COMP 433 Introduction to Deep Learning (Fall 2025)

Major Assignment #3

Due: 11:59PM, November 23rd, 2025

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

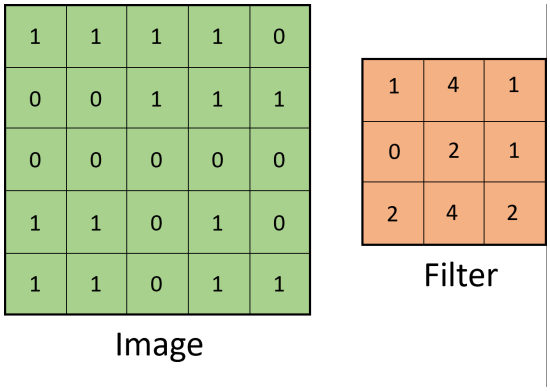
(a) **One(1) .pdf file:** containing answers to questions as well as reported results and snapshots from coding you develop. **Please write all the steps**. It is not necessary to type your answers. You may write them by hand in a legible manner and include photos of your solution as part of your submission.

(b) **One(1) .zip folder:** containing all developed Python ~~codes~~ notebooks with outputs. As mentioned, the snapshots of the outputs should be included in the PDF file.

(c) **Format:** For both files, please use a firstname_lastname_stuid formt. Example: John_Doe_1234567.pdf. If your first name or last name is composed of multiple parts, you can just write one part.

Theoretical Questions

Question 1 **(20 Points)** For each of the following cases, apply convolution using the given filter with the specified parameters to the input volume. Refer to PyTorch documentations on how to handle cases with fractions.



(a) **(4 Points)** No padding, Stride = 1

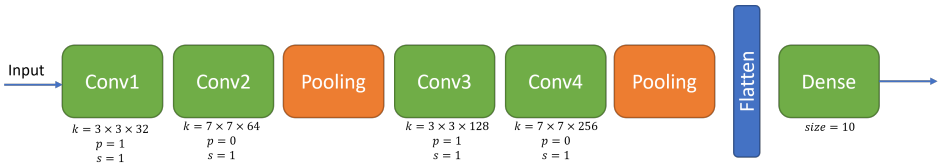
(b) **(4 Points)** No padding, Stride = 2

(c) **(4 Points)** 1 pixel padding, Stride = 1

(d) **(4 Points)** MaxPooling with kernel size = 2 and stride of 2 (no filter).

(e) **(4 Points)** No padding, Stride = 1, Dilation 2 (same filter should be adjusted for dilation 2).

Question 2 **(10 Points)** For the following architecture, write down the dimensions of the output volume (in $H \times W \times N$ format) for each layer. Refer to PyTorch documentations on how to handle cases with fractions. (Pooling refers to Max Pooling with a 2x2 kernel and no overlap). The input image size is $64 \times 64 \times 3$.



Question 3 **(15 Points)** Consider the following convolutional neural network (CNN) that predicts the class of images between two available classes (binary classification). Color RGB Images with the size of 32x32 train this network. In the first convolutional layer, we do not want any down sampling to occur, so we used 1-pixel padding.

(a) 4p How many neurons in the first convolutional layer are required to implement this CNN? (Hint: We have 3D feature maps) [Justify]

Component	Description
Images	Image size: 32×32 RGB color pictures
First Convolutional Layer	Filter size: 3×3, 16 filters, stride = 1, padding = 1
Second Convolutional Layer	Filter size: 3×3, 16 filters, stride = 1, no padding
Average Pooling	2×2 filters, stride = 2
Fully Connected (FC) Layer	ReLU activation
Output	Binary classification

Table 1: Architecture description of the convolutional neural network.

(b) 3p How many weights and how many bias terms in the first convolutional layer do we need to learn? [Justify]

(c) 2p How many neurons in the second convolutional layer are required [Justify]

(d) 2p How many weights and bias terms in the second convolutional layer do we need to learn? [Justify]

(e) 2p How many features does the whole convolutional part (Encoder) provide for the Fully-connected part (i.e., output volume of the Relu layer after the max-pooling) [Justify]?

(f) 2p In general, why is it preferable to use multiple convolutional layers with smaller filters rather than a single convolutional layer with a larger filter—even if both approaches produce feature maps of the same size?

Question 4 **(5 Points)** Consider the convolutional network described in the previous question. Suppose that instead of using a fully connected (dense) layer at the end of the network, we decide to replace it with a Logistic Regression classifier. Explain how this replacement can be implemented and determine the size (dimensionality) of the input to the Logistic Regression model, given the CNN in the previous question.

Question 5 **(5 Points)** Assume you have a 4x3 one-channel input volume with the following pixel intensities. A 2x2 filter is applied to the image followed by a ReLu. Find the resulting activation map (output volume) before and after pooling when no padding is applied, followed by 2x2 **average pooling**.

