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:
Zhifan Xu, Haotian Huang and Kai Song

Supervisor:
Qingyao Wu

Student ID:
201530613269,201530611708,201530612729

Grade:
Undergraduate

December 21, 2017

Face Classification Based on AdaBoost Algorithm

***Abstract*—In order to understanding Adaboost further, we used this algorithm to solve the face classification problem, and combined the theory with the actual project in this experiment.**

Ⅰ. Introduction

Adaboost algorithm, as a new human face detection algorithm, has high rate of detection and can be applied to real-time monitoring. In this experiment we used this algorithm to detect if there is a human face in a picture, so that we can finish a classification question. This experiment helps us better understanding the Adaboost algorithm and classification problems. What's more, this method works well.

Ⅱ. Methods and Theory

A. Adaboost Algorithm

The Adaboost, short for Adaptive Boosting, is a algorithm that begins by fitting a classifier on the original dataset and then fits additional copies of the classified on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Suppose that we have a data set $\{(\mathrm{x}_1, y_1), ..., (x_N, y_N)\}$ where each item x_i has an associated class $y_i \in \{-1,1\}$, and a set of work classifiers $\{h_1,...,h_L\}$ each of which outputs a classification $h_j(x_i) \in \{-1,1\}$ for each item. After the $m-1$ -th iteration our boosted classifier is a linear combination of the weak classifiers of the form:

$$\mathrm{H}(x) = sign(\sum_{m=1}^{M} \alpha_m h_m(x)) \tag{1}$$

We calculate the weighted error rate of the weak classifier to be $\varepsilon_{\mathrm{m}} = \sum_{y_i \neq h_m(x_i)} \omega_i^{(m)} / \sum_{i=1}^{\mathrm{N}} \omega_i^{(m)}$, so it follows that:

$$\alpha_{\mathrm{m}} = \frac{1}{2}\log(\frac{1-\varepsilon_m}{\varepsilon_m}) \tag{2}$$

The Error rate is:

$$\varepsilon_{\mathrm{m}} = p(h_m(x_i) \neq y_i) = \sum_{i=1}^{n} \omega_m(i)\mathrm{II}(h_m(x_i) \neq y_i) \tag{3}$$

The normalization of ω is:

$$\mathrm{Z} = \sum_{i=1}^{\mathrm{N}} \omega(x) e^{-\alpha f(x)h(x)} \tag{4}$$

And finally use the weight to improve the boosted classifier C_{m-1} to $\mathrm{C}_m = C_{(m-1)} + \alpha_m h_m$.

B. The produce of Adaboost algorithm

Algorithm 2: Adaboost

Input: $D = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, **where** $\mathbf{x}_i \in X, y_i \in \{-1, 1\}$
Initialize: Sample distribution w_m
Base learner: $\mathcal{L}$

$w_1(i) = \frac{1}{n}$
for m=1,2,...,M **do**
  $h_m(x) = \mathcal{L}(D, w_m)$
  $\epsilon_m = \sum_{i=1}^{n} w_m(i)\mathbb{I}(h_m(\mathbf{x}_i) \neq y_i)$
  if $\epsilon_m > 0.5$ **then**
    break
  end
  $\alpha_m = \frac{1}{2}\log\frac{1-\epsilon_m}{\epsilon_m}$
  $w_{m+1}(i) = \frac{w_m(i)}{z_m} e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$, **where** $i = 1, 2, ..., n$ **and**
  $z_m = \sum_{i=1}^{n} w_m(i) e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$
end
Output: $H(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})$

III.EXPERIMENTS

A. Data set

This experiment provides 1000 pictures, of which 500 are human face RGB images, the other 500 is a non-face RGB images. The data set is divided into training set and validation set. And we converted all of the images into a size of 24 * 24 gray scale, labeling the human face RGB images as 1 and others as -1 respectively.

B. Implementation

In this experiment we extracted the NPD features using the NPDFeature class in *feature.py*. Then we written all *AdaboostClassifier* functions based on the reserved interface in *ensemble.py*. The following is the guide of *fit* function in the *AdaboostClassifier* class:

1) Initialize training set weights ω, each training sample is given the same weight.

2) Training a base classifier, which can be sklearn.tree library DescisionTreeClassifier, passing ω as a parameter at the training time.

3) Calculate the classification error rate ε of the base classifier on the training set.

4) Calculate the parameter α according to the classification error rate ε.

5) Update training set weights ω.

6) Repeat steps 2) - 6) above for iteration, the number of iterations is based on the number of classifiers.

Then predicting and verifying 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 *report.txt.*

C. Results and analysis:

The results in experiment is shown in the following:

TABLE 1

	precision	recall	f1-score	support
nonface	0.96	0.97	0.96	165
face	0.97	0.96	0.96	165
avg / total	0.96	0.96	0.96	330

RESULT OF FACE CLASSIFICATION

From the results we can see that the accuracy of Adaboost algorithm is high enough, meaning that our face detection present good results.

IV. CONCLUSION

we have a deeper understanding of the Adaboost algorithm and classification problems after this experiment, implementing it in python. What's more, writing the code and analyzing the results make us know that Adaboost works well in this classification problem. Finally, teamwork makes us work more efficiently.