# 使用OpenCV的神经网络识别文字

## OpenCV中ANN的使用

关于OpenCV中ANN的使用，请参考我的另一篇文档“OpenCV中神经网络的使用”，这里只是将自己封装好的一个ocr识别的类记录下来，方便以后参考。

最终的结果：



## 代码

### 2.1 ocr.h

|  |
| --- |
| #pragma once  #include <opencv2\opencv.hpp>  #include <opencv2\ml.hpp>  using namespace std;  using namespace cv;  class ocr  {  public:  ocr(int n);  ~ocr();  void read\_train\_data(string filename);  Mat get\_feature(Mat sample);  void set\_grid\_num(int \_grid\_num);  int predict(Mat input);  void train();  void save\_classifer(string filename);  void load\_classifer(string filename);  private:  void init\_ann();  private:  int grid\_num;  Ptr<ml::TrainData> tdata;  vector<pair<Mat, int>> train\_data;  int class\_num;  Ptr<ml::ANN\_MLP> ann;  }; |

### 2.2 ocr.cpp

|  |
| --- |
| #include "stdafx.h"  #include "ocr.h"  #include <fstream>  #include <sstream>  ocr::ocr(int n)  {  grid\_num = 10;  //类别数量  class\_num = n;  }  ocr::~ocr()  {  }  void ocr::init\_ann()  {  ann = ml::ANN\_MLP::create();  Mat layerSizes(1, 3, CV\_32SC1);  layerSizes.at<int>(0) = grid\_num\*grid\_num;  layerSizes.at<int>(1) = 100;  layerSizes.at<int>(2) = class\_num;  ann->setLayerSizes(layerSizes);  ann->setActivationFunction(ml::ANN\_MLP::SIGMOID\_SYM, 1, 1);  ann->setTermCriteria(TermCriteria(TermCriteria::MAX\_ITER + TermCriteria::EPS, 300, FLT\_EPSILON));  ann->setTrainMethod(ml::ANN\_MLP::BACKPROP, 0.001);  }  Mat ocr::get\_feature(Mat sample)  {  if (sample.channels() != 1)  {  cvtColor(sample, sample, CV\_RGB2GRAY);  }  //二值化  Mat sample\_binary;  Scalar t = mean(sample);  threshold(sample, sample\_binary,t[0], 255, CV\_THRESH\_BINARY);  //归一化尺寸  resize(sample\_binary, sample\_binary, Size(grid\_num, grid\_num));  Mat out = Mat::zeros(1, grid\_num\*grid\_num, CV\_32F);  int k = 0;  for (int i = 0;i < sample\_binary.rows;i++)  {  for (int j = 0;j < sample\_binary.cols;j++)  {  out.ptr<float>(0)[k++] = float(sample\_binary.ptr<uchar>(i)[j]) / 255.0;  }  }  return out.clone();  }  void ocr::set\_grid\_num(int \_grid\_num)  {  grid\_num = \_grid\_num;  }  int ocr::predict(Mat input)  {  Mat testSample = get\_feature(input);  int res = ann->predict(testSample);  return res;  }  void ocr::train()  {  //准备训练数据：数据矩阵和标签矩阵  int sample\_num = train\_data.size();  Mat trainClasses = Mat::zeros(sample\_num,class\_num,CV\_32FC1);  Mat trainSample;  for (int i = 0;i < train\_data.size();i++)  {  Mat feature = get\_feature(train\_data[i].first);  trainClasses.ptr<float>(i)[train\_data[i].second] = 1.0f;  if (!trainSample.data)  {  trainSample = feature.clone();  continue;  }  else  {  trainSample.push\_back(feature);  }  }  tdata = ml::TrainData::create(trainSample, ml::ROW\_SAMPLE, trainClasses);  //初始化网络  init\_ann();  //训练  cout << "开始训练~" << endl;  ann->train(tdata);  cout << "训练结束!" << endl;  }  void ocr::save\_classifer(string filename)  {  ann->save(filename);  }  void ocr::load\_classifer(string filename)  {  ann = Algorithm::load<ml::ANN\_MLP>(filename);  }  void ocr::read\_train\_data(string filename)  {  fstream file(filename, fstream::in);  if (!file.is\_open())  {  cout << "打开文件失败！" << endl;  return;  }  string line;  cout << "正在读取训练文件..." << endl;  while (getline(file,line))  {  string path, str\_label;  stringstream ss;  ss.str(line);  ss >> path >> str\_label;  Mat im = imread(path);  int label = atoi(str\_label.c\_str());  if (im.data)  {  train\_data.push\_back(make\_pair(im, label));  }  }  cout << "读取训练文件完毕,共读取到"<<train\_data.size()<<"副样本!" << endl;  } |

### 2.3 main.cpp

|  |
| --- |
| #include "stdafx.h"  #include "ocr.h"  #include <iostream>  #include<opencv2\opencv.hpp>  int main()  {  ocr cls(10);  cls.set\_grid\_num(20);  std::cout << "载入分类器（0） 训练分类器（1）：" << std::endl;  int choice;  std::cin >> choice;  if (choice)  {  //训练  cls.read\_train\_data("train.txt");  cls.train();  cls.save\_classifer("ocr\_10.xml");  }  else  {  //载入  cls.load\_classifer("ocr\_10.xml");  }  //测试  cv::Mat image = cv::imread("3.bmp", 0);  int res = cls.predict(image);  printf("result: %d\n", res);  imshow("image", image);  waitKey(0);  return 0;  } |

### 2.4 训练数据及标签



