今天我将从这四方面 来介绍我的毕业设计，主要对二三部分进行一些展开叙述。10秒

树莓派自2010年发布以来，尺寸始终未变，和我的学生证一样大，计算能力和存储能力却得到了不少提升。除了搭建一些基本的小型服务器和物联网应用外，近年来已有一些研究人员和业余爱好者尝试在其上开展一些高计算负荷的任务，也收到了一定效果。其中不乏简单的人工智能任务。自alexnet点燃深度学习的热潮以来，表情识别作为图像识别的一个重要分支，近年来也迎来了不少突破。我期望将这两个技术有机结合起来，在合适的场景部署人脸表情识别系统。45秒

接下来我讲一下系统的设计和实现。我将系统分为两部分，模型部分和识别部分。在模型部分，我选取的训练数据集是fer2013，其中一些图像样本如图所示，其规模适中，共有35877张图片，每张图片的像素仅为48\*48大小，适用于训练一个小型识别模型。右侧的流程图是一个比较常见的数据集处理过程。30秒

在网络架构方面，我主要用到了这些网络层或是函数，卷积，ReLU，maxpooling和lrn。其中lrn正是alexnet作者提出的一种归一化手段，虽然在后来的论文中被否定，但由于本文提出的模型属于浅层网络，与alexnet类似，lrn在训练中对准确率的提升有明显贡献。在进行研究的过程中，我发现很多深度学习办法往往无法讲清楚好用在哪里，但人人都在用，因为有效。50秒（可提升）

我在三种平台上训练网络，尝试做一些对比。在开题时我心里已经很清楚了，树莓派目前还干不了训练模型的重活儿。一方面我依然试图搭建出足够简单有效的网络给树莓派试试，另一方面我也尝试在服务器端训练进行对比。此处除了自己设计的网络外，我还试着使用比较全能的Inception网络来进行重训练，利用其在ImageNet上区分上千种不同类别图像的出色表现来对七种表情进行分类识别。在以往的学习中，我了解到tensorflow有checkpoint这一特性，在本文的训练中大有用处，若训练到一半时树莓派过热断电保存能够已有参数，还可以用训练一半的模型进行识别。

刚刚讲完了模型部分，在我设计时来看，识别部分的流程正如右图所示。从树莓派官方摄像头中读入视频，利用opencv中haarcascadeclassifier来切割出人脸，那么表情识别是我们刚刚谈到的模型要去做的事，最后一步的展示我希望突出一些亮点。如左图所示，一方面给出人脸在七种表情类别上的得分，以柱状图形式展示，另一方面使用比较可爱的emoji替代识别出的表情。

这个表展示了各平台关键环境的选择……。其中我把tensorflow和opencv圈起来了。为什么要用tensorflow呢？因为其对树莓派的支持是目前深度学习框架中最好的。

这里我展示的是我调用的tensorflow和opencv中的一些类和函数。有一部分我会直接调用，另一部分针对本系统我会自己定义一些函数去使用。比如网络的搭建我会把conv2d, max\_pool, relu, lrn, dropout都放到一个函数中，调用更方便，代码冗余也大大降低。

这一页是我自建网络的一个具体情况。输入一张图片后，经过两个卷积层两个全连接层和一个线性层得出最后的得分。由于fer2013中的图像已经是48\*48这样一个尺寸，故此我卷积过程中的stride设为了1，padding设为了SAME，这样卷积前后每一张feature map的尺寸是不变的，这样最大程度上保存了图像的spatial特征。在对网络调整的过程中，我会更关注内存占用和参数的数量。对于当前这一个比较好的模型，我计算结果如图：

接下来我总结了一下实验结果向大家展示，并针对一些特点进行分析。表1是三种平台下的训练情况，可以看到，服务器上12分钟就能完成的训练，笔记本电脑上8小时仅能完成一半，而树莓派上，完全无法完成，时间无从计算。这也彻底否定了树莓派独自完成深度学习任务的可能性。图二展示的是几种模型训练的结果

这张图展示的是最终的界面，中间的蓝方框圈出的是识别到的人脸，左上角7行绿字是表情类别，后方对应七个蓝色矩形看作柱状图，矩形宽度与得分成正比，界面左侧中央是替换的emoji。这个界面能够基本实现设计时的各种需求。