CSC420 Assignment 5
1002077726
Guanxiong Liu
liuguanx

1a. Yes, it is a good to train binary classification neural networks since I only have five images per employee. The available data is relatively small, so it would be appropriate to have a binary classifier for each employee. i.e. One vs. All classifier.

1b. TF-IDF is useful because common objects (visual words) should weigh less when match an image in a query since if every picture has this visual word then this word is not salient enough to be a candidate of matching the two pictures.

1c. Please see code in facenet.py

1d. The embeddings have to be clustered to calculate mean (separate different faces), thus we can use mean to build inverted index for fast image retrieval.

1e. Choose cluster's center (mean) vector as visual word representation.

1f. Please see code in facenet.py

1g. First encode each image to embedding, then feed it into kmeans.predict(). When I got the classification (0~5) then I look up visual word representation in kmeans.cluster_centers_[classification]. Then I calculate the Euclidean distance between the embedding and the visual word, as long as the distance is less than 0.5, I consider they are matched. So, I look up in the inverted index dictionary[classification] and get all images that are belong to this class.

1h. Please see the code and output file matching.csv.

1i. In this case, each image should have more than one embeddings because there are more than one faces. Then we consider multiple visual words appear in the same image. The challenge is that we do not have a visual word representation in this case. When we have only one face in an image, we can simply do clustering to separate different faces, but in this case, we are not able to do clustering to choose a visual word representation, thus is hard to find all faces that are close to the representation.

2a. When we do back propagation (using gradient based optimization techniques) and calculating the gradients of loss with respect to the weights, the gradients is getting smaller and smaller, so the hidden layers that close to input learn from loss very slow. When you have a large neural network, the gradients in the earlier layers tend to be vanished. We can use ReLU activation function to mitigate the problem because the derivative for ReLU is 1 for all positive value and 0 for all negative value.

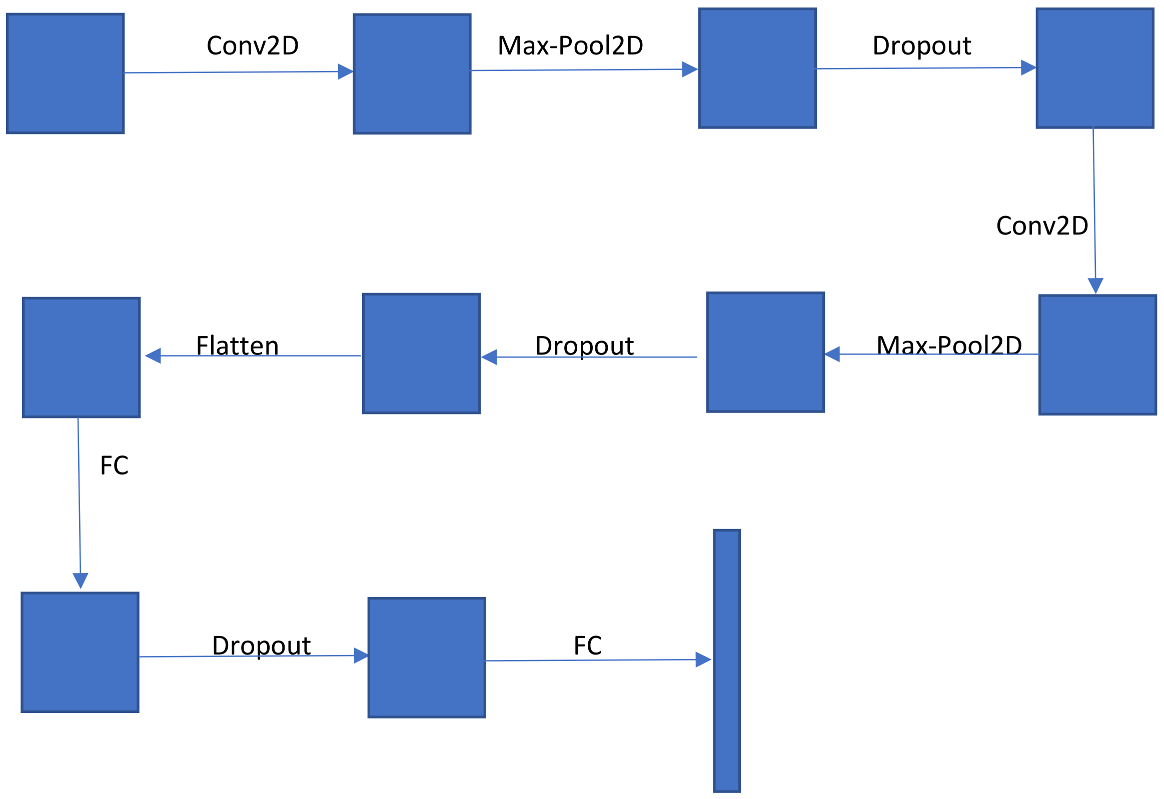
2b. Because the exact location of a feature is less important than its rough location relative to other features in an image, so the max pool exists to reduce the spatial size of the representation and reduce the number of parameters and computational cost.

