Hong Kong University of Science and Technology
COMP 4211: Machine Learning
Spring 2021

Programming Assignment 2
Due: 26 March 2021, Friday, 11:59pm

1 Objective

The objective of this assignment is to practise the use of the `PyTorch` machine learning framework through building and tuning a Siamese convolutional neural network to solve the face comparison problem. Throughout the assignment, students will be guided to develop the model step by step and study the effects of different model configurations.

2 Major Tasks

The assignment consists of a coding part and a written report:

CODING: Build a face comparison system using `PyTorch`.

WRITTEN REPORT: Report the results and answer some questions.

The tasks will be elaborated in Sections 4 and 5. Note that [Qn] refers to a specific question (the nth question) that you need to answer in the written report.

3 Setup

- Make sure that the libraries `torch`, `torchvision`, `tensorboard` and `matplotlib` have been installed on your machine.
- For this assignment, it suffices to use only `PyTorch` with `Python 3.7` or above. Other machine learning frameworks including `TensorFlow` and `Keras` should not be used.
- You are highly recommended to use GPU resources like those provided by the GPU servers of the Department of Computer Science and Engineering (`https://cssystem.cse.ust.hk/Facilities/ug_cluster/gpu.html`) or the Google Colab (`https://colab.research.google.com`).
- The dataset files for this assignment are provided as a ZIP file (`pa2.zip`).

4 Face Comparison System

A face comparison system essentially answers this question: does the face in an image match the face in another image? A face comparison system takes two face images as input and predicts their similarity score. The system then compares the score with a predefined similarity threshold and decides whether the faces match. For example, in a Face ID application used to authenticate users, if the similarity threshold is set at 0.8, then the application will return a match when the predicted similarity score exceeds 0.8 and no match otherwise. In this assignment, you will build a face comparison model using a Siamese convolutional neural network (Siamese CNN) which will be described below.

4.1 Siamese Convolutional Neural Network

From the lectures and tutorials, you have learned different multi-class classification models which handle one image per forward pass. However, in the face comparison problem, the system processes two images in each pass. This can be achieved by using the Siamese network architecture as illustrated in Figure 1.

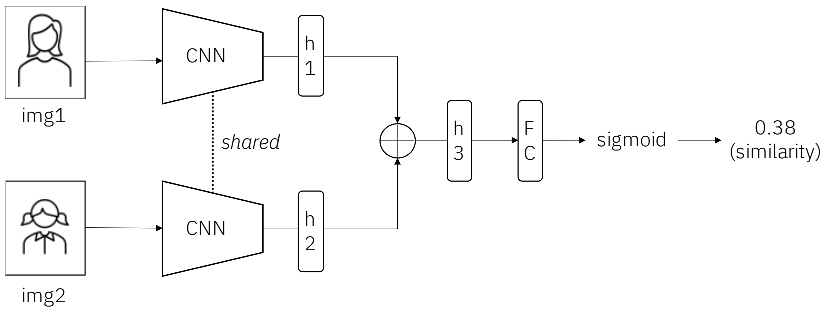

Figure 1: A simple Siamese network for face comparison

The word 'Siamese' means joined or connected. The Siamese network you will use for this assignment consists of three components: convolutional layers (CNN), aggregation function ($\oplus$), and fully connected layers (FC). In a forward pass, *img1* and *img2* go through the **same** convolutional layers and are flattened as the hidden state vectors *h1* and *h2*, respectively. The vectors *h1* and *h2* are combined to give *h3* using a given aggregation function $\oplus$, such as *h3* = $|h1 - h2|$. The aggregated state vector *h3* then passes through some fully connected layers followed by a sigmoid function to give a similarity score between 0 and 1.

4.2 Tasks

This section will guide you to build the Siamese CNN model.

4.2.1 Dataset and Dataloader

The Face Dataset can be found in the `./pa2` directory. All csv files contain two id columns (**id1** and **id2**) where each row shows the indices of the two images being compared. In `train.csv` and `valid.csv`, the **target** column contains the ground-truth labels (1: match; 0: not match). Notice that the target labels of `test.csv` are not provided. With the indices provided in the csv files, you can look up the corresponding image paths from `index.txt`. Table 1 summarizes the four files. All the face images are located under the `/Images` directory.

Table 1: Description of the dataset

File	# of instances	Has label?	Purpose
`index.txt`	-	-	look up image paths
`train.csv`	4,900	yes	training
`valid.csv`	700	yes	validation
`test.csv`	700	no	prediction

Your first task is to implement a custom `Dataset` class for the Face Dataset. You are required to preprocess the images by: (1) resizing all the images to a size of **32x32**, and (2) converting them to **single-channel** images. The `Dataset.getitem()` function should return two 32x32, single-channel images and one target label.

