**集美大学计算机工程学院**

**数字图像处理课程设计报告**

**题 目** 基于MTCNN的实时人脸检测方法及系统

**专 业：**  信息管理与信息系统 **班 级：** 信管1611

**学生姓名：**  刘佳昇 **学 号：**  201621124025

2019年7月12日

# 需求分析

1. 系统的输入；

系统的输入为一张图像或者视频帧

1. 系统的输出；

系统的输出为一张标定有人脸的图像，或者视频帧

1. 系统实现的功能。

实现了图像的多人脸标定以及摄像头实时人脸检测

# 概要设计

## 2.1 基本要求

使用了MTCNN深度级联神经网络的多任务框架，该框架由三层网络构成，分别是P-Net、R-Net、以及O-Net。通过分别训练这三个神经网络。将一个网络的输出作为后一个网络的输入。以粗略到精细的方式预测面部的地标位置。指的一提的是，该框架通过选取loss前70%的hard example（比较难训练的样本）执行方向传播，提高了网络的训练效果。经过实验分析，该方法在面部检测中有着卓越精度，同时保持了实时性能。此外，在训练好的模型基础上，设计并开发出一个实时人脸检测系统，适用于门禁，照相以所有需要运用到人脸检测的场景。代码和模型已公开至: <https://github.com/ohMyJason/tensorflow-MTCNN.git>

### 2.1.1算法原理

#### 2.1.1.1 面部分类：

学习目标被制定为两类分类问题。 对于每个样本，我们使用交叉熵损失：

(1)

其中是网络产生的概率，表明样本是一个面。 符号 表示ground-truth(正确的数据标注，其实就是数据集中的人脸位置以及特征位置) 标签

#### 2.1.1.2 边界框回归

对于每个候选窗口，我们预测它与最近的ground-truth 之间的偏移（即边界框的左边，顶部，高度和宽度）。 学习目标被制定为回归问题，我们对每个样本使用欧几里德损失：

(2)

其中是从网络获得的回归目标， 是地面实况坐标。 有四个坐标，包括左上角，高度和宽度，因此

#### 2.1.1.3 面部地标定位

类似于边界框回归任务，facial landmark（面部地标）检测被公式化为回归问题，我们最小化欧几里德损失：

(3)

其中 是从网络获得的面部地标坐标，是第个样本的地面实况坐标。有五个面部标志，包括左眼，右眼，鼻子，左嘴角和右嘴角，因此

#### 2.1.1.4 多源训练

由于我们在每个CNN中执行不同的任务，因此在学习过程中存在不同类型的训练图像，例如面部，非面部和部分对齐的面部。 在这种情况下，不使用一般损失函数。

例如，对于背景区域的样本，我们仅计算，而另外两个损失设置为0.这可以直接使用样本类型指示符来实现。 然后整体学习目标可以表述为：

(4)

其中N是训练样本的数量，表示任务重要性。

我们在P-Net和R-Net中使用

(5)

而在O-Net中使用

以获得更准确的面部地标本地化 是样本类型指示器。 在这种情况下，采用随机梯度下降来训练这些CNN是很合适的

### 2.1.2界面设计方案

### 2.1.2.1 前端界面实现

界面由图片识别模式和摄像头实时识别模式两个部分构成，方便用户操作。如图4-2所示。界面左上侧部分是显示数据源的图片，右上部分是显示人脸检测结果。下班部分用于显示摄像头实时检测情况，可以手动选择开启或者关闭摄像头显示，方便用户操作。

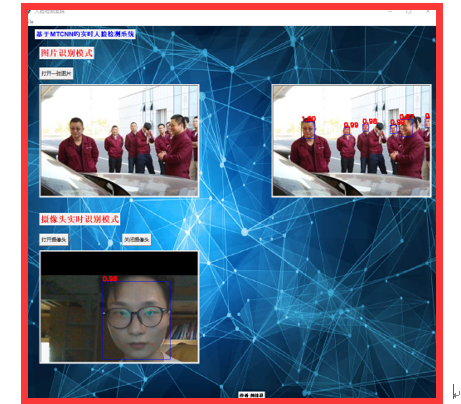


图1 人脸识别系统主界面

#### 2.1.2.2功能逻辑实现

功能逻辑是属于中间部件，需要和前端界面交互，又要调用算法进行处理，属于架构里面不可或缺的部分。首先数据源通过OpenCV进行读取，两种不同数据源，其中图片需要用户先选择，文件选择通过TKinter实现。如图4-3所示，图片支持\*.JPG、\*.PNG、\*.GIF、\*.BMP格式。摄像头支持USB、笔记本自带摄像头。

