COMP 6721 Applied Artificial Intelligence (Winter 2023)

Assignment #3: Deep Neural Networks

Due: 11:59PM, April 2nd, 2023

Note Your will be submitting two separate files from this assignment as follows:

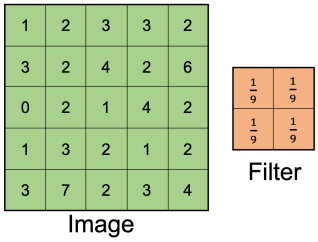
(a) **One(1) .pdf file:** containing answers to Question as well as reported results from coding you develop.

(b) **One(1) .zip folder:** containing all developed Python codes including a README.txt file on explaining how to run your code. **Do not** upload datasets in your .zip folder. We should be able to run your code on the original dataset downloaded from Kaggle.

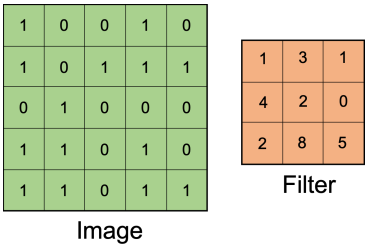
Theoretical Questions

Question 1 For each of the following 2D maps (images), apply convolution using the given filter with each of the following parameters: (1) stride = 1, (2) stride = 2, and (3) padding = 1 (with stride = 1). Refer to PyTorch (documentations1, documentations2) in special cases where the filter goes out of boundaries when applied to the image. In such cases, use "ceil_mode=False".

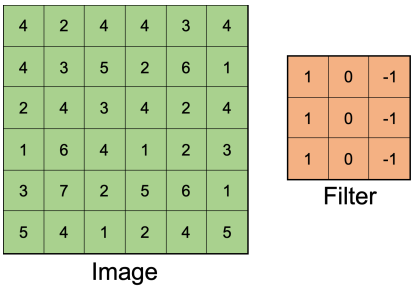
(a) Input image size: 5×5



(b) Input image size: 5×5



(c) Input image size: 6×6



Question 2 For each of the following scenarios, compute the dimensions of the output map produced by applying the given filter to the input image. Refer to PyTorch (documentations1, documentations2) on how to handle cases where you end up with fractions.

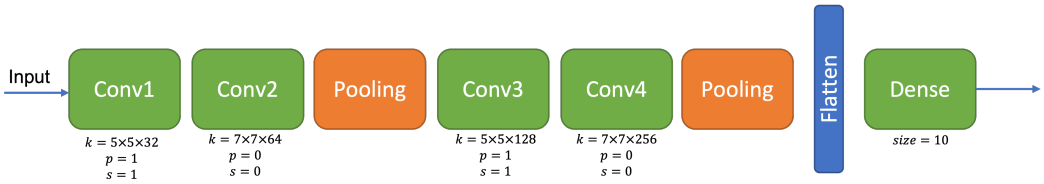
(a) Input image size: 224×224, filter size: 7×7, padding = 2, stride = 2.

(b) Input image size: 117×117, filter size: 11×11, padding = 1, stride = 1.

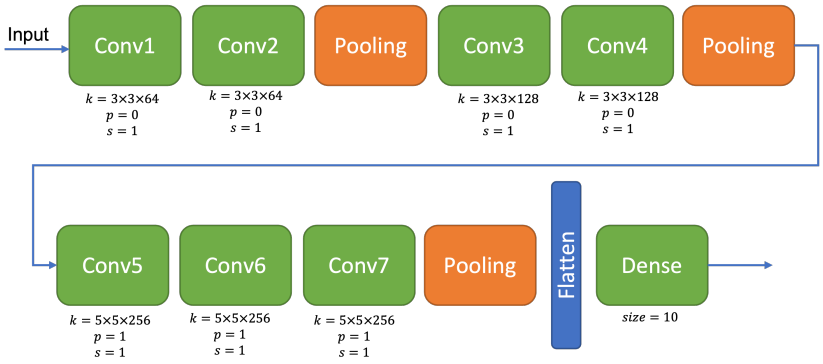
(c) Input image size: 235×235, filter size = 9×9, padding = 2, stride = 2.

Question 3 For each of the architectures given below, write down the output dimensions (in $H \times W \times N$ format) of each layer. Refer to the aforementioned PyTorch documentations when needed.

(a) Input image: $224 \times 224 \times 3$



(b) Input image: $224 \times 224 \times 3$



Question 4 Design a Convolutional Neural Network (CNN) for image classification with the following requirements:

- Input image size is $224 \times 224 \times 3$.
- The CNN encoder contains exactly three(3) convolutional layers.
- You are allowed a maximum of two MaxPool layers, each with a 2×2 kernel and a stride of 2.
- The output dimensions before flattening is $5 \times 5 \times 128$.
- For the convolutional layers, filters (kernels) have to be of odd dimensions with a maximum size of 7×7. The maximum padding allowed is 2 and the maximum stride allowed is 2.
- The classifier contains one Fully-Connected (FC) layer with 5 output prediction classes.

You are required to draw/sketch the whole CNN pipeline describing the details of each layer in the network (kernel size, padding, stride, input/output dimensions).

Implementation Questions

Question 1 Design and implement a CNN to be used in a task of Fingers Count classification. Given an image of a hand, your classifier should return the number represented (between 0 and 5). For example, the classification of the example image given below should be "3". The main aim is to maximize the classification accuracy. You should use and download the dataset from Kaggle. Your implementation should include the following:

- Data pre-processing: as per the provided dataset, the images are not already categorized into folders, and are just named according to the given class. Your data pre-processing implementation should take care of categorizing the data points and creating data loaders with the correct classes. This should be done internally in your implementation, and not manually beforehand (we should be able to reproduce your results by running the code on the original dataset from Kaggle). You should also split the data into a 70-10-20 train-val-test split. As per your design, you should choose appropriate pre-processing techniques to be applied to the images.
- Architecture design: you should design your own CNN architecture with choices of number of layers, activation functions, kernel specifications, etc. You could try several architectures, but only report your final design.
- Training and optimization: the student has the freedom of choosing hyperparameter values (learning rate, number of epochs, loss function, etc), with the aim of maximizing the accuracy. Here, hyperparameter optimization/search is encouraged to find the best set of hyperparameters.

You should report on the designed architecture and the chosen hyperparameters. You should also report results by plotting training and validation accuracy vs batch/epoch. Finally, you should report classification scores (accuracy, recall, confusion matrix, etc) on the test set.

Question 2 The dataset given in the previous question includes images taken for both right and left hands. Repeat the process in the previous question by designing

a classifier that outputs the number **and** the hand orientation. For example, the output for the example image given in the previous question should be (3-right). Report on the training, validation, and testing results.

Question 3 Replace your designed architecture in the previous question with the pre-defined ResNet18 architecture in PyTorch, to be **trained from scratch**. While maintaining the same hyperparameters and data pre-processing techniques (aside from resizing), compare the performance of the two architectures by plotting training/validation accuracy and reporting on the testing results.

Question 4 Repeat the process in the previous question by using the pre-defined ResNet18 with transfer learning, by loading the "ResNet18_Weights.IMAGENET1K_V1". Report on the training, validation, and testing results.