The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

December 18, 2017

Author:
Yingjie Lin Yang Lin Dongcheng Mai
林英杰 林杨 麦栋铖

Supervisor:
Mingkui Tan

Student ID：
201530612286 and 201530612279 and 201536612525

Grade:
Undergraduate

Face Classification Based on AdaBoost Algorithm

Abstract—

I. Introduction

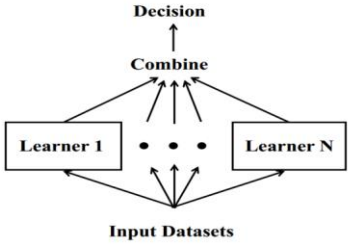
This experiment is for:

1. Understand Adaboost further
2. Get familiar with the basic method of face detection
3. Learn to use Adaboost to solve the face classification problem,and combine the theory with the actual project
4. Experience the complete process of machine learning

II. Methods and Theory

Ensemble learning: Combine numerous weak learners to a strong learner

Main methods: Boosting, Bagging



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. Experiment

A. Dataset

1. 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*.
2. The dataset is included in the example repository. Please download it and divide it into training set and validation set.

B. Experiment Step

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 , 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 you need to pass the weight as a parameter).

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

4.4 Calculate the parameter according to the classification error rate .

4.5 Update training set weights .

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 (the lab report template will be included in the example repository).

C. Implementation

Source code:

train.py

```
from PIL import Image
import pickle
import os
from feature import NPDFeature
import numpy as np
from ensemble import AdaBoostClassifier
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

# parameter
use_cache = True


def get_path(path):
    # 给路径添加前缀
    return [os.path.join(path, f) for f in
os.listdir(path) if f.endswith('.jpg')]


def conver_image(paths):
    ims = list()
    i = 0
    for path in paths:
        im = Image.open(path)
        im = im.convert(mode='L')
        im.thumbnail(size=(24, 24))
        im = np.asarray(im)
        im = NPDFeature(im)
        im = NPDFeature.extract(im)
        print(i)
        i +=1
        ims.append(im)
    return ims


if __name__ == "__main__":
    if os.stat("cache").st_size> 1024 and
use_cache  :
        # 已有缓存
        with open('cache', mode='rb') as cache:
            dataset = pickle.load(cache)
        print("load cache done")
    else:
        # 没有缓存
        faces_path =
get_path('./datasets/original/face')
        nonface_path =
get_path('./datasets/original/nonface')
        dataset = dict()
        dataset['face'] = conver_image(faces_path)
        dataset['nonface'] =
conver_image(nonface_path)
        with open('cache', mode='wb') as cache:
            pickle.dump(dataset, file=cache)
            print('save cache done')

    # conduct x_train, x_test, y_train, y_test
    y_true = [[1]] * len(dataset['face'])
    y_false = [[-1]] * len(dataset['nonface'])
    X = dataset['face'] + dataset['nonface']
    Y = y_true + y_false
    x_train, x_test, y_train, y_test =
train_test_split(X, Y)
    print("conduct train set and testing set done")

    # fit
    classsfier =
AdaBoostClassifier(tree.DecisionTreeClassifier, 5)
    classsfier.fit(x_train, y_train)

    # predict
```

```
y_predict = classsfier.predict(x_test)


# conduct report
y_truth = [i[0] for i in y_test]
y_pred = [i[0] for i in y_predict]
report = classification_report(y_truth, y_pred)
with open('report.txt', 'w') as report_file:
    report_file.write(report)
print(report)
```

IV. CONCLUSION

We read over 1000 images and extracted their NPD features. Then, the 1000 sets of feature data are disrupted, and randomly divided into a training set and test set, the ratio is 3: 1. The ratio of face and nonface in each set is about 1: 1.
We chose the decision tree algorithm as our basis learning algorithm to generate our base classifier. The number of layers in each tree is three, with a total of ten trees. After 10 iterations, we obtained an AdaBoost classifier by weighting the basis classifiers for each iteration. Test the test set, the accuracy rate can reach 95%.

Through this experiment, we felt the joy of teamwork. During the experiment, we constantly found problems and solved problems together. Finally, we were rewarded with the joy of results. All these are the wealth brought by our teamwork. Second, we have a deeper understanding of the AdaBoost method in ensemble learning. Not only understand its main learning process, but also understand the idea of integrated learning.We also have a clear understanding of the processing of image feature information, understand the method of obtaining image information. Previously amazing face recognition, through our hands-on experiment is no longer so far out of reach.