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:
Chengneng Jin
Dewei Su
Xue Lei

Supervisor:
Mingkui Tan

Student ID:
Jin 201530611838
Su 201530612736
Lei 201530611876

Grade:
Undergraduate

December 21, 2017

Face Classification Based on AdaBoost Algorithm

***Abstract*—This experiment aims at understanding Adabosst further and getting familiar with the basic method of face detection.It uses Adaboost to solve the face classification problem,and combine the theory with the actual project.Also ,this experience the complete process of machine learning.**

I. Introduction

IN this report,face detection method based on Adaboost algorithm is presented. Face detection is a difficult task in image analysis which has each day more and more applications.The existing methods for face detection can be divided into image based methods and feature based methods.In this experiments we use a boosting algorithm to train a classifier which is capable of processing images rapidly while having high detection rates.The main idea in the building of the detector is a learning algorithm based on boosting: AdaBoost.AdaBoost is an aggressive learning algorithm which produces a strong classifier by choosing visual features in a family of simple classifiers and combining them linearly.The family of simple classifiers contains simple decison trees which is a basic machine learning method for classification problem.The result of the experiment shows a highly qualitative efficiency and we address a splendid accuracy rate.Adaboost for face classification problem has proved to be an effective method and we will discuss the future work in the later section.

II. Methods and Theory

In this section, we first introduce the ensemble learning which is the base of the boosting method.Then we introduce the boosting algorithm and adaboost to specify the ensemble learning.

A. Ensemble Learning

In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.Unlike a statistical ensemble in statistical mechanics, which is usually infinite, a machine learning ensemble consists of only a concrete finite set of alternative models, but typically allows for much more flexible structure to exist among those alternatives.

An ensemble is itself a supervised learning algorithm, because it can be trained and then used to make predictions. The trained ensemble, therefore, represents a single hypothesis. This hypothesis, however, is not necessarily contained within the hypothesis space of the models from which it is built. Thus, ensembles can be shown to have more flexibility in the functions they can represent. This flexibility can, in theory, enable them to over-fit the training data more than a single model would, but in practice, some ensemble techniques (especially bagging) tend to reduce problems related to overfitting of the training data.

Empirically, ensembles tend to yield better results when there is a significant diversity among the models.Many ensemble methods, therefore, seek to promote diversity among the models they combine.Although perhaps non-intuitive, more random algorithms (like random decision trees) can be used to produce a stronger ensemble than very deliberate algorithms (like entropy-reducing decision trees).Using a variety of strong learning algorithms, however, has been shown to be more effective than using techniques that attempt to dumb-down the models in order to promote diversity.

A variety types of ensembles methods were proposed by excellent scholars including Bayes optimal classifier,Bootstrap Aggregating Method,Boosting Method and so on.In this report we focus on only the boosting method and more specifically the Adaboost.

B. Boosting

Boosting is a machine learning ensemble meta-algorithm for primarily reducing bias, and also variance in supervised learning, and a family of machine learning algorithms which convert weak learners to strong ones.Boosting is based on the question posed by Kearns and Valiant (1988, 1989):Can a set of weak learners create a single strong learner? A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing). In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.Robert Schapire's affirmative answer in a 1990 paper to the question of Kearns and Valiant has had significant ramifications in machine learning and statistics, most notably leading to the development of boosting.

In short, boosting is a general ensemble method that creates a strong classifier from a number of weak classifiers. This is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added.

AdaBoost was the first really successful boosting algorithm developed for binary classification. It is the best starting point for understanding boosting.Modern boosting methods build on AdaBoost, most notably stochastic gradient boosting machines.However,in this experiment we use the basic form of AdaBoost to solve the face classification problem.

C. AdaBoost

AdaBoost, short for Adaptive Boosting, is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire. It can be used in conjunction with many other types of learning algorithms to improve performance. The

output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. 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.

The detail of the algorithm is in Fig. 1.

First, suppose we have a preprocessing dataset $D = \{(x_1, y_1), (x_2, y_2)...(x_n, y_n)\}$ whcich x is the feature vector and y is the label of the sample.We initialize the weight for every samples and each samples is assigned to $\frac{1}{N}$,which N is the number of the samples.