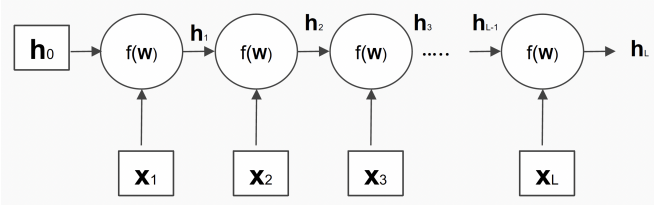
Input volume (4x3)

$$\begin{bmatrix} -1 & 0 & 1 \\ 2 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 10 \end{bmatrix}$$

Filter (2x2)

$$\begin{bmatrix} 2 & 1 \\ 0 & -1 \end{bmatrix}$$

Question 6 (20 Points) The following architecture for a simple Recurrent Neural Network (RNN) is given:



In this network, all the inputs, hidden states, and weights are assumed to be one dimensional. We use this network for a many-to-one regression task. The network produces an output y only at the last time step (L), which is given as $y = f(w_{hy}h_L)$. Assume no activation functions is used. Answer the following questions.

(a) **(4 Points)** Write a general expression for h_t in terms of w_{in}, w_{hh}, x_t, and h_{t-1}, where t is the timestep.

(b) **(4 Points)** Given the sequential input $\mathbf{x} = [0.3, 0.4, 0.2]$, the initial hidden state $h_0 = 0.5$, and all weights initialized to 0.2, compute h_1, h_2, h_3, and y.

(c) **(8 Points)** You want to perform *Backpropagation Through Time (BPTT)*, which is used to update the weights of RNN networks. Given a target output y_r, Assume that you are using the function $l = \frac{1}{2}(y-y_r)^2$ to calculate the loss. The goal is to compute $\frac{\delta l}{\delta w_{in}}$, $\frac{\delta l}{\delta w_{hh}}$, and $\frac{\delta l}{\delta w_{hy}}$, and use them in the typical update rule given as $w_i \leftarrow w_i - \eta\frac{\delta l}{\delta w_i}$. Find the expressions for $\frac{\delta l}{\delta w_{in}}$, $\frac{\delta l}{\delta w_{hh}}$, and $\frac{\delta l}{\delta w_{hy}}$. Your expressions can include only the following: y_r, h_0, w_{in}, w_{hh}, w_{hy}, and x_i for $i \in [1, 2, 3]$. Show your work. (hint: for $\frac{\delta l}{\delta w_{in}}$ and $\frac{\delta l}{\delta w_{hh}}$, you need to consider the sum across all timesteps)

(d) **(4 Points)** Assume that the target for the previously given data sequence is $y_r = 0.4$, with a learning rate of $\eta = 0.15$. Calculate the updated value of each weight.

Question 7 (5 points) Consider a transformer with an input sequence $X = \{x_1, x_2, x_3\}$, where $x_i \in \mathbb{R}^3$ represents the feature vector at each input position. Explain how the self-attention mechanism calculates the output representation for x_1. Explain how query, key, and value vectors are being used.

Implementation Questions

Question 1 **(15 Points)** 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). The dataset provided comes with train/test folders. You should keep the folders as is and use the given split. You can use the test folder for validation. 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 **(10 Points)** 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. (note: you cannot change the structure of any of the Conv layers in ResNet18. You can only change the last FC layer to match the number of outputs needed.).

Question 3 **(20 Points)** Build an LSTM-based model for time-series forecasting using PyTorch. Given a series of data points, the model should be able to predict the next data point. You should use the Daily Climate Time Series dataset for this task. We will only use the "meantemp" column, where the aim is to use the temperatures from the previous days to predict the next temperature value.

(a) **(5 Points)** We will only use the "meantemp" column to train our model, hence you should remove the remaining columns from the dataset. Additionally, since our prediction is based on the historical trend, each data point (each row) in the dataset should be in the form of sequence -> prediction. Assume we want to use a sequence window of size 10 in this problem. Preprocess the dataset such that each sequence of 10 values is used to predict the 11th value. You can refer to this article for some examples on how to preprocess the data into sequences. (hint: if your original dataset has 100 data points, then the preprocessed dataset would have 100-10=90 data points/sequences).

(b) **(3 Points)** Preprocess the data using the MinMaxScaler from scikit-learn. There is no need to split or shuffle the data for this question.

(c) **(9 Points)** Train a vanilla LSTM to forecast the mean temperature based on the last 10 values. You can refer to this article for some guidance. You are free to choose and optimize the hyperparameters (optimizer, batch size, etc), but you should use the MSE loss. Aim for better performance.

(d) **(3 Points)** Analyze the performance of your model by plotting the training loss. Using the available data, compare the original plot of the data points with the predicted plot (each data point on the predicted plot is obtained by feeding the previous 10 data points from the original data).

Question 4 **(15 points)** In the previous question, you worked on designing an LSTM-based model for time-series forecasting using PyTorch. In this question, you are required to address the same problem using a transformer-based model. Given a series of data points, the model should be able to predict the next data point. You should use the Daily Climate Time Series dataset for this task. We will only use the "meantemp" column, where the aim is to use the temperatures from the previous days to predict the next temperature value. You are free to choose the architecture and the hyperparameters such that the performance is optimized. The aim is to use the last 10 temperature values to predict the next value. Refer to the third assignments for suggestions on how to preprocess the dataset.