CS440

Xinyu Meng xm73 RUID:193002567

Qichao Lin ql180 RUID:196005340

Final Project Report

In the final project, we implemented the Perceptron and Naïve bayes classifier from scratch, and we utilized the Tensorflow library to build the Neural Network model.

Data loading:

I read the file into a list using readlines(), for face data, I split each image to 60x70 and store them into a list. For digit data, I split each image to 28x28 and store them into their corresponding lists. During training, I just utilize each pixel as a feature. So for each 2d list of image, I convert ‘ ’ into 0, ‘+’ into 1, and ‘#’ into 2. I flatten the matrix into a 1d array, which is easier to do the dot products using numpy. However, for the neural network with tensorflow, I had to convert the symbol into their corresponding values before the training due the implementation.

Perceptron:

I will initialize the weight vector as 0s, set the bias w0 as 0 also. For digit data, I initialize 10 different perceptron, during training, for each training image, I run predict function for each of the 10 perceptrons. If the prediction is correct, I don’t do any update. If prediction result is I and the true label is j, I will make the weight of PerceptronI as wik <- wki – feature_k(x) for each k in the weight vector and the bias w0 <- w0-1, and make the weight of Perceptron j as wjk <- wkj + feature_k(x), w0 <- w0+1. In our implementation, the calculation is achieved by summation of two np.arrays. During testing, I just utilize the weight vector I got for each

perceptron and get an list of 10 results. I pick the maximum value, and its digit(represented by its index in the list) is the output of the model. For face data, I just check if the prediction result is greater or equal to 0 or not.

Naïve bayes:

I initialize the probability as a vector of 0. I have three array stored in a single list, each representing the probability of the pixel being 0, 1, or 2. For digit images, I will first split the training set into 10 classes each with label = 0, 1, 2, 3….. I create 10 classifiers with digit 0 to 9. I fit the corresponding class of image into the classifier and get a probability vector. For testing, I just choose the maximum value returned by the model as output.

Neural Network:

I imported Tensorflow library to build neural networks. I use keras.sequential to create a simple neural network. I set the training epoch as 5 to get the model performance for 5 iterations. For face data, I created a convolutional neural network to better predict the faces.

Executed result:

accuracy for Perceptron on 20 percent digit image for 5 iterations: mean- 0.6786 standard deviation- 0.026446927987953516

time spent for training Perceptron on 20 percent digit image for 5 iterations: mean- 0.7010560989379883 standard deviation- 0.006051291022221893

accuracy for Perceptron on 40 percent digit image for 5 iterations: mean- 0.72 standard deviation- 0.03825702549859306

time spent for training Perceptron on 40 percent digit image for 5 iterations: mean- 1.503451442718506 standard deviation- 0.06108726806853301

accuracy for Perceptron on 60 percent digit image for 5 iterations: mean- 0.745 standard deviation- 0.018665476152512173

time spent for training Perceptron on 60 percent digit image for 5 iterations: mean- 3.1280555725097656 standard deviation- 0.5402991188604274

accuracy for Perceptron on 80 percent digit image for 5 iterations: mean- 0.7702 standard deviation- 0.027461973709112776

time spent for training Perceptron on 80 percent digit image for 5 iterations: mean- 4.05173921585083 standard deviation- 0.7266908061962182

accuracy for Perceptron on 100 percent digit image for 5 iterations: mean- 0.7632000000000001 standard deviation- 0.024152846623120866

time spent for training Perceptron on 100 percent digit image for 5 iterations: mean- 3.6727647304534914 standard deviation- 0.00828654250255467

accuracy for Perceptron on 20 percent face image for 5 iterations: mean- 0.7306666666666667 standard deviation- 0.06512893195637233

time spent for training Perceptron on 20 percent face image for 5 iterations: mean- -0.035579252243041995 standard deviation- 0.006655066287527573

accuracy for Perceptron on 40 percent face image for 5 iterations: mean- 0.7773333333333333 standard deviation- 0.027519689920733746

time spent for training Perceptron on 40 percent face image for 5 iterations: mean- -0.06981730461120605 standard deviation- 0.006789746075907763

accuracy for Perceptron on 60 percent face image for 5 iterations: mean- 0.7933333333333333 standard deviation- 0.048258562855610296

time spent for training Perceptron on 60 percent face image for 5 iterations: mean- -0.10855464935302735 standard deviation- 0.005256351630491313

