Assignment 4: Intro to Artificial Intelligence(report)

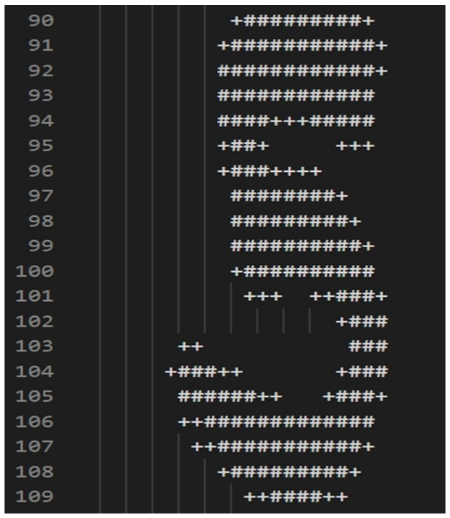
Yuting Chen Xudong Jiang

August 13 2021

1 Data

1.1 Pre-processing

First of all, the size of two data sets and each image is obtained, then, each file is transformed into a matrix with size of [i, j, k], where 'i' refers to the number of samples, 'j' to the row number and 'k' to the column number. With this data format, we can easily extract the feature map using the sum pooling method of each image. Last, "#" and "+" in images (Fig. 1) are replaced by "1", empty values are replaced by "0".



```
     +#########+
    +###########+
    ############+
    ############
    ####+++#####
    +##+     +++
    +###++++
     ########+
     #########+
     ##########+
     +##########
      +++  ++###+
             +###
 ++           ###
+###++       +###
 ######++   +###+
 ++#############
  ++###########+
    +#########+
      ++####++
```

Figure 1: Fig. 1 One digit image sample

1.2 Feature extraction

As one digit image have small image size, every pixel of one image is regarded as a feature. However, face image has relatively large size, which will reduce the training speed and model accuracy(especially in NaiveBayes), so, the sum pooling method is also used for face images in the implementation of three algorithms. Comparison between single pixel feature and sum pooling is made.

2 Methods

Face image detection corresponds to binary classification, and digit detection corresponds to multi-class classification.

2.1 Perceptron

The basic idea of perceptron is to find a hyper-plane(Eq. (1)), which will divide the input samples into two categories : positive (f >0, ŷ=1) and negative (f <0, ŷ= -1).

$$f = w * \varphi + b \tag{1}$$

In the training process, w and b are firstly initialized with random values and 1 respectively. Then, if the output ŷ differs from the true label y, w and b will be updated (Eq. (3) and (4)) by their differential (Eq. (5) and (6)) to the loss function (Eq.(2)):

$$loss = -\Sigma_i y_i (w\varphi_i + b) \tag{2}$$

$$w = w - \Delta w \tag{3}$$

$$b = b - \Delta b \tag{4}$$

$$\Delta w = -y_i \varphi_i \tag{5}$$

$$\Delta b = -y_i \tag{6}$$

Updating will stop if every y_i is the same $\hat{y_i}$.
Thee final w and b will be used to predict test samples and calculate model accuracy.
Perceptron's linear hyper plane can only classify binary categories. In the implementation, perceptron method is redesigned by using max function, which introduces non-linearity, to determine the label, thus, digit multi-class classification problem is successfully resolved. Besides, we use pixel values (feature) as φ.

2.2 Naive Bayes

Naive Bayes model is based on Bayes formula.

$$P(label = i \mid feature = j) = \frac{P(feature = j \mid label = i) * P(label = i)}{P(feature = j)} \tag{7}$$

$$P(label = i) = \frac{numberoflabel = iSamples}{numberOfAllSamples} \tag{8}$$

$$P = (feature = j \mid label = i) = \frac{numberOfFeature = jInLabel = iSamples}{numberOfLabel = iSamples} \tag{9}$$

$$P(feature) = constant \tag{10}$$

In the training process Eq.(7) is calculated by Eq.(8) and (9). Test data's label probability is predicted by calculate $P(label = i \mid testDataFeature)$ (Eq.(7)), and the final label is determined by the max label probability.
It must be stated that when feature(j) and certain label(i) do not appear at the same time in the training data, $P(feature = j \mid label = i) = 0$, in order to avoid such situation, Laplace correction must be applied. In the code, $P(feature = j \mid label = i)$ is set to 0.001 when it's value smaller than 0.001.

