华南理工大学

South China University of Technology

Face Classification Based on AdaBoost Algorithm

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Guoshi Lin Zhiyuan Lin and Zanjie Fang

Supervisor:
Mingkui Tan

Student ID:
201530612217 201530612323 and
201530611456

Grade:
2015

December 22, 2017

Face Classification Based on AdaBoost Algorithm

***Abstract*—AdaBoost, short for Adaptive Boosting, is a machine learning meta-algorithm well with classification problem. We will use AdaBoost Algorithm to solve problems of face classification and to have a better understanding of AdaBoosting and face classification.**

I. Introduction

ADABOOST is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers.The motivations of the experiment are understanding Adaboost further, getting famaliar with the basic method of face detecion, learning to use Adaboost to solve the face classification problem, and combine the theory with the actual project, and experiencing the complete process of machine learning.

II. Methods and Theory

The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner.

Every learning algorithm tends to suit some problem types better than others, and typically has many different parameters and configurations to adjust before it achieves optimal performance on a dataset.

The chosen methods, the related theories and the related equations are

1) Making the wrong predictive samples more important, and handling it in next round is the basic idea of AdaBoost.
2) Sample weight updating formula:
   - For right predictive sample:
   $$\omega_{m+1}(i) = \frac{\omega_m(i)}{Z_m} e^{-\alpha_m} \tag{1}$$
   - For wrong predictive sample:
   $$\omega_{m+1}(i) = \frac{\omega_m(i)}{Z_m} e^{\alpha_m} \tag{2}$$
   - $Z_m = \Sigma\omega_m(i)e^{-\alpha_m y_i h_m(x_i)}$ is normalization term, makes ω_m (i) become probability distributions.
3) So in next round,
   $$\frac{\omega_{wrong}(i)}{\omega_{right}(i)} = e^{2\alpha_m} = \frac{1-\epsilon_m}{\epsilon_m} \tag{3}$$
   and $\epsilon_m < 0.5$, wrong samples will be more important.
4) Every iteration generates a new base learner $h_m(x)$ and its importance score α_m.
   Base learner:
   $$h_m(x) : x \to -1, 1 \tag{4}$$
   Error rate:
   $$\epsilon_m = p(h_m(X_i) \neq y_i) = \sum_{i=1}^{n} \omega_m(i)\, \amalg\, (h_m(X_i) \neq y_i) \tag{5}$$
   $\epsilon_m < 0.5$ means the performance of Adaboost is weaker than random classification.
5) Make the base learner with lower ϵ_m more important:
   $$\alpha_m = \frac{1}{2} log \frac{1-\epsilon_m}{\epsilon_m} \tag{6}$$
6) Final learner:
   $$H(x) = sign(\sum_{m=1}^{M} \alpha_m h_m(x)) \tag{7}$$
   Note: $h_m(x) = sign(w^T x)$ is a nonlinear function, so the Adaboost can deal with nonlinear problem.

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 is a non-face RGB images, stored in datasets/original/nonface.

Before using this dataset, we need to divide the dataset into training set and validation set. We decide to divide 20% of the dataset for validation.

B. Implementation

The process of the experiment is:

1.Read data set data. The images are supposed to converted into a size of 24 * 24 grayscale, the number and the proportion of the positive and negative samples is not limited, the data set label is not limited.

2.Processing data set data to extract NPD features. Extract features using the NPDFeature class in feature.py. (Tip: Because the time of the pretreatment is relatively long, it can be pretreated with pickle function library dump () save the data in the cache, then may be used load ()function reads the characteristic data from cache.)

3.The data set is divisded into training set and validation set, this experiment does not divide the test set.

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

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

4.2 Training a base classifier , which can be sklearn.tree library DecisionTreeClassifier(note that the training time you need to pass the weight ω as a parameter).

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

4.4 Calculate the parameter α according to the classification error rate ϵ.

4.5 Update training set weights ω.

4.6 Repeat steps 4.2-4.6 above for iteration, the number of iterations is based on the number of classifiers.

5.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 report.txt .

C. Initialization Method

TABLE I
PARAMETERS INITIALIZATION

Number of base learner	5
Max-depth of decision tree	3

D. Training Method

The model and algorithm of AdaBoost:

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: $H(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})$

E. Experiment result

The accuracy in the validation set is 86% for classification of face or nonface images.More details are shown in the report.txt and source code.

TABLE II
EXPERIMENT RESULT

	precision	recall	fl-score	support
face	0.92	0.83	0.88	111
nonface	0.85	0.91	0.87	89
avg/total	0.88	0.88	0.88	200

IV. CONCLUSION

1.Adaboost is a effective method when solving the linear classification problem and even non-linear classification problem.By combining some weak classifiers, it can provide better performances in problems.

2.In the training process of Adaboost, it's important to set and train the base classifier, and we must keep the performance of it better than random guessing (error rate must greater than 0.5)

3.AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms.