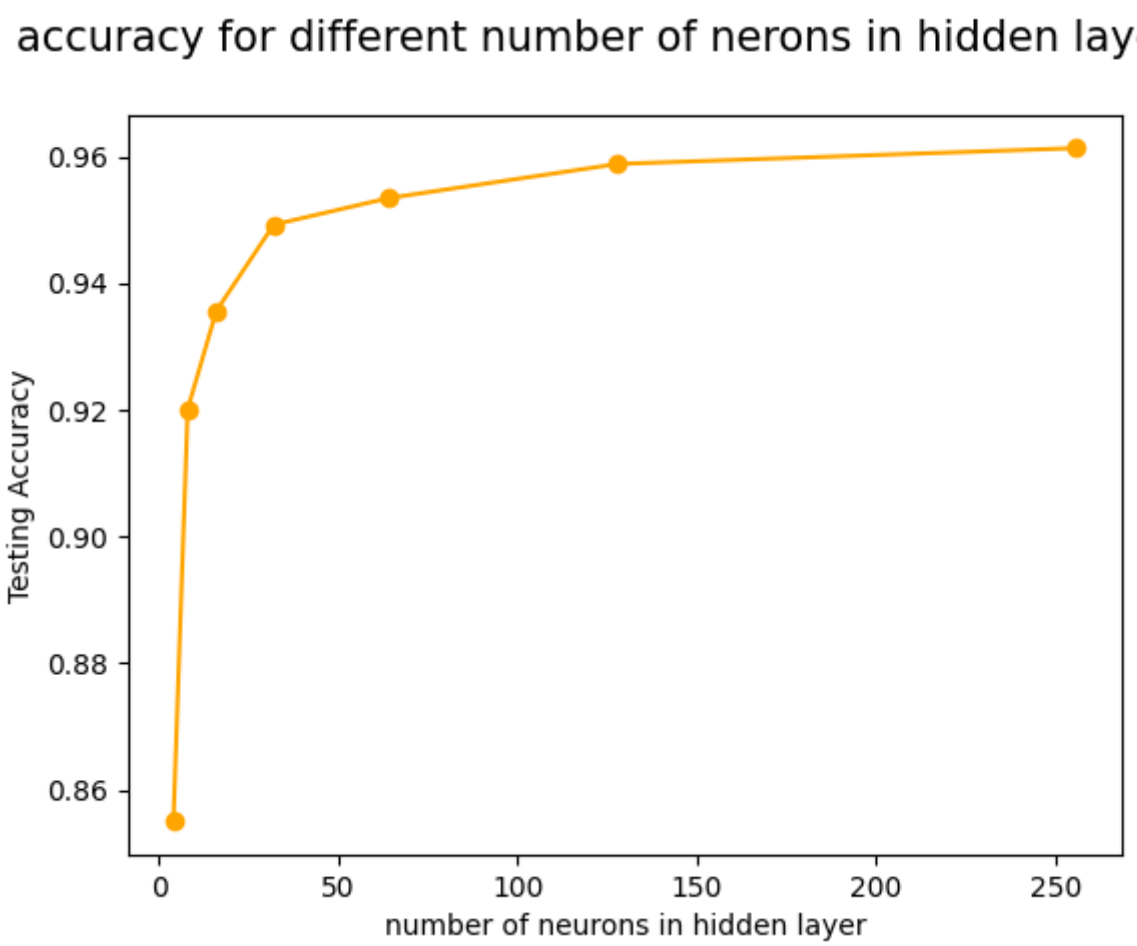
- **The accuray for K=1 is 0.9631.**
- The accuracy reaches the highest when K=3, with accuacy of 0.9633.
- A plot of accuracy versus K for K from 1 to 10 is shown below:



