Face Comparison Siamese Convolutional Neural Network using Pytorch

COMP 4211 Programming Assignment 2

Lee Jae Yeol (20308109) | jyleeag@connect.ust.hk

Introduction

For programming assignment 2, 10 frontal images are provided for 500 people in the train dataset and 700 individual images in the test set. One label column is provided for each train and validation dataset, whether the pair of the frontal images are the same person (1) or not (0).

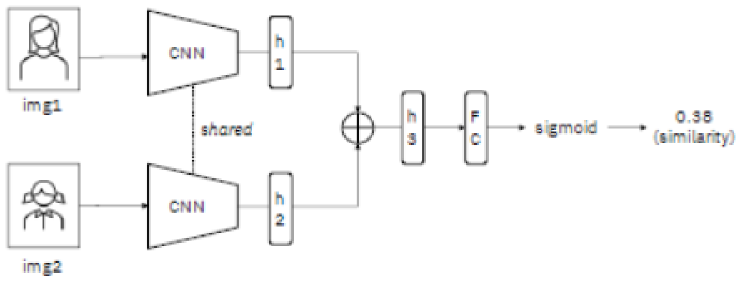
The main objective of pa2 is to be familiar with Pytorch machine learning framework and to implement a Siamese convolutional neural network for face comparison, outputting a similarity score for each of the pair of the frontal images. After the dataset and data-loader classes are configured in the first part, the baseline of the Siamese CNN model and the train function are built. Then, the model undergoes several enhancement techniques, ones that are chosen for assessment are different hyperparameter tunings, data augmentation through image transformation and CosineEmbeddingLoss function implementation. Please note that the CosineEmbeddingLoss hasn't been adopted fully and the model was unsuccessful.

Experiment Environment

CPU	Memory	Python	PyTorch
Intel(R) Core (TM) i5-8250U CPU @ 1.60GHz	8 GB	3.7.9	1.8.0+cu101

Part 1: Siamese Convolutional Neural Network

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]

The correct kernel size in the average pooling layer is 16. Since the input size for the average pooling layer is [128, 512, 16, 16] and we are expecting an output size of [128, 512, 1, 1], average pooling of kernel 16 will output an 1 x 1 image data. The full input output shapes with batch size = 128 is as follows.

Layer	Kernel Shape	Output Shape	Params	Mult-Adds
0_conv.0.Conv2d_0	[1, 32, 3, 3]	[128, 32, 32, 32]	320.0	294912.0
1_conv.0.Conv2d_0	[1, 32, 3, 3]	[128, 32, 32, 32]	NaN	294912.0
2_conv.0.BatchNorm2d_1	[32]	[128, 32, 32, 32]	64.0	32.0
3_conv.0.BatchNorm2d_1	[32]	[128, 32, 32, 32]	NaN	32.0
4_conv.0.ReLU_2	-	[128, 32, 32, 32]	NaN	NaN
...	...	...	...	...
79_conv.5.AvgPool2d_3	-	[128, 512, 1, 1]	NaN	NaN
80_fc1	[512, 512]	[128, 512]	262656.0	262144.0
81_drop	-	[128, 512]	NaN	NaN
82_fc2	[512, 1]	[128, 1]	513.0	512.0
83_sigmoid	-	[128, 1]	NaN	NaN

84 rows × 4 columns

Fully Connected Layers

[Q2]

In deep learning, dropout is equivalent to 'switching off' some arbitrary nodes of the network. It's simple method has proven its efficiency in mitigating overfitting issues and improving generalization errors.

One method of preventing overfitting the data is to average all the predictions from different neural networks. However, this approach is time-consuming and requires huge computations. Dropping out provides a similar mechanism in the sense that as the nodes are arbitrarily turned off, the same neural network acts as different neural networks. So, through dropout, generalization errors are improved, and overfitting issues are also addressed simultaneously.

Siamese Network

[Q3]

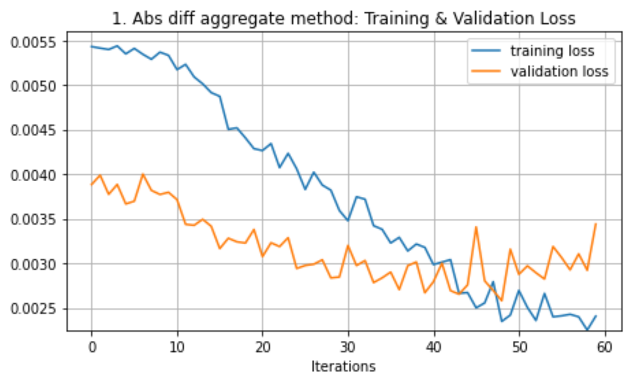
The total number of trainable parameters in the Siamese network model above is equal to 1842465.