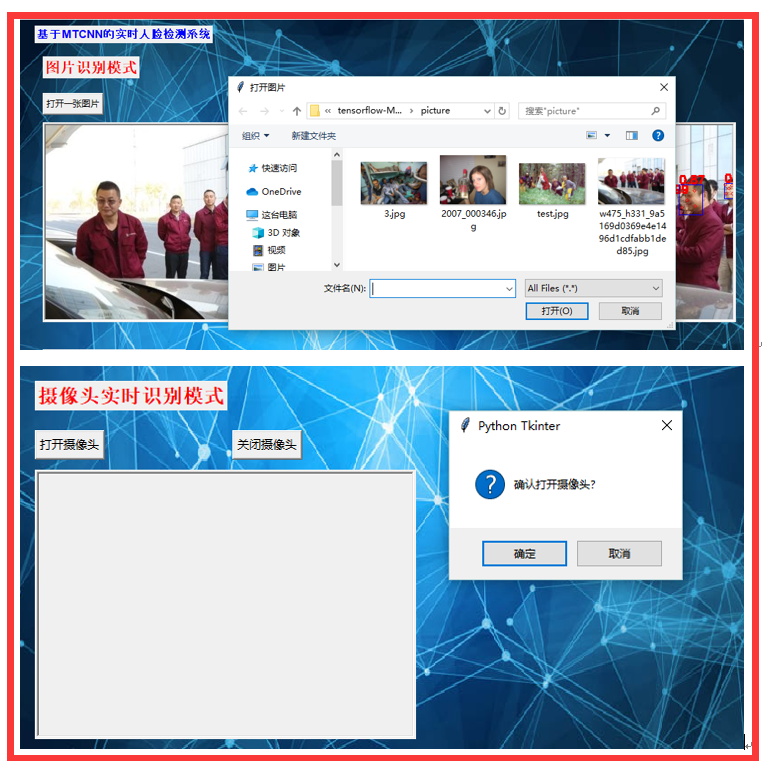


图2 两种种选择方式效果图

# 详细设计

data下放置训练所用的原始数据和划分数据，生成的tfrecord等  
detection下的fcn\_detector.py主要用于PNet的单张图片识别，detector.py用于RNet和ONet的一张图片通过PNet截取的多个人脸框的批次识别，MtcnnDetector.py为识别人脸和生成RNet，ONet输入数据  
graph里放置的是训练过程中生成的graph文件  
output里放置识别图像或视频完成后存储放置的路径  
picture里是要测试的图像放置路径  
preprocess里是预处理数据程序，BBox\_utils.py和utils.py，loader.py是一些辅助程序，gen\_12net\_data.py是生成PNet的pos，neg,part的程序，gen\_landmark\_aug.py是生成landmark数据的程序，gen\_imglist\_pnet.py是pnet的四种数据组合一起，gen\_hard\_example.py是生成rnet,onet的三种数据程序，gen\_tfrecords.py是生成tfrecord文件的程序  
train中的config是一些参数设定

model.py是模型,train.py是训练，train\_model.py针对不同网络训练  
test.py是测试代码

# 实验结果与分析

## 4.1多种方法对比

将训练好的模型和多种主流人脸检测方法进行对比，如表4-7所示，人脸检测模型MTCNN在WIDER FACE训练集上表现优异，通过三层训练网络的特征融合、损失函数合理设计，模型效果大大提升，速度达到99帧。

表 1 多种方法在WIDER FACE VAL上评估对比情况

|  |  |  |
| --- | --- | --- |
| Method | GPU | Speed |
| Cascade CNN | Titan | 100 |
| Faceness | Titan | 20 |
| DP2MFD | Titan | 0.285 |
| **ours** | **Titan** | **99** |