accuracy for Perceptron on 80 percent face image for 5 iterations: mean- 0.7933333333333333 standard deviation- 0.03887301263230203

time spent for training Perceptron on 80 percent face image for 5 iterations: mean- -0.13887448310852052 standard deviation- 0.010528445353732757

accuracy for Perceptron on 100 percent face image for 5 iterations: mean- 0.82 standard deviation- 0.04361447262345636

time spent for training Perceptron on 100 percent face image for 5 iterations: mean- -0.16668963432312012 standard deviation- 0.009388631705467203

accuracy for Naive Bayes Classifier on 20 percent digit image for 5 iterations: mean- 0.7624000000000001 standard deviation- 0.009002221947941527

time spent for fitting Naive Bayes Classifier on 20 percent digit image for 5 iterations: mean- -0.12551140785217285 standard deviation- 0.008779260753667691

accuracy for Naive Bayes Classifier on 40 percent digit image for 5 iterations: mean- 0.776 standard deviation- 0.0053665631459995

time spent for fitting Naive Bayes Classifier on 40 percent digit image for 5 iterations: mean- -0.23476901054382324 standard deviation- 0.010032943630972938

accuracy for Naive Bayes Classifier on 60 percent digit image for 5 iterations: mean- 0.7728 standard deviation- 0.0037094473981982845

time spent for fitting Naive Bayes Classifier on 60 percent digit image for 5 iterations: mean- -0.3576666831970215 standard deviation- 0.008066148764367216

accuracy for Naive Bayes Classifier on 80 percent digit image for 5 iterations: mean- 0.7752000000000001 standard deviation- 0.0021354156504062643

time spent for fitting Naive Bayes Classifier on 80 percent digit image for 5 iterations: mean- -0.47588386535644533 standard deviation- 0.006821521587210647

accuracy for Naive Bayes Classifier on 100 percent digit image for 5 iterations: mean- 0.775 standard deviation- 0.0

time spent for fitting Naive Bayes Classifier on 100 percent digit image for 5 iterations: mean- -0.5948931217193604 standard deviation- 0.0032327194345388214

accuracy for Naive Bayes Classifier on 20 percent face image for 5 iterations: mean- 0.7786666666666667 standard deviation- 0.05771962885843563

time spent for fitting Naive Bayes Classifier on 20 percent face image for 5 iterations: mean- -0.05969395637512207 standard deviation- 0.006345355363426483

accuracy for Naive Bayes Classifier on 40 percent face image for 5 iterations: mean- 0.8559999999999999 standard deviation- 0.01717879830230017

time spent for fitting Naive Bayes Classifier on 40 percent face image for 5 iterations: mean- -0.11939973831176758 standard deviation- 0.006048087771141409

accuracy for Naive Bayes Classifier on 60 percent face image for 5 iterations: mean- 0.8800000000000001 standard deviation- 0.017888543819998326

time spent for fitting Naive Bayes Classifier on 60 percent face image for 5 iterations: mean- -0.17108054161071778 standard deviation-

0.002528243131786376

accuracy for Naive Bayes Classifier on 80 percent face image for 5 iterations: mean- 0.8786666666666667 standard deviation- 0.009797958971132715

time spent for fitting Naive Bayes Classifier on 80 percent face image for 5 iterations: mean- -0.23717041015625 standard deviation- 0.011916856246851085

accuracy for Naive Bayes Classifier on 100 percent face image for 5 iterations: mean- 0.8800000000000001 standard deviation- 1.1102230246251565e-16

time spent for fitting Naive Bayes Classifier on 100 percent face image for 5 iterations: mean- -0.2901020050048828 standard deviation- 0.0095535017074825

Epoch 1/5

32/32 [==============================] - 0s 2ms/step - loss: 1.2922 - accuracy: 0.6000

Epoch 2/5

32/32 [==============================] - 0s 2ms/step - loss: 0.4292 - accuracy: 0.8860

Epoch 3/5

32/32 [==============================] - 0s 2ms/step - loss: 0.2706 - accuracy: 0.9400

Epoch 4/5

32/32 [==============================] - 0s 2ms/step - loss: 0.1884 - accuracy: 0.9580

Epoch 5/5

32/32 [==============================] - 0s 2ms/step - loss: 0.1457 - accuracy: 0.9690

32/32 [==============================] - 0s 982us/step - loss: 0.5558 - accuracy: 0.8400

accuracy for Neural Network with Tensorflow on 20 percent digit image for 5 epochs: accuracy- 0.8399999737739563 loss- 0.555781364440918

Epoch 1/5

63/63 [==============================] - 0s 1ms/step - loss: 0.9652 - accuracy: 0.7130

Epoch 2/5

63/63 [==============================] - 0s 1ms/step - loss: 0.3280 - accuracy: 0.9105

Epoch 3/5

63/63 [==============================] - 0s 1ms/step - loss: 0.2170 - accuracy: 0.9485

Epoch 4/5

63/63 [==============================] - 0s 1ms/step - loss: 0.1562 - accuracy: 0.9600

Epoch 5/5

63/63 [==============================] - 0s 1ms/step - loss: 0.1106 - accuracy: 0.9795

32/32 [==============================] - 0s 1ms/step - loss: 0.4066 - accuracy: 0.8690

accuracy for Neural Network with Tensorflow on 40 percent digit image for 5 epochs: accuracy- 0.8690000176429749 loss- 0.4066341817378998

Epoch 1/5

94/94 [==============================] - 0s 1ms/step - loss: 0.7687 - accuracy: 0.7693

Epoch 2/5

94/94 [==============================] - 0s 1ms/step - loss: 0.2946 - accuracy: 0.9157

Epoch 3/5

94/94 [==============================] - 0s 1ms/step - loss: 0.2129 - accuracy: 0.9417

Epoch 4/5

94/94 [==============================] - 0s 1ms/step - loss: 0.1536 - accuracy: 0.9597

Epoch 5/5

94/94 [==============================] - 0s 1ms/step - loss: 0.1127 - accuracy: 0.9737

32/32 [==============================] - 0s 895us/step - loss: 0.4082 - accuracy: 0.8750

accuracy for Neural Network with Tensorflow on 60 percent digit image for 5 epochs: accuracy- 0.875 loss- 0.4082314670085907

Epoch 1/5

125/125 [==============================] - 0s 1ms/step - loss: 0.6373 - accuracy: 0.8200

Epoch 2/5

125/125 [==============================] - 0s 1ms/step - loss: 0.2566 - accuracy: 0.9285

Epoch 3/5

125/125 [==============================] - 0s 1ms/step - loss: 0.1763 - accuracy: 0.9540

Epoch 4/5

125/125 [==============================] - 0s 1ms/step - loss: 0.1269 - accuracy: 0.9690

Epoch 5/5

125/125 [==============================] - 0s 1ms/step - loss: 0.0926 - accuracy: 0.9795

32/32 [==============================] - 0s 1ms/step - loss: 0.3943 - accuracy: 0.8760

accuracy for Neural Network with Tensorflow on 80 percent digit image for 5 epochs: accuracy- 0.8759999871253967 loss- 0.3943262994289398

Epoch 1/5

157/157 [==============================] - 0s 1ms/step - loss: 0.5911 - accuracy: 0.8196

Epoch 2/5

157/157 [==============================] - 0s 1ms/step - loss: 0.2580 - accuracy: 0.9248

Epoch 3/5

157/157 [==============================] - 0s 1ms/step - loss: 0.1657 - accuracy: 0.9568

Epoch 4/5

157/157 [==============================] - 0s 1ms/step - loss: 0.1169 - accuracy: 0.9698

Epoch 5/5

157/157 [==============================] - 0s 1ms/step - loss: 0.0888 - accuracy: 0.9766

32/32 [==============================] - 0s 1ms/step - loss: 0.3342 - accuracy: 0.9050

accuracy for Neural Network with Tensorflow on 100 percent digit image for 5 epochs: accuracy- 0.9049999713897705 loss- 0.33420413732528687

Epoch 1/5

3/3 [==============================] - 1s 47ms/step - loss: 0.9593 - accuracy: 0.4667

Epoch 2/5

3/3 [==============================] - 0s 39ms/step - loss: 0.6847 - accuracy: 0.4667

Epoch 3/5

3/3 [==============================] - 0s 40ms/step - loss: 0.6560 - accuracy: 0.7000

Epoch 4/5

3/3 [==============================] - 0s 49ms/step - loss: 0.5898 - accuracy: 0.6667

Epoch 5/5

3/3 [==============================] - 0s 39ms/step - loss: 0.4480 - accuracy: 0.9444