[Q4]

The total number of trainable parameters for the aggregation function by concatenation is 2104609.

Part 2: Training and Validation

[Q5]



According to the plot above, the training should be stopped when iterations reach above 40, which is around 13 to 14th epoch. The reason is that when the validation accuracy already reached a ceiling for the model in that iteration, validation loss started to move sideways and increase, showing sign of overfitting.

Part 3: Evaluation

[Q6]

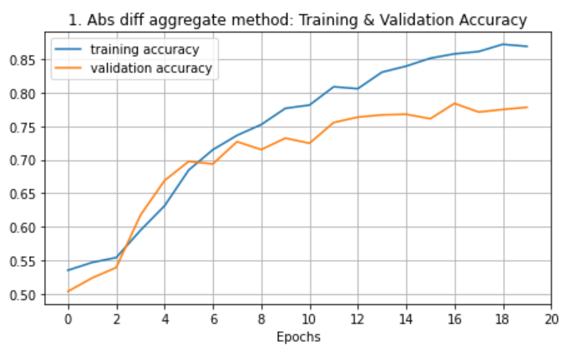
The optimal threshold came out to be 0.45 with the corresponding validation accuracy 0.7790 for the model above.

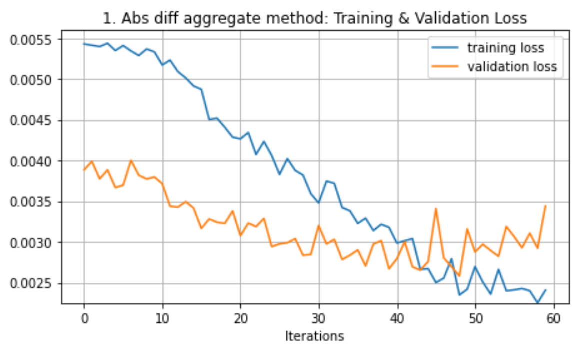
[Q7]

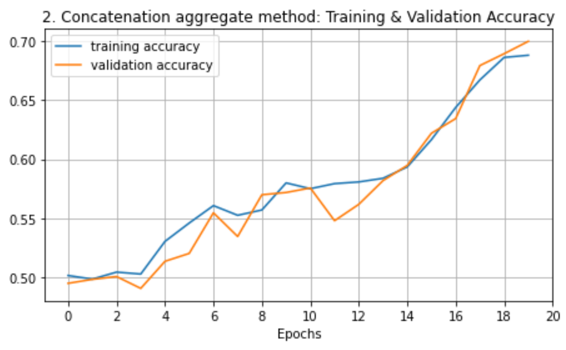
If the model is being implemented for similar-looking family members, an average to low threshold is sufficient. On the other hand, if its usage is face authentication for mobile banking, then a high threshold is imperative. The reason is that mobile banking must be highly private and secure so that others can't access without authorization. Therefore, if the face matching score isn't high, the high threshold will be able to deny the access.

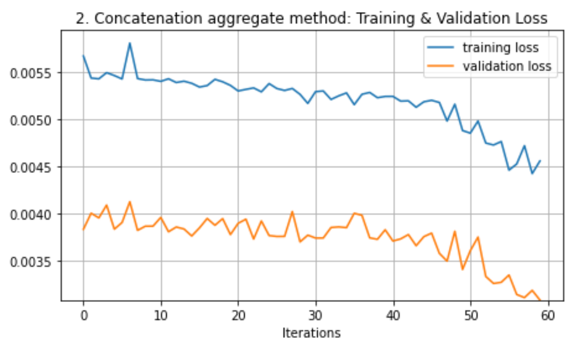
Part 4: Improving your Model

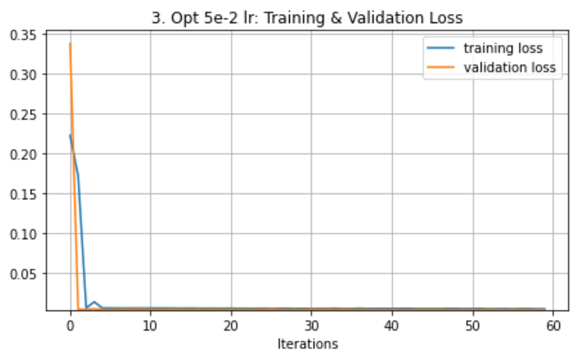
[Q8]