Create the `Dataset` and `Dataloader` objects for `train.csv`, `valid.csv` and `test.csv`. You are suggested to use a batch size of 128 for model training.

4.2.2 Siamese Convolutional Neural Network

Convolutional Layers

The convolutional layers take an image as input and produce a hidden state vector as output. Here, you need to build a stack of convolutional layers according to the network architecture illustrated in Table 2. In addition, each convolutional layer is followed by **batch normalization** and a **ReLU activation**. Notice that the kernel size of the final average pooling layer is not provided. Given that the size of the output vector is [1 x 512], calculate the correct kernel size for the final average pooling layer and implement it in your model.

Table 2: Description of the convolutional layers

Layer	Kernel size	# of kernels	Stride	Padding
Input	1 x 32 x 32 image			
Conv	3 x 3	32	1	1
Conv	3 x 3	32	1	1
Max pooling	2 x 2	-	2	0
Conv	3 x 3	64	1	1
Conv	3 x 3	128	1	1
Conv	3 x 3	256	1	1
Conv	3 x 3	512	1	1
Average pooling	? x ?	-	?	0
Output	1 x 512 vector			

[Q1] What is the correct kernel size in the average pooling layer? Explain your answer. (Hint: You may assume that there is a flattening operation after the average pooling layer.)

Aggregation Function

We take the absolute difference as the aggregation function. To be specific, the aggregation function takes in two input vectors of size 512 and outputs the element-wise absolute difference vector of size 512.

Fully Connected Layers

The aggregated vector goes through two fully connected layers. There are 512 hidden units in the first layer, followed by a **ReLU activation** and a **dropout layer** with 0.5 drop probability. The second layer is the output layer. Apply the **sigmoid** function to obtain a similarity score in the range from 0 to 1.

[Q2] What is dropout in deep learning? Why does it work? Explain it briefly.

Siamese Network

Connect the convolutional layers, aggregation function, and the fully connected layers to complete the `forward` function of the Siamese network model. The function takes in two images and outputs a similarity score between 0 and 1. Remember that the two input images go through the same convolutional layers. Write a helper function to count the total number of trainable parameters and report it in Q3.

[Q3] What is the total number of trainable parameters in this Siamese network model?

[Q4] If we replace the absolute difference aggregation function by concatenation (i.e., concatenating the hidden state vectors of the two input images into one before feeding into the fully connected layers), what is the change in the total number of trainable parameters when compared to the original setting?

4.2.3 Training and Validation

The model is trained by minimizing the binary cross-entropy loss[1] using the Adam optimizer with the default setting. You are expected to train your model for 20 epochs. During training, you are required to record the training and validation losses after every 10 steps (one step means one gradient update). Plot the training and validation loss curves versus the number of steps in the same graph using `Tensorboard` or `matplotlib`.

[Q5] Paste the screenshot of the training and validation loss curves and report the final training and validation losses obtained. According to your plot, when should you stop training? Why?

[1]Two binary cross-entropy loss functions are provided by `PyTorch`: `BCELoss` and `BCEWithLogitsLoss`. You should check the documentation carefully for the correct loss function to use.

4.2.4 Evaluation

Output the prediction of the validation set as 1 if the similarity score exceeds a predefined threshold θ, and as 0 otherwise. Write a helper function to search for the optimal θ that maximizes the validation accuracy.

[Q6] Report the optimal threshold θ and the corresponding validation accuracy.

[Q7] Suppose you are given a photo application used to identify similar-looking family members and a face authentication application used for mobile banking. Should you set a relatively high or low threshold for each of the applications? Why?

4.2.5 Improving your Model

Now, improve your model for better validation performance. You are required to make changes in **at least two** of the five categories (A-E) defined below. You may duplicate your previous code and modify it from there. Alternatively, you may also modify your previous code to accept new arguments for different configurations. Make sure that your changes are clearly shown in the code (e.g., you can put a comment like '`# Q8A`' in your code to indicate the changes for category A) and **do not overwrite your work for the previous tasks (Section 4.2.1 – 4.2.4)**. In addition to the changes listed below, you may make further changes to the model or training procedure to maximize the validation performance.

A. Optimization: Experiment with different optimization settings, e.g., learning rate, weight decay, the choice of optimizer and/or the number of training epochs. Report the validation accuracy for at least 3 different configurations in Q8.

B. Aggregation function: Replace or complement the current aggregation function with other aggregation method(s). Describe the final aggregation function and report the validation accuracy in Q8 for at least 3 different settings.

C. Data augmentation: Propose and implement two image augmentation methods. You can use the image transformation functions provided by `torchvision.transform`. Describe the augmentation methods and briefly explain how they improve model training for the face comparison task and report the validation accuracy in Q8.

