COM S 474/574: Introduction to Machine Learning
Homework #4

1. Please put required code files and report into a compressed file "HW#_FirstName_LastName.zip"
2. Unlimited number of submissions are allowed on Canvas and the latest one will be graded.
3. No later submission is accepted.
4. Please read and follow submission instructions. No exception will be made to accommodate incorrectly submitted files/reports.
5. All students are required to typeset their reports using latex. Overleaf (`https://www.overleaf.com/learn/latex/Tutorials`) can be a good start.

1. (20 points) **Hierarchical clustering**

   Use the similarity matrix in Table 1 to perform (1) single (MIN) and (2) complete (MAX) link hierarchical clustering. Show each step with dendrogram and the corresponding similarity matrix update. The dendrogram should clearly show the order in which the points are merged. Suppose we choose to use 3 clusters, Show the cut in each final dendrogram.

   Table 1: Similarity matrix.

	p1	**p2**	**p3**	**p4**	**p5**
p1	1.00	0.10	0.41	0.55	0.35
p2	0.10	1.00	0.64	0.47	0.98
p3	0.41	0.64	1.00	0.44	0.85
p4	0.55	0.47	0.44	1.00	0.76
p5	0.35	0.98	0.85	0.76	1.00

2. (20 points) **K-Medians Clustering**

   The K-means algorithm can be summarized as below:

   ```
   (a) Select K points as the initial centroids.
   (b) repeat
   (c)     Form K clusters by assigning all points to the closest centroid.
   (d)     Recompute the centroid of each cluster.
   (e) until The centroids don't change.
   ```

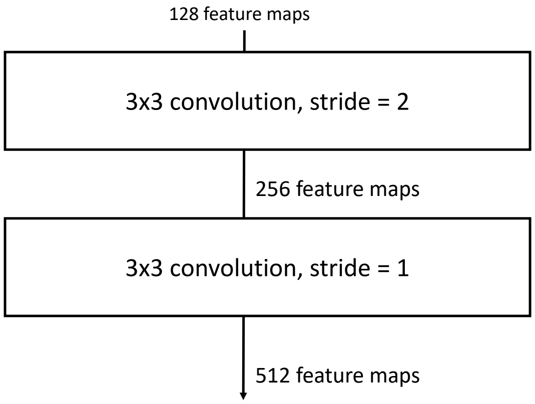
   K-medians clustering is a variation of k-means clustering where it calculates the median for each cluster to determine its center instead of using the mean. Also, K-medians makes use of the Manhattan distance for points assignment.

   (a) (8 points) Please show the algorithm of K-medians in the above format.

   (b) (6 points) Please explain how you will compute the median for each cluster.

   (c) (6 points) Does K-medians help to avoid the outlier problem? Justify your answer.

3. (15 points) Given the convolutional neural network block as below



Given the input feature maps $\boldsymbol{X} \in \mathbb{R}^{64\times64\times128}$, all convolutional layers perform zero-padding of 1 on each side of H and W dimensions.

(a) (5 points) What is the total number of parameters in the block (you can skip bias terms)?

(b) (5 points) What is the total number of multi-add operations in the block?

(c) (5 points) What is memory requirement change to store the input and output features of this block (Use percentage)?

4. (45 points) **Neural Network for Image Recognition:** In this coding assignment, you will need to complete the implementation of a neural network (Fully-Connected Network for 474 students and Convolutional Neural Network for 574 students) using PyTorch and apply the network to the image recognition task on Cifar-10 (10-classes classification). You will need to install the python packages "tqdm" and "pytorch". Please read the installation guides of PyTorch here (https://pytorch.org/get-started/locally/). You are expected to implement your solution based on the given codes. The only file you need to modify is the "solution.py" file. You can test your solution by running the "main.py" file.

(a) (25 points) Complete the class ***Net()***. In particular, define operations in function ***__init__()*** and use them in function ***forward()***. The input of ***forward()*** is an image.

For 474 Students: Please use ***ReLU*** function to activate the outputs of the first two fully-connected layers. The sequential layers are:

Inputs →
Reshape to vector → Fully-connected (512 out units) →
Fully-connected (512 out units) → Fully-connected (n_classes out units)

For 574 Students: The sub-sampling is implemented by using the max pooling. And the kernel size for all the convolutional layers are 5×5. Please use ***ReLU*** function to activate the outputs of convolutional layers and the first two fully-connected layers. The sequential layers of the network are:

Inputs →
Convolution (6 out channels) → Max Pooling →
Convolution (16 out channels) → Max Pooling →
Reshape to vector → Fully-connected (120 out units) →
Fully-connected (84 out units) → Fully-connected (n_classes out units)

For this part, you are only allowed to use the APIs in ***torch.nn***. Please refer to the PyTorch API documents below for the usage of those APIs before you use them:
https://pytorch.org/docs/stable/nn.html.

Run the model by "***python main.py***" and report the testing performance as well as a short analysis of the results.

(b) (10 points) Tune the hyperparameter of batch in "***solution.py***". Please try 8, 16, 32, and 64. Run the model by "***python main.py***" and report the testing performances as well as a short analysis of the results.

(c) (10 points) Tune the hyperparameter of learning rate (lr) in "***solution.py***". Please try 0.01, 0.001, 0.0001, and 0.00001. Run the model by "***python main.py***" and report the testing performances as well as a short analysis of the results.