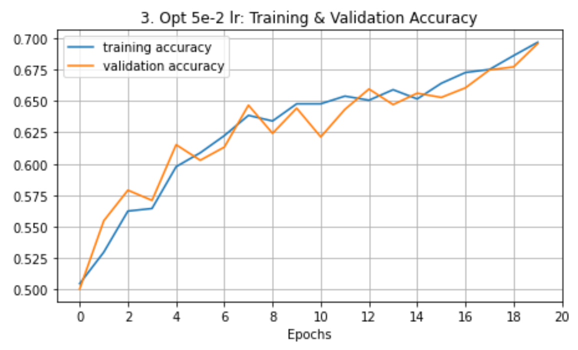


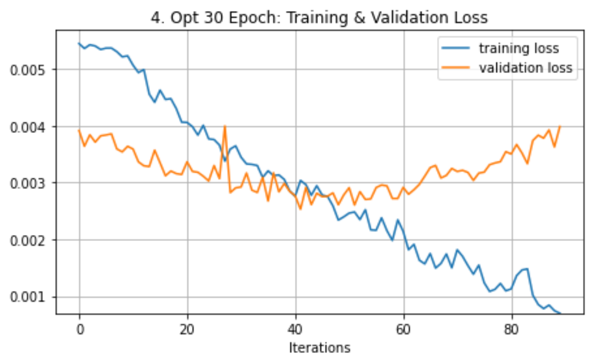


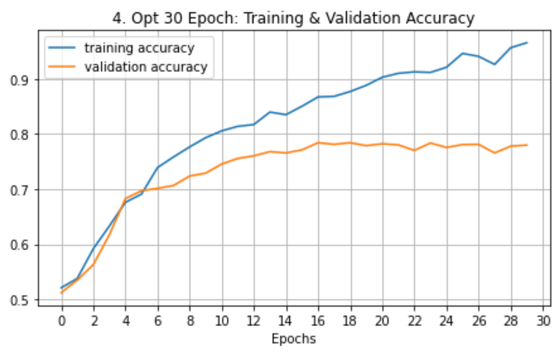


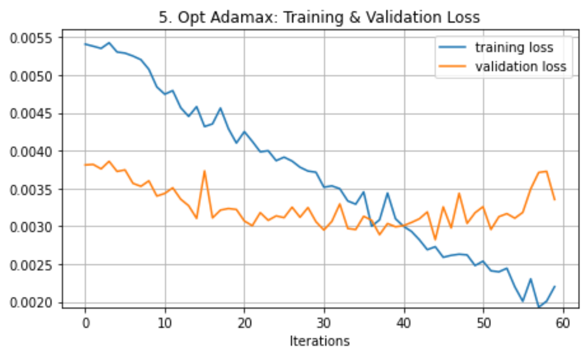


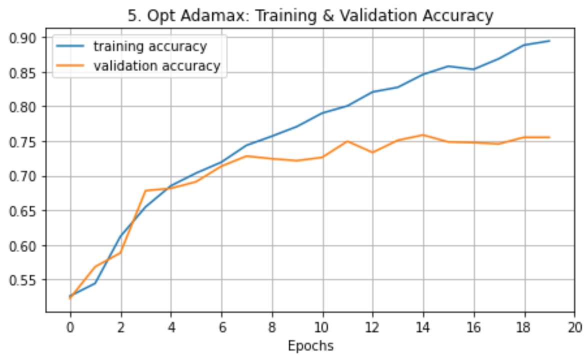


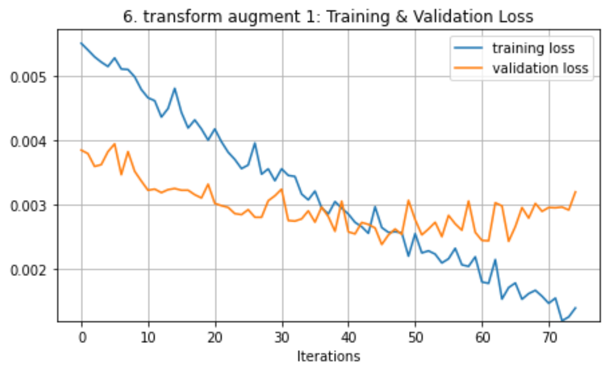


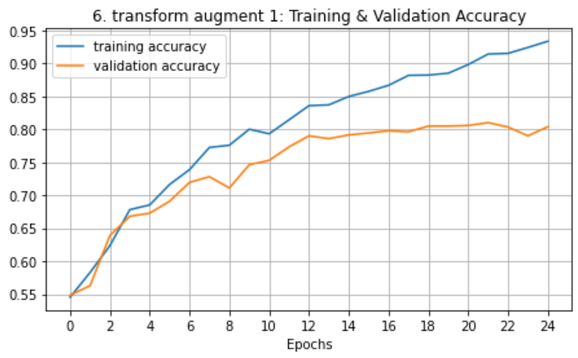


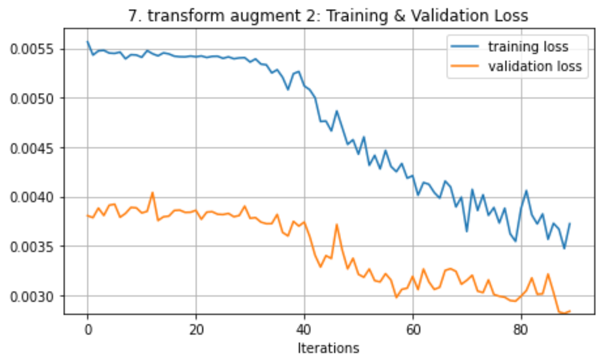


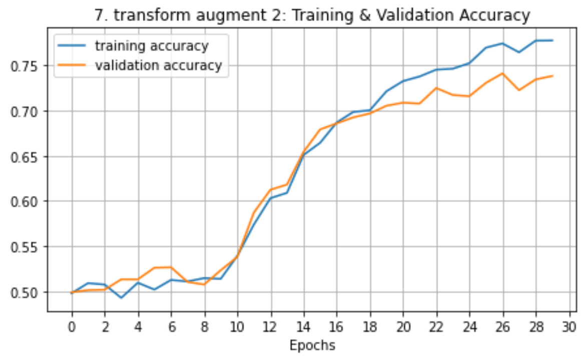












Validation Set Result Table

	Loss Score	Accuracy Score
1. Abs diff aggregate method	0.003439	0.778571
2. Concatenation aggregate method	0.003079	0.699524
3. Opt 5e-2 lr	0.003602	0.695714
4. Opt 30 Epoch	0.003987	0.78
5. Opt Adamax	0.003353	0.755238
6. transform augment 1	0.003198	0.80381
7. transform augment 2	0.002839	0.738095

Model 6 transform augment 1 gave out the highest accuracy score (0.80381) among all the models. Data augmentation, in this case image augmentation, is a technique of enlarging data size by adding images that are transformed from the original images. Since the images are transformed with the predefined probability in every iteration, having more epochs will be able to train the model with more augmented (i.e. transformed) image data and thus, result in a higher validation accuracy. Other models failed to show a noticeable difference in the validation accuracy score compared to the first model implemented.

To describe transform augment 1 model, random horizontal flip with probability 0.7 and normalization method has been added into the base (i.e. model1). As explained above, epoch was initially set to 25 for sufficient training.

Contrastive loss: Use CosineEmbeddingLoss

$$\text{loss}(x, y) = \begin{cases} 1 - \cos(x_1, x_2), & \text{if } y = 1 \\ \max(0, \cos(x_1, x_2) - \text{margin}), & \text{if } y = -1 \end{cases}$$

CosineEmbeddingLoss evaluates the loss of the model with two image tensors and the correct label as input. The main objective of the CosineEmbeddingLoss function is to measure whether the model can differentiate the two input images and decide whether the two are similar or not. It is a very suitable loss function for a face comparison model. The label originally 0, 1 needs to be modified to 2 * label - 1 for the label to be either 1 or -1.

Part 5: Prediction

[P1]

p1.csv has been saved with the same row order as test.csv