D. Architecture: Use a more complex model architecture, e.g., by increasing the number of layers in the existing model or adopting the VGG-11 or ResNet-18 model.[2] Report the architecture and the validation accuracy for at least 3 different settings in Q8.

E. Contrastive loss (advanced): Use `CosineEmbeddingLoss`[3] provided by `PyTorch` in your model. You have to make necessary changes to the model and the training procedure. In Q8, state the loss function, explain how it helps the model learn the face comparison task, and report the validation accuracy for at least 3 different margin values in `CosineEmbeddingLoss`.

[Q8] Describe your changes and compare the validation performance with the original setting.

[2]You can reference the architecture design of these existing models. However, you need to implement them from scratch, i.e., you cannot use any pre-built or pre-trained model from the `torchvision.model` library.

[3]`https://pytorch.org/docs/stable/generated/torch.nn.CosineEmbeddingLoss.html`

Discuss why the changes improve model performance. You are expected to use visual aids like a graph or a table to explain your answer.

4.2.6 Prediction

Finally, you have built your own face comparison model! The last step is to predict the similarity scores in `test.csv` and save the predictions as `p1.csv`. Your predictions should be aimed to produce an optimal accuracy score.

[P1] Submit your predictions as `p1.csv`. The format of `p1.csv` should be the same as that of `train.csv`: first two columns for the image indices and the third column for the predicted label (1: match; 0: not match). It is important to **keep the row order** to be the same as that in `test.csv`.

5 Written Report

Answer [Q1] to [Q8] in one single written report.

6 Some Programming Tips

As is always the case, good programming practices should be applied when coding your program. Below are some common ones but they are by no means complete:

- Using a small subset of data to test the code
- Using checkpoints to save partially trained models
- Using functions to structure program clearly
- Using meaningful variable and function names to improve readability
- Using consistent styles
- Including concise but informative comments

7 Assignment Submission

Assignment submission should only be done electronically using the Course Assignment Submission System (CASS):

`https://cssystem.cse.ust.hk/UGuides/cass/student.html`

There should be three files in your submission with the following naming convention required:

1. **Report** with filename `<StudentID>_report.pdf`: in PDF format.

2. **Prediction** with filename `<StudentID>_p1.csv`: in csv format.

3. **Source code and a README file** with filename `<StudentID>_code`: compressed into a single ZIP or RAR file.

Note: The source code files should contain all necessary code for running your program as well as a brief user guide for the TA to run the programs easily to verify your results, all compressed into a single ZIP or RAR file. The data should not be submitted to keep the file size small. When multiple versions with the same filename are submitted, only the latest version according to the timestamp will be used for grading. Files not adhering to the naming convention above will be ignored.

8 Grading Scheme

This programming assignment will be counted towards 15% of your final course grade. The maximum scores for different tasks are shown below:

Table 3: [C]: Code, [Q]: Written report, [P]: Prediction

Grading Scheme	Code (62)	Report (33)	Prediction (5)
Dataset and Dataloader (10)			
- [C1] Dataset Class	4		
- [C2] Image Preprocessing	4		
- [C3] Dataloader	2		
Model Design (30)			
- [C4] CNN Model + [Q1]	4	+4	
- [C5] Aggregation Function + [Q4]	4	+4	
- [C6] Fully Connected Layer + [Q2]	4	+2	
- [C7] Siamese Network + [Q3]	6	+2	
Training and Validation (38)			
- [C8] Loss Function	2		
- [C9] Optimizer	2		
- [C10] Training Loop	6		
- [C11] Validation Loop	4		
- [C12] Loss Curve + [Q5]	2	+6	
- [C12] Model Evaluation + [Q6]	6	+6	
- [Q7] Application		4	
Improving your model (13)			
- [C13] Model Improvement 1	4		
- [C14] Model Improvement 2	4		
- [Q8] Model Improvement		+5	
Prediction (9)			
- [C15] Prediction + [P1]	4		+5

Late submission will be accepted but with penalty. The late penalty is deduction of one point (out of a maximum of 100 points) for every minute late after 11:59pm. Being late for a fraction of a minute is considered a full minute. For example, two points will be deducted if the submission time is 00:00:34. At most one NQA coupon may be used to entitle you to submit this assignment late for one day without grade penalty.

9 Academic Integrity

Please refer to the regulations for student conduct and academic integrity on this webpage: `https://acadreg.ust.hk/generalreg`.

While you may discuss with your classmates on general ideas about the assignment, your submission should be based on your own independent effort. In case you seek help from any person or reference source, you should state it clearly in your submission. Failure to do so is considered plagiarism which will lead to appropriate disciplinary actions.