5/5 [==============================] - 0s 17ms/step - loss: 0.4130 - accuracy: 0.8933

accuracy for Convolutional Neural Network with Tensorflow on 20 percent face image for 5 epochs: accuracy- 0.8933333158493042 loss- 0.4129582643508911

Epoch 1/5

6/6 [==============================] - 1s 42ms/step - loss: 0.6635 - accuracy: 0.6167

Epoch 2/5

6/6 [==============================] - 0s 43ms/step - loss: 0.2820 - accuracy: 0.9333

Epoch 3/5

6/6 [==============================] - 0s 39ms/step - loss: 0.1482 - accuracy: 0.9500

Epoch 4/5

6/6 [==============================] - 0s 39ms/step - loss: 0.1018 - accuracy: 0.9556

Epoch 5/5

12/12 [==============================] - 0s 39ms/step - loss: 0.3259 - accuracy: 0.9333

Epoch 3/5

12/12 [==============================] - 0s 40ms/step - loss: 0.1269 - accuracy: 0.9583

Epoch 4/5

12/12 [==============================] - 1s 42ms/step - loss: 0.0603 - accuracy: 0.9806

Epoch 5/5

12/12 [==============================] - 1s 44ms/step - loss: 0.0246 - accuracy: 0.9972

5/5 [==============================] - 0s 13ms/step - loss: 0.0584 - accuracy: 0.9733

accuracy for Convolutional Neural Network with Tensorflow on 80 percent face image for 5 epochs: accuracy- 0.9733333587646484 loss- 0.058382369577884674

Epoch 1/5

15/15 [==============================] - 1s 40ms/step - loss: 0.5174 - accuracy: 0.7517

Epoch 2/5

15/15 [==============================] - 1s 40ms/step - loss: 0.1486 - accuracy: 0.9401

Epoch 3/5

15/15 [==============================] - 1s 44ms/step - loss: 0.0647 - accuracy: 0.9734

Epoch 4/5

15/15 [==============================] - 1s 53ms/step - loss: 0.0274 - accuracy: 0.9911

Epoch 5/5

15/15 [==============================] - 1s 52ms/step - loss: 0.0065 - accuracy: 1.0000

5/5 [==============================] - 0s 14ms/step - loss: 0.0254 - accuracy: 0.9933

accuracy for Convolutional Neural Network with Tensorflow on 100 percent face image for 5 epochs: accuracy- 0.9933333396911621 loss- 0.0253569595515728

Performances:

Since all our prediction models reach 70% to 90% of accuracy, we ignored the corssvalidation. For each model, we generated random samples with random sequences for 20%,40%,60%,80%, and 100% of the two types of data. I trained each model with 5 iterations and calculated the time spent during training and the mean/stand deviation of accuracy after testing. For perceptron on digit data, we can see a slight increase(from 0.6786 to 0.77) in the accuracy for each increase in sample size. The time also increased from 0.701s to 3.67s. I reached 77% mean accuracy. For perceptron on face data we can see a slight increase (from 0.73 to 0.82) in the accuracy for each increase in sample size. The time increased from 0.035 to 0.166. I was able to reach 82% mean accuracy eventually. For naïve bayes classifier on digit data, we can see a slight increase(from 0.7624 to 0.775) in the accuracy. The time increased from 0.125s to 0.595s per iteration. I was able to reach 77.5% of mean accuracy of 5 iterations. For naïve bayes on face data, the mean accuracy increased from 0.778 to 0.880. The time spend during training increased from 0.059 to 0.290. For Neural Network with Tensorflow on digit data, the accuracy increased from 0.84 to 0.90 and the loss decreased from 0.55 to 0.33. For CNN with face data, the accuracy increased from 0.97 to 0.99 and loss decreased from 0.41 to 0.025.

In conclusion, the neural network and CNN has the highest accuracy for all the data. While the CNN take longer to train. Both the model will reach 77% accuracy eventually but naïve bayes clearly has higher accuracy when the sample size is smaller. Also, naïve bayes classifier has faster training time especially when sample size is big. For face data, the naïve bayes has higher accuracy overall but its training speed seems a bit slower than perceptron. Eventually, the Neural Network from library definitely out performs all the other models even when CNN runs a bit slow due to their extremely high accuracy and efficient data transformation. The naïve

bayes seems performs slightly better than perceptron.