Second,the algorithm starts the iteration using m represents the iteration time.For each iteration,use the base linear learner(decision tree in this experiment) to train the data and get the learner after trained.

$$h_m(x) : \chi \to \{-1, +1\} \tag{1}$$

Then calculate the error rate of the base learner which is the sum of the weights of the wrong classified samples.

$$\epsilon_m = \sum_{i=1}^{n} w_{mi} I(h_m(x_i) \neq y_i) \tag{2}$$

Next,calculate α_m, the coefficient of the base learner which represents the importance of the base learner in the final model meanwhile the weight of the base learner in the final model.

$$\alpha_m = \frac{1}{2} log \frac{1 - \epsilon_m}{\epsilon_m} \tag{3}$$

It can be noticed that when ϵ_m is smaller than $\frac{1}{2}$,$\alpha_m \geq 0$ and the smaller ϵ_m is the greater α_m is which means the low error rate base learner will get high weight in the final model.If the $\epsilon > 0.5$ the base learner will be dropped and choose a new base learner.

Before introduce how to update the weight of the samples, we represent the z_m which is the normalization term, makes $w_m(i)$ become probability distributions

$$z_m = \sum_{n}^{i=1} w_m(i) e^{-\alpha_m y_i h_m(x_i)} \tag{4}$$

In the final of each iteration the weights of each samples are update through the original weight,the normalization term and the result of the current base learner.

$$w_{m+1}(i) = \frac{w_m(i)}{z_m} e^{-\alpha_m y_i h_m(x_i)} \tag{5}$$

After multiple iterations to get the base learner,the final model could be outputted by

$$H(x) = sign(\sum_{m=1}^{M} \alpha_m h_m(x)) \tag{6}$$

Notice that the existence of $sign$ function makes the final model a nonlinear classifier,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; the other 500 is a non-face RGB images.

B. Implementation

The experiment step is as follow

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 calidation 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 w, each training sample is given the same weight.

4.2Training a base classifier , which can be sklearn.tree library DecisionTreeClassifier (note that the training time we need to pass the weight w 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 w.

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 .

6.Organize the experiment results and complete the lab report.

As for initialization of the weight of each samples We assigned the weight to $\frac{1}{N}$ for each sample,which N is the number of the samples.

The parameter we choose in this experiment is in Table I.

The experiment results are shown in Fig. 2 and Table II.

TABLE I
MODEL PARAMETERS

Iteration Time	$maxIteration = 20$
Sample of the Data	$sampleSize = 1000$

TABLE II
EXPERIMENTS RESULTS

	precision	record	f1-score	support
-1	0.97	0.95	0.96	150
1	0.95	0.97	0.96	150
avg / total	0.96	0.96	0.96	300

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

Fig. 1. The detail of AdaBoost algorithm.

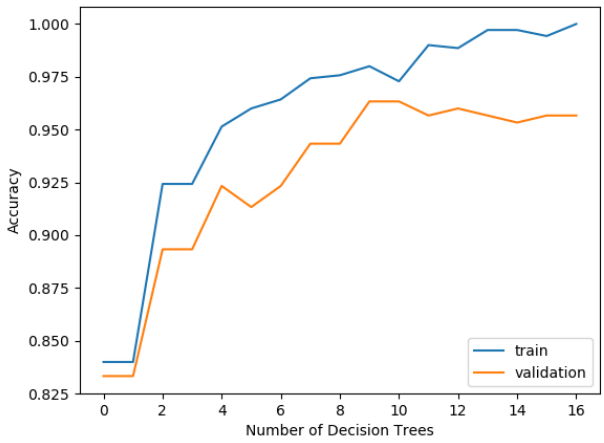

Fig. 2. The accuracy of the experimetns.

IV. CONCLUSION

According to our experiment,AdaBoost performs well which can prove it is useful for the face classification problem.Our future work may aim at the following two aspects.First,the range of the weight can be more flexible meaning that if the wrong classification occurs,the larger bias is, the larger the weight of the samples will be.How to adjust the parameter can be explored.Second,even some samples are classified correctly, the bias between them and the label is quite large which means it should be separated as wrongly classified samples.So the max margin between two classifications can be larger.Therefor the model will be more general.