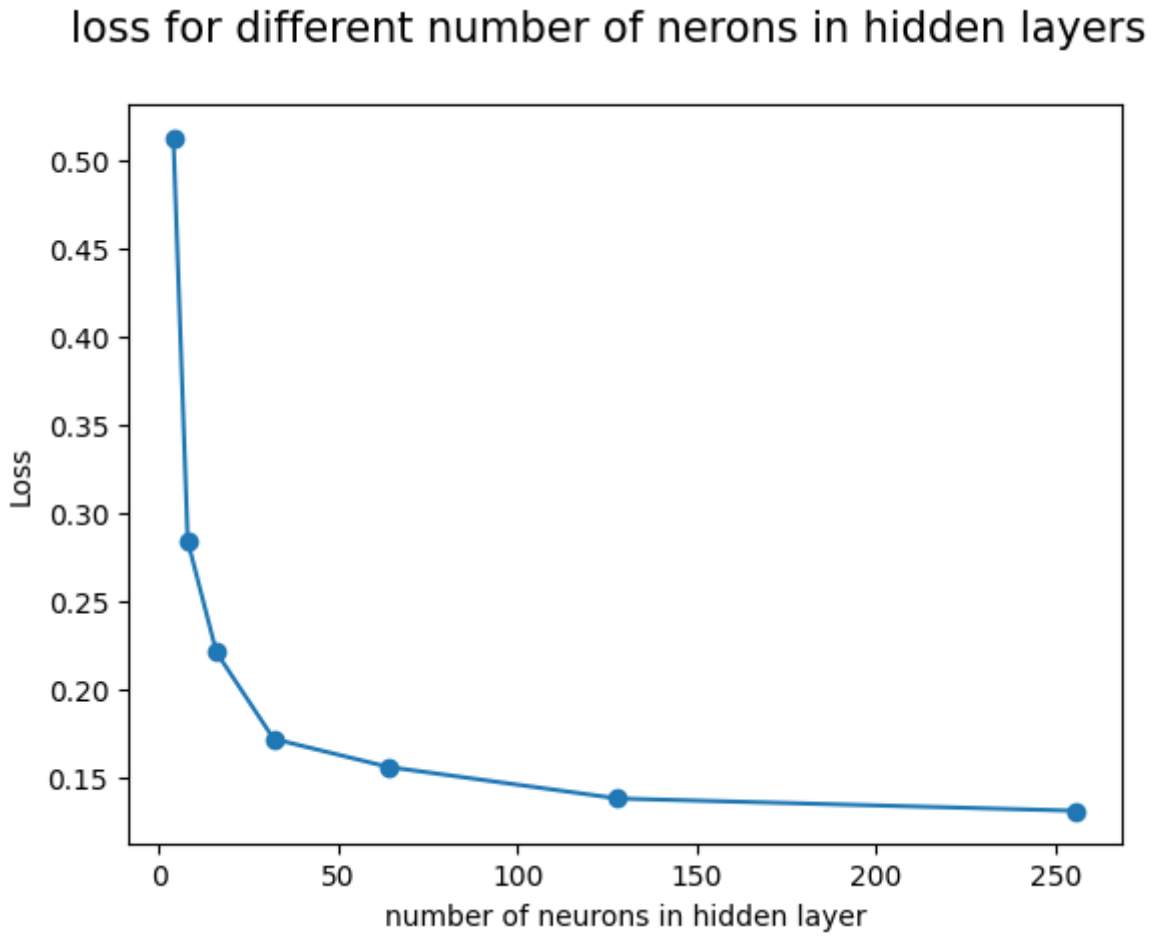
In General, the accuracy goes down after reaching the top at K=3.

Multilayer Perceptron (MLP)

By experimenting with different number of neurons in the hidden layer, we obtained the following plot of accuracy versus the number of neurons in the hidden layer:



We also obtained the plot illustrating the trend of loss change:



From the graph we can see that the accuracy becomes higher and higher as the number of neurons in the hidden layer doubles. However, the rate of accuracy improvement becomes slower as more and more neurons are added.

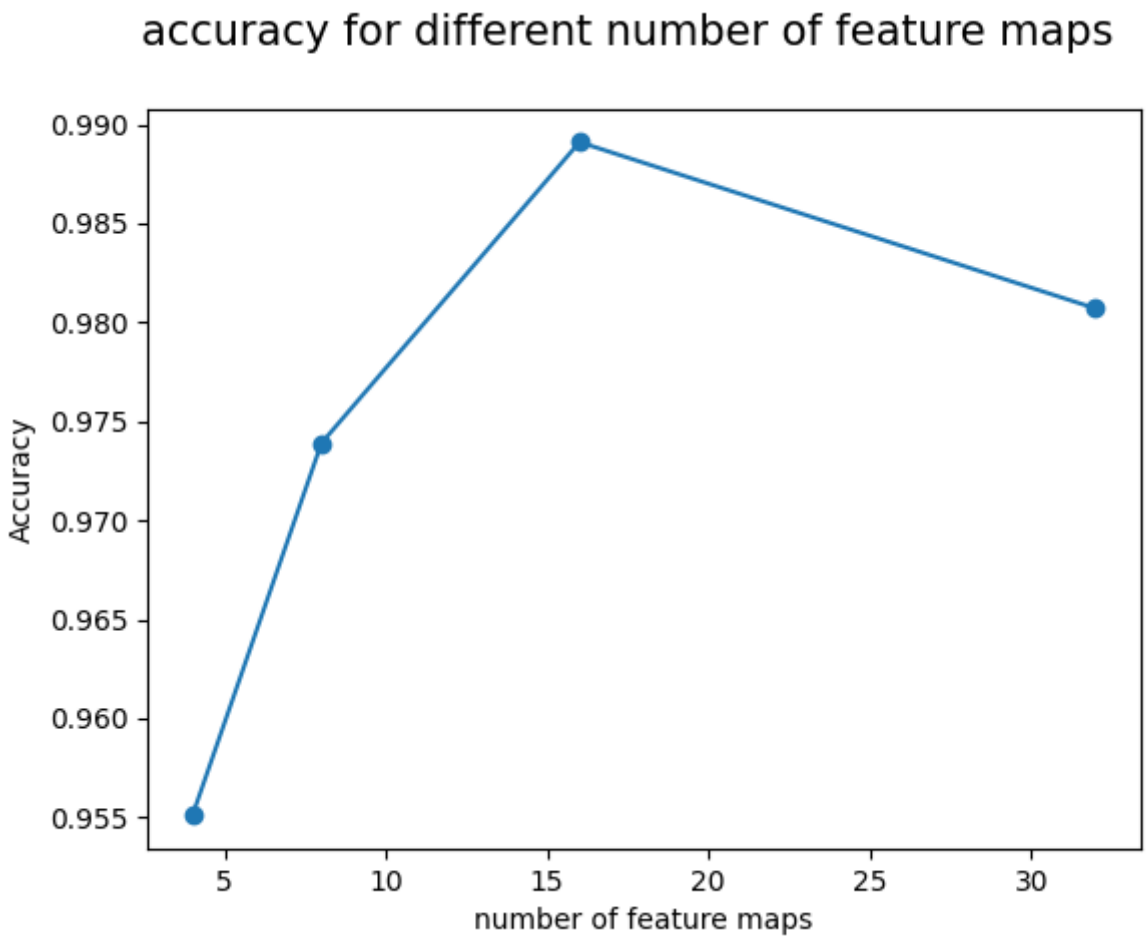
Convolutional Neural Networks (CNN)

The accuracy on the test data for this network is 0.9698, which is better than the accuracy of the MLP model when 256 neurons were added in the hidden layers. This suggests that adding convolutional layers before fully connected layers may have better performance in tasks like image classification.

Context Aggregation Networks (CAN)

For this model, without padding, the input will not be able to pass the 4th convolutional layer whose kernel size will be bigger than the local input.

The Accuracy of the CAN model as illustrated in `table 1` is 0.9807. Note that for this CAN model, it is observed that the speed of convergence is rather slow, so I increased the learning rate from 0.01(which was the learning rate used in MLP and CNN), to 0.1, to increase the speed of convergence. I experimented this model with 4, 8, 16, 32 feature maps, and obtained the following relationships between the number of feature maps and the accuracy:



It is possible that with 32 feature maps and a learning rate of 0.1, it may have over-fitted. Since we simpley uses the cross entropy loss, without adding any regularization, this might be the reason why the performance is better when the number of feature maps is 16.