# OpenCV中BoostTree的使用

OpenCV中机器学习算法的使用大体相同，在此介绍BoostTree的使用方法，BoostTree是一个二类分类器，它可以返回置信度信息。

## 定义BoostTree分类器

只需要简单的定义如下：

CvBoost boost;

## 建立训练样本

训练样本需要是类型为CV\_32FC1(float)的矩阵，类别标志为类型CV\_32SC1(int)的矩阵。

## 训练

CvBoostParams params(CvBoost::DISCRETE, // boost\_type

100, // weak\_count

0.95, // weight\_trim\_rate

2, // max\_depth

false, //use\_surrogates

0 // priors

);

Mat var\_types(1, trainData.cols + 1, CV\_8UC1, Scalar(CV\_VAR\_ORDERED));

var\_types.at<uchar>(trainData.cols) = CV\_VAR\_CATEGORICAL;

boost.train(trainData, CV\_ROW\_SAMPLE, labels, Mat(), Mat(), var\_types, Mat(), params);

## 分类

Mat testSample(1, 2, CV\_32FC1);

testSample.at<float>(0, 0) = (float)x;

testSample.at<float>(0, 1) = (float)y;

int response = (int)boost.predict(testSample);

## 示例代码

#include "stdafx.h"

#include <opencv2\opencv.hpp>

#include <iostream>

using namespace std;

using namespace cv;

Mat trainData;

Mat labels;

CvBoost boost;

void makeSample()

{

Point neg[6] = { Point(176, 186), Point(282, 186), Point(284, 77), Point(435, 73), Point(566, 74)

, Point(62, 185) };

Point pos[6] = { Point(63, 251), Point(173, 251), Point(285, 250), Point(438, 248), Point(431, 172)

, Point(579, 169) };

trainData = Mat(12, 2, CV\_32FC1);

labels = Mat(12, 1, CV\_32SC1);

for (int i = 0; i < 6; i++)

{

trainData.at<float>(i, 0) = neg[i].x;

trainData.at<float>(i, 1) = neg[i].y;

labels.at<int>(i, 0) = 0;

}

for (int i = 6; i < 12; i++)

{

trainData.at<float>(i, 0) = pos[i - 6].x;

trainData.at<float>(i, 1) = pos[i - 6].y;

labels.at<int>(i, 0) = 1;

}

}

void train()

{

CvBoostParams params(CvBoost::DISCRETE, // boost\_type

100, // weak\_count

0.95, // weight\_trim\_rate

2, // max\_depth

false, //use\_surrogates

0 // priors

);

Mat var\_types(1, trainData.cols + 1, CV\_8UC1, Scalar(CV\_VAR\_ORDERED));

var\_types.at<uchar>(trainData.cols) = CV\_VAR\_CATEGORICAL;

boost.train(trainData, CV\_ROW\_SAMPLE, labels, Mat(), Mat(), var\_types, Mat(), params);

cout << "train finished!" << endl;

boost.save("boost.xml");

}

void test()

{

for (;;)

{

float x, y;

cout << "X = ";

cin >> x;

cout << "Y = ";

cin >> y;

Mat testSample(1, 2, CV\_32FC1);

testSample.at<float>(0, 0) = (float)x;

testSample.at<float>(0, 1) = (float)y;

int response = (int)boost.predict(testSample);

cout << "RESULT:" << response << endl;

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

int choice;

cout << "训练或载入已训练的分类器：训练（0）载入（1）：";

cin >> choice;

if (!choice)

{

makeSample();

train();

}

else

{

boost.load("boost.xml");

}

test();

return 0;

}