2c. Please see code in fashion.py.

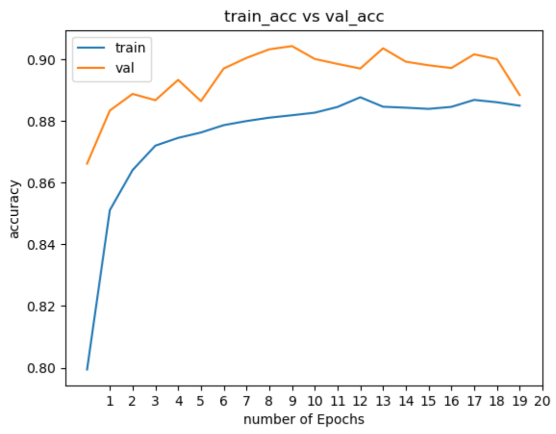
2d. I chose ReLU for the layers in the middle and SoftMax in the final layer because ReLU mitigates gradient vanishing problem and SoftMax assigns decimal probabilities to each class in a multi-class problem. The probabilities add up to 1. This constraint helps training converge more quickly.

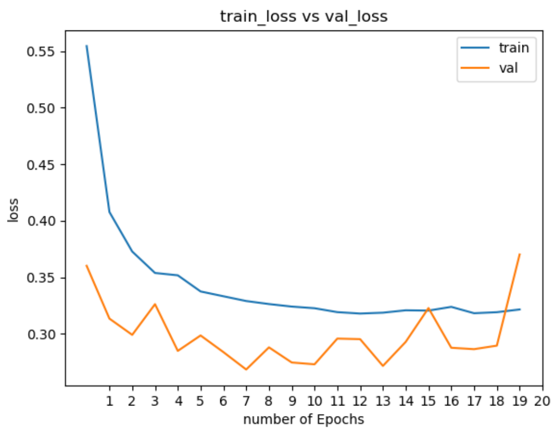
CSC420 Assignment 5
1002077726
Guanxiong Liu
liuguanx
2e. $L(y, \hat{y}) = -\sum_i y_i log\hat{y_i}$ where y is the ground truth and $\hat{y}$ is the prediction.

2f.