2.3 Neural Network

The designed neural network contains one input layer, one hidden layer and one output layer.
From input layer to hidden layer:

$$f = w * x + b \tag{11}$$

at hidden layer:

$$activation = sigmoid(f) \tag{12}$$

from hidden layer to output:

$$\hat{y} = argmax(activation) \tag{13}$$

w and b is optimized by their differential to loss function (back propagation), loss function is defined by cross-entropy.

$$loss = -\frac{1}{m}\Sigma_i y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i) \tag{14}$$

$$w = w - \Delta w \tag{15}$$

$$b = b - \Delta b \tag{16}$$

$$\Delta = \frac{1}{m}\Sigma_i(\hat{y}_i - y_i)x_i \tag{17}$$

$$\Delta b = \frac{1}{m}\Sigma_i(\hat{y_i} - y_i) \tag{18}$$

The optimization function will be processed for N iterations before final w and b come out.
Test data label is predicted by Eq.(13). For digit multi-class classification, in order to adapt to loss function(14), labels are transformed to one-hot format.
When compared perceptron with neural network, perceptron performs like a simple one-layer network, but it can only work with binary classification.

3 Result

After times of implementations, it is found that the best sum pooling size is 3*3 for face image.

3.1 Preceptron

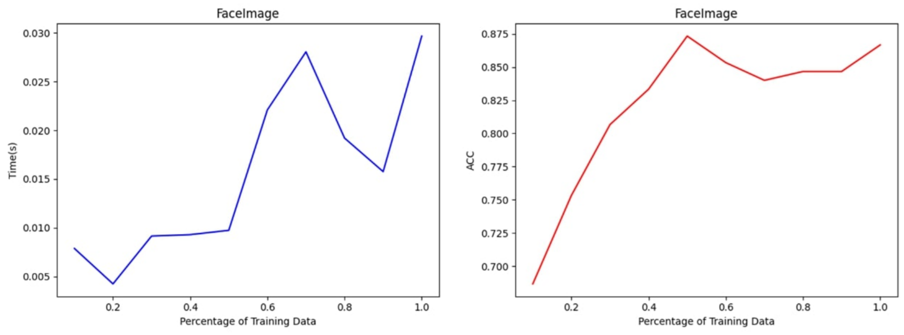


Figure 2: Fig. 2 the changing trend of time cost and accuracy with percentage of face image training data using perceptron method.

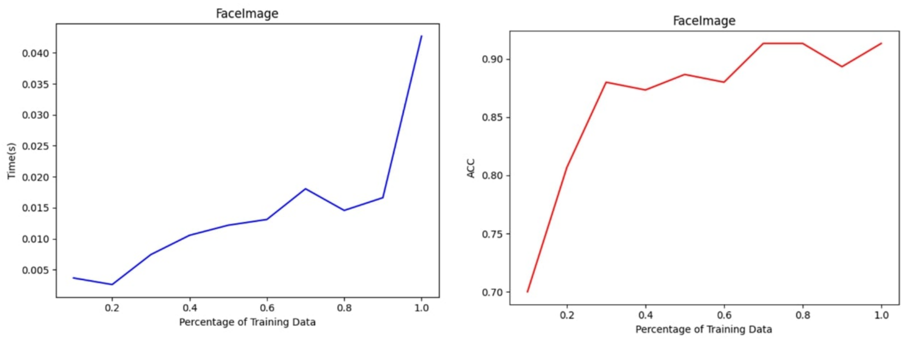


Figure 3: Fig. 3 same as Fig. 2, but with pooling size of 3*3.

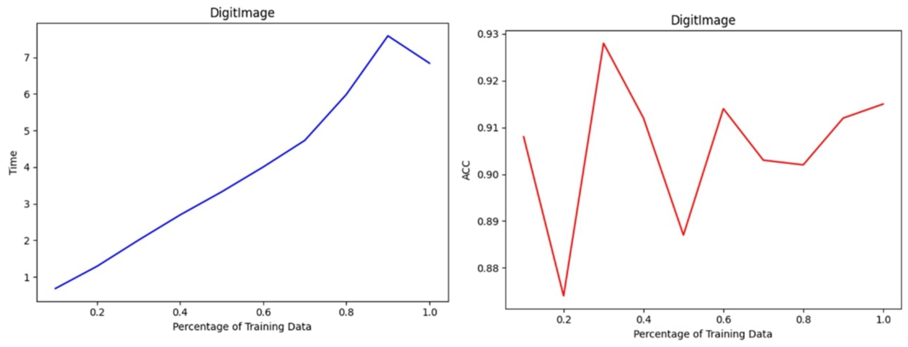


Figure 4: Fig. 4 same as Fig.2, but of digit training data.