4. 2多种环境对比

训练好的模型在多种较为严苛的条件下如外物遮挡，水平倾角变换，垂直上倾角变换，垂直下倾角变换，表情变换，环境中增加烟雾等情况下的的人脸检测结果，记录如下：



图2脸部遮挡情况

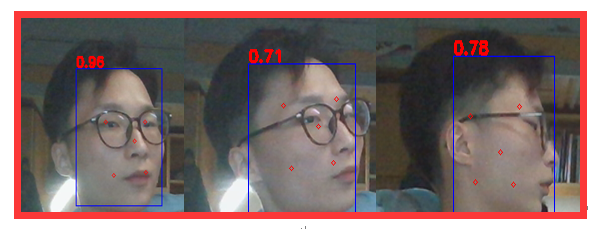


图3脸部水平角度变换分别为45度、60度、90度的识别情况



图4 脸部垂直水平上倾角分别为45度、60度、70度的识别情况

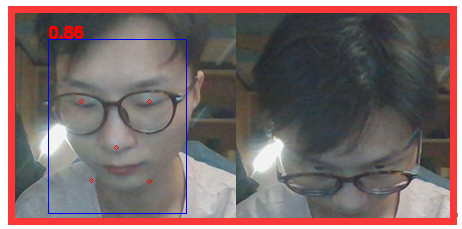
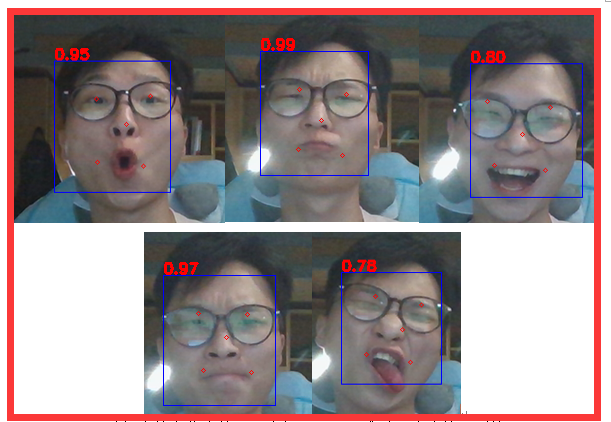


图5脸部垂直水平下倾角分别为45度，60度的识别情况

图6脸部表情变化为惊讶，生气，开心，难过，疼痛的识别情况

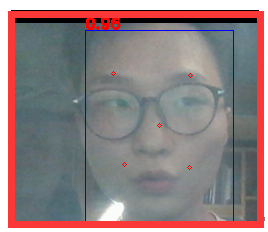


图7 识别环境中存在烟雾的识别情况

实验结果显示，在有遮挡物的情况下，左右遮挡可以正常识别，上下遮挡（如遮挡眼睛）无法正常识别。在水平倾角变换分别为45度，60度，90度的情况下均可以正常识别但是特征点显示标定错误。在垂直上倾角分别为45度，60度的情况下识别正常，上倾角提高到70度时，无法识别。在垂直下倾角为45度的时候可以正常死别，但是角度增加至60度时无法正常识别。在脸部存在各种表情的时候可以正常识别。在环境中有烟雾的情况下可以正常识别。

有上述实验结果可以得出结论，系统在检测人脸的过程中具备水平变换，表情变化，环境噪声的鲁棒性，但是在识别上下角度变化以及特征点被遮挡过多的情况下，识别率会收到影响。

## 4.3 结论

根据上述实验结果，可以看出MTCNN人脸检测算法相比于其他算法，有着高效，速度快，计算量较小的各方面优势。人脸检测模型识别精度极高。但是也有需要克服的问题：过小的人脸识别精度低、对局部特征识别效果低下。

由上述人脸识别系统可以看出系统将算法和界面融合为一体，可以实现实时的摄像头人脸检测。在多人，遮挡，低光，模糊等条件下有着比较高的匹配精度。

本文从对严苛环境下的人脸检测的需求出发，对基于MTCNN级联神经网络的人脸检测算法进行研究，训练出在各个环境下鲁棒性较高的的人脸检测模型，最后基于Tkinter将算法、界面集成在一起，开发出一款可靠的人脸检测系统。

在人脸检测算法上，采用深度级联神经网络对样本进行分别训练，将上一个网络的输出作为下一个网络的输入，实现高效且准确的人脸检测效果。实验表明这种方法有利于检测不同尺度，不同角度，存在环境噪声的人脸，表现优异。人脸的关键特征检测效果符合人脸识别的标准，实验表明这种结构适用于人脸检测任务，可以有效提取人脸特征，在实时检测中效果好，可用于多种场景下人脸检测。

最后，基于训练好的模型，针对本文需求，开发出了一个人脸检测系统，基本满足各种环境规模下的人脸检测的需求，并且可以用于其他任务。