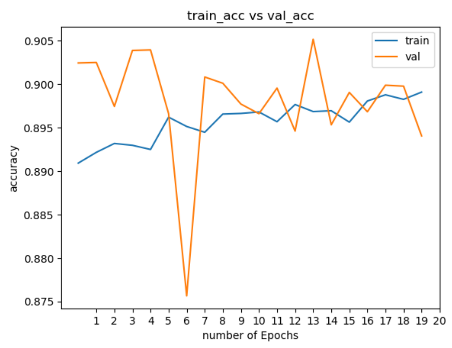
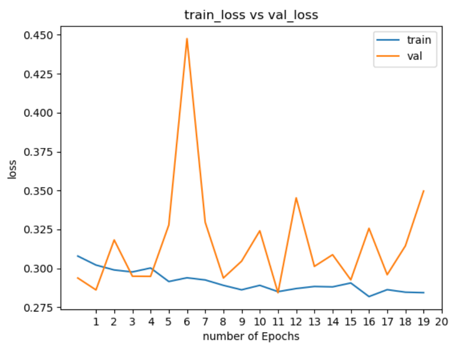
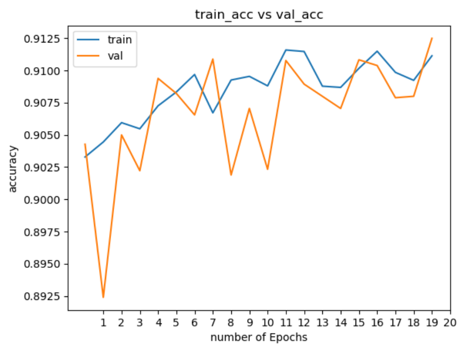
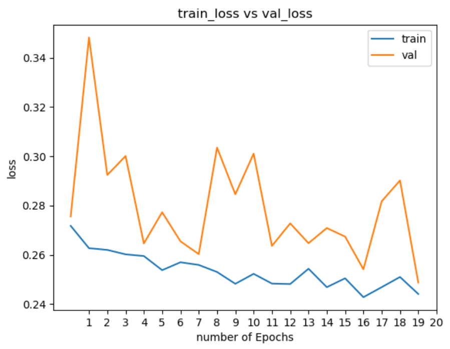
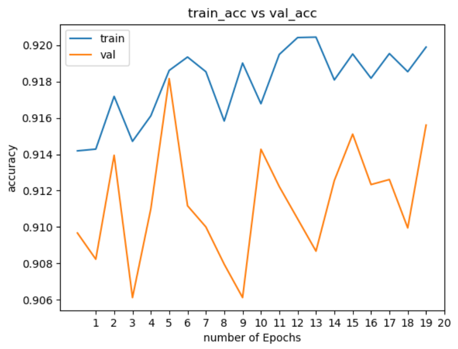
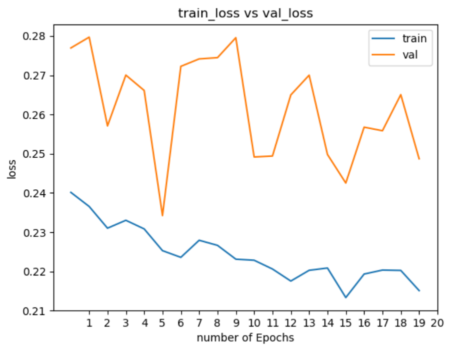
Learning rate=0.001, batch size=5, number of epochs=20, we should stop training when training accuracy increases, and validation accuracy steadily decreases because it is overfitting. Final training/validation loss are: 0.3215/0.3702, final training/validation accuracy are: 0.8850/0.8884.



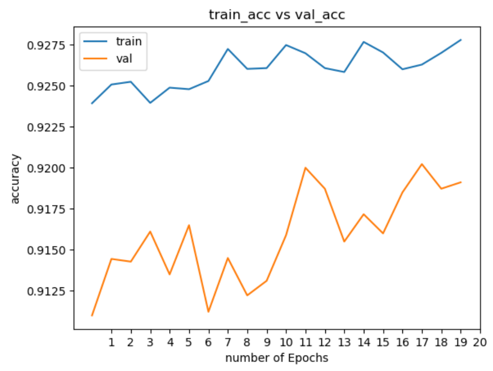
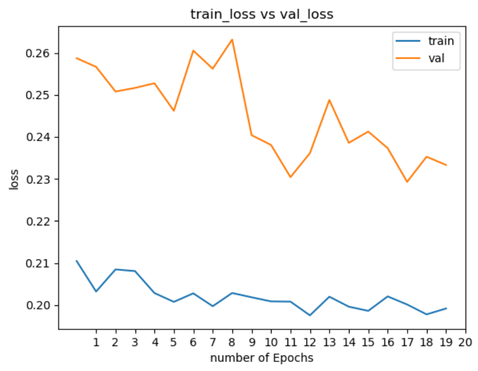


CSC420 Assignment 5
1002077726
Guanxiong Liu
liuguanx

2g. I found that the larger the batch size the higher the test accuracy. This makes sense because when we have larger batch size, we can get more correct gradient. So, when we look at the contour map, it moves to the minima more steadily. If we have relatively small batch size, it will fluctuate more on the contour map thus less accurate. Plots are below.

Batch size	plot
8	 
16	 
32	 

CSC420 Assignment 5
1002077726
Guanxiong Liu
liuguanx

64	 train_acc vs val_acc train val accuracy number of Epochs  train_loss vs val_loss train val loss number of Epochs