3.2 NaiveBayes

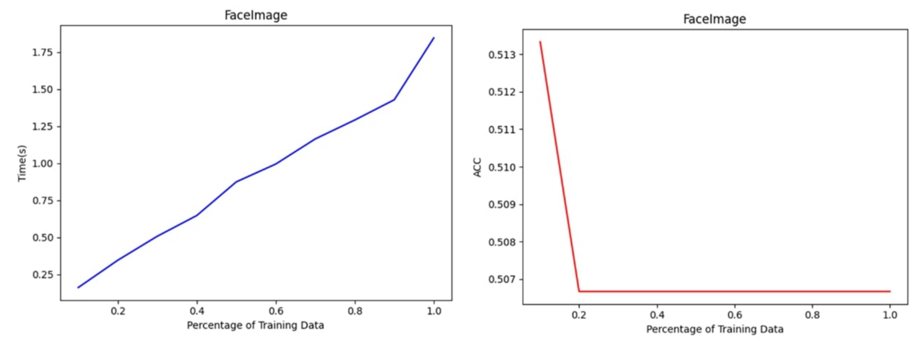


Figure 5: Fig. 5 the changing trend of time cost and accuracy with percentage of face image training data using naiveBayes method.

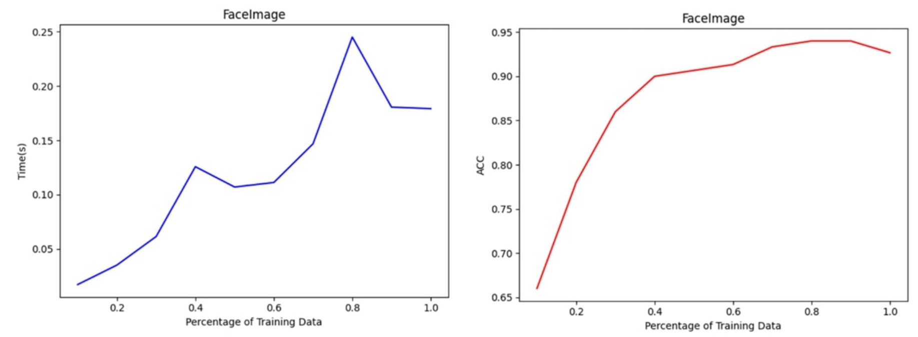


Figure 6: Fig. 6 same as Fig. 5, but with pooling size of 3*3

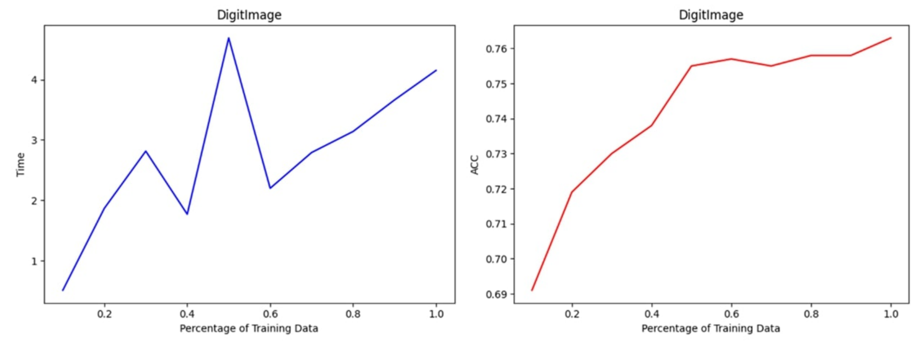


Figure 7: Fig. 4 same as fig. 5, but of digit image training data

3.3 Neural Network

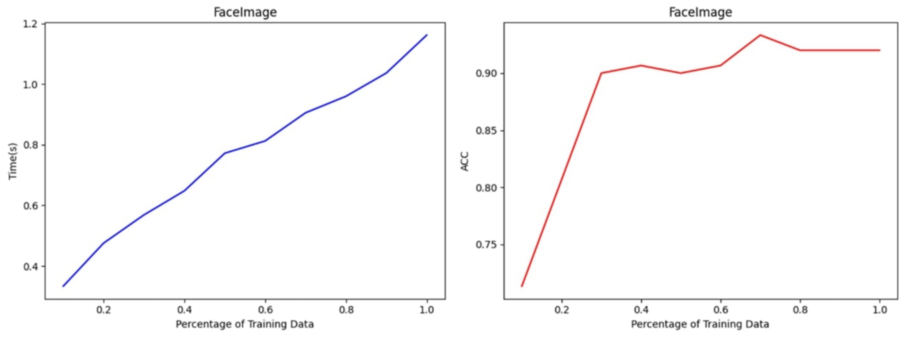


Figure 8: Fig. 8 the changing trend of time cost and accuracy with percentage of face image training data using neuralNetwork method.

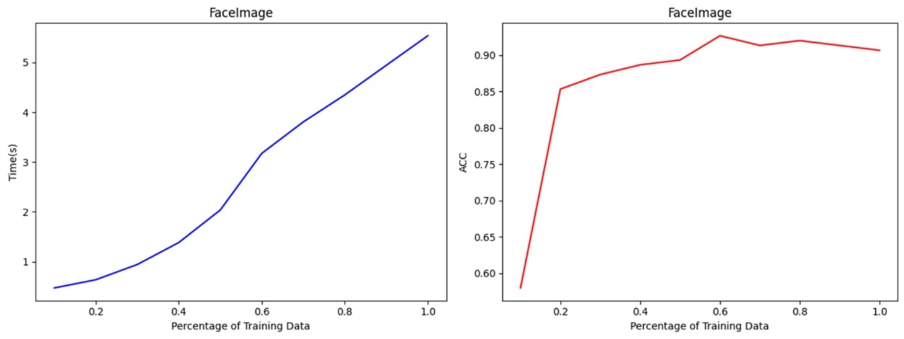


Figure 9: Fig. 9 same as fig. 8, but with pooling size of 3*3.

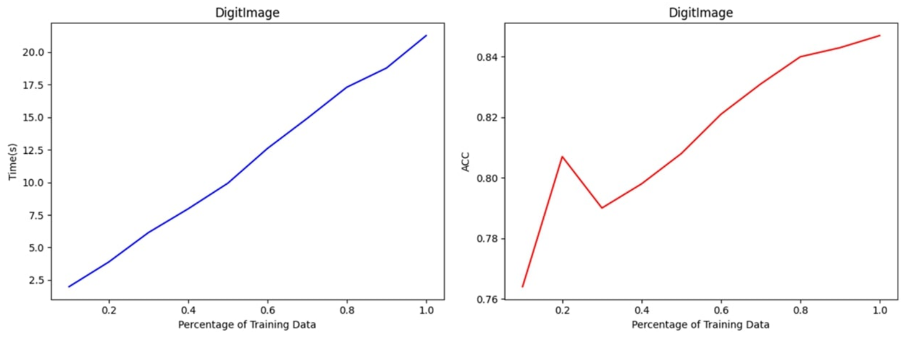


Figure 10: Fig. 10 same as fig.8, but of digit image training data.

4 Accuracy

	Face Image	Digit Image
Perceptron	0.87	0.925
Perceptron pool	0.913	
NaiveBayes	0.503	0.864
NaiveBayed pool	0.931	
NeuralNetwork	0.922	0.847
NeuralNetwork pool	0.935	

5 Conclusion

5.1 Model performance on face image

(1)It is clear that the training time of models are significantly reduced when using sum pooling method, meanwhile, higher classification accuracies appear. (2)Pooling is not used in Fig. 5, and the performance of naiveBayes appears abnormal. The reason is that there are too many features of one image, and every feature's label probability is small, which results in lots of near-zero value when calculate Eq.(7).

5.2 Model performance on digit images

(1)Without using pooling method, the training time of models are clearly high. (2)Perceptron method performs best of three models.