华南理工大学

South China University of Technology

The Experiment Report of *Machine Learning*

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Haiying zhou

Supervisor:
Mingkui Tan

Student ID:
201630666547

Grade:
Undergraduate

November 16, 2018

Face Detection Based on AdaBoost Algorithm

***Abstract*—The experiment includes two parts: Face Classification and Face Detection.**

I. Introduction

This experiment intends to make us understand Adaboost further, get familiar with the basic method of face detection, use Adaboost to solve the face detection problem, and combine the theory with the actual project.

II. Methods and Theory

A. AdaBoost Algorithm

1) Initial sample weights:

$$w_{(1,i)} = \frac{1}{N}(i = 1, ..., N)$$

2) Fit the classifier $h_m(x) \in 1, 1$with sample weights $w_{(1,i)}$on the training data
3) Compute error rate $\epsilon_m = \sum_{i=1}^{N} w_{(m,i)} \prod(y_i) \neq h_m(x_i))$
4) if $\epsilon_m > 0.5$, then set $\alpha_m = 0$ and break
5) Compute learner confidence $\alpha_m = \frac{1}{2} \log \frac{1-\epsilon_m}{\epsilon}$
6) Set $w_{(m+1,i)} \leftarrow w_{(m,i)} * e^{[}-y_i \alpha_m h_m(x_i)]$, and renormalize it so that $\sum_i w_i = 1$
7) return $F(x) = sign[\sum_{m=1}^{M} \alpha_m h_m(x)]$

III. Experiments

A. Dataset

This experiment provides 1000 pictures, of which 500 are human face RGB images, stored in datasets/original/face; the other 500 are non-face RGB images, stored in datasets/original/nonface.

B. Implementation

1) Face Classification :

i) Load data set data. The images are supposed to converted into grayscale images with size of 24 * 24, the number and the proportion of the positive and negative samples is not limited, the data set label is not limited.

```
def load_img():
for i in range(0, 500):
with Image.open("./datasets/original/face/
    face_"+"{:0>3d}".format(i)+".jpg") as
    image:
image = image.convert('L')
image = image.resize((24, 24))
imgs.append(np.array(image))
img_labels.append(1)
with Image.open("./datasets/original/
    nonface/nonface_" + "{:0>3d}".format(i
    ) + ".jpg") as image:
image = image.convert('L')
image = image.resize((24, 24))
imgs.append(np.array(image))
img_labels.append(-1)
```

ii) The data set is divisded into training set and calidation set, this experiment does not divide the test set.

```
def npd_feature():
for i in range(0, len(imgs)):
print(i)
features = NPDFeature(imgs[i]).extract()
img_features.append(features)
```

iii) The data set is divisded into training set and calidation set, this experiment does not divide the test set.

```
if __name__ == "__main__":
load_img()
npd_feature()
img_features = np.array(img_features)
img_labels = np.array(img_labels).reshape
    ((-1, 1))
print(img_features.shape)
X_train, X_val, y_train, y_val =
    train_test_split(img_features,
    img_labels, test_size=0.25)
```

iv) Write all AdaBoostClassifier functions based on the reserved interface in ensemble.py.The following is the guide of fit function in the AdaBoostClassifier class:

```
def fit(self, X, y):
  self.classifiers = self.n_weakers_limit *
      [0]
  self.classifier_w = self.n_weakers_limit
      * [0]
  feature_w = np.ones((X.shape[0])) / X.
      shape[0]

  for i in range(self.n_weakers_limit):
  self.classifiers[i] = self.
      week_classifier(max_depth=2)
  self.classifiers[i].fit(X, y,
      sample_weight=feature_w)

  prediction = self.classifiers[i].predict(
      X)
  prediction = prediction.reshape(
      prediction.shape[0], 1)

  e = 0
  for j in range(prediction.shape[0]):
  if (y[j][0] != prediction[j][0]):
  e += feature_w[j]

  if(e <= 0.5):
  self.classifier_w[i] = 1 / 2 * np.log((1
      - e) / e)
  else:
  self.classifier_w[i] = 0

  for k in range(feature_w.shape[0]):
  feature_w[k] = feature_w[k] * np.exp(-
      self.classifier_w[i] * y[k][0] *
      prediction[k][0])
  feature_w = feature_w / np.sum(feature_w)
  pass
```

v) Predict and verify the accuracy on the validation set using the method in AdaBoostClassifier and use classification_report () of the sklearn.metrics library function

writes predicted result to classifier_report.txt .

```
             precision recall f1-score support

 -1              0.91    0.96    0.93     129
 1               0.96    0.89    0.92     121

avg / total 0.93    0.93    0.93     250
```

2) Face Detection:

i) Run the face_detection.py file. Experience the OpenCV's built-in method of face detection using Haar Feature-based Cascade Classifiers.The result will be save as detect_result.jpg.

ii) You can provide your own images to replace the default test image.

we can see the result of face detection according to figure 1

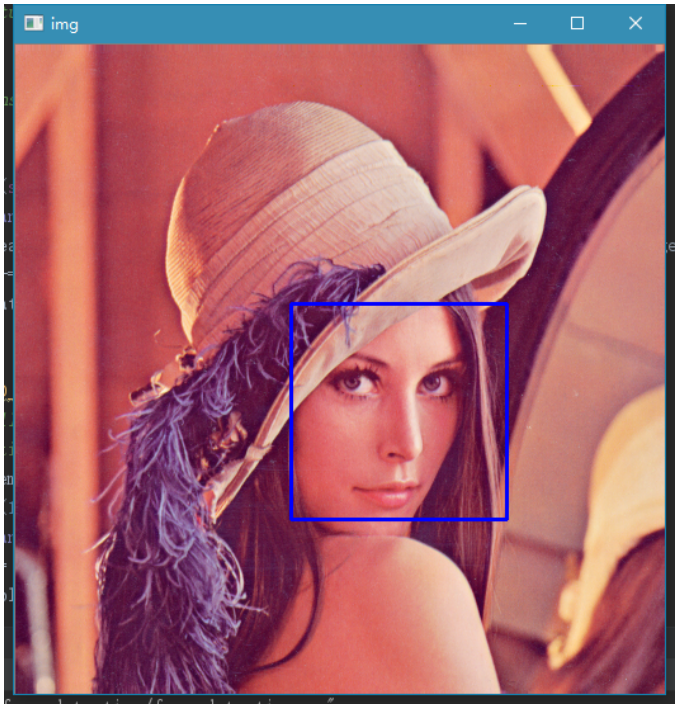

Fig. 1. This shows face detection result

C. Analysis

From experience's result, we can find that face detection based on AdaBoost algorithm Accuracy can reach about 93% with high accuracy.

IV. Conclusion

This experience is a little bit more difficult the last one.But by consulting various documents and communicating with partners, not only do I Understand Adaboost further, but also I learn to use Adaboost to solve the face detection problem,and combine the theory with the actual project.What' more, I experience the complete process of machine learning.