个人项目文档

视界-基于 YoloV5-lite 本地部署的安卓 APPV1.0

一、软件简介

该软件为一个视觉辅助出行软件，主要针对视障群体。通过对图像信息数据的数字化处理和分析，得到模型的识别结果并反馈给用户，让用户能够利用移动端软件有效进行避障等十大功能（开发中）。目前该软件已有三个通道

软件设计：视障群体、家属入口及志愿者服务，每一位用户（视障群体）在首次注册时会获得自己专属识别码，家属只需注册绑定即可实时查看用户的出行情况；同时，志愿者会在视障群体需要帮助时给予帮助。软件可基于用户的使用习惯进行个性化定制。同时，软件增设网上商城、服务包售卖以及基础功能体验，用户通过软件即可进行试用，通过手机软件进行识别播报

二、工程笔记

模型部署方法

基于模型转换算法将训练好的模型软件转化为Tensorflow Lite可部署在安卓设备的轻量化模型，基于Android Studio的交互编程实现各个页面的跳转

通过编写mMoudle方法成功实现在安卓端加载轻量模型并进行计算：

```
// 读取文件，加载模型
try {
    mModule = LiteModuleLoader.load(assetFilePath(getApplicationContext(), assetName: "yolov5s.torchscript.ptl"));
    BufferedReader br = new BufferedReader(new InputStreamReader(getAssets().open( fileName: "classes.txt")));
    String line;
    List<String> classes = new ArrayList<>();//classes 放置classes
    while ((line = br.readLine()) != null){
        classes.add(line);
    }
    //
    PrePostProcessor.mClasses = new String[classes.size()];
    classes.toArray(PrePostProcessor.mClasses);
} catch (IOException e) {
    Log.e( tag: "Object Detection", msg: "Error reading assets",e);
    finish();
}
```

安卓端满意度回馈算法

在部署在安卓的同时，基于传统的满意度新增了响应时间、检测精度等回馈指标，响应时间达到10ms之内，ACC指标达到97.63%，并且在处理因遮挡而导致精确度降低的情况下对两个物体的交占比进行比较，从而筛选正确的信息。

```
boolean done = false;
for (int i=0; i<boxes.size() && !done; i++){
    if (active[i]){
        Result boxA = boxes.get(i);
        selected.add(boxA);
        if (selected.size() >= limit) break;

        for(int j = i+1; j<boxes.size();j++){
            if(active[j]){
                Result boxB = boxes.get(j);
                if (IOU(boxA.rect, boxB.rect)>threshold){
                    active[j] = false;
                    numActive -= 1;
                    if  (numActive <= 0){
                        done = true;
                        break;
                    }
                }
            }
        }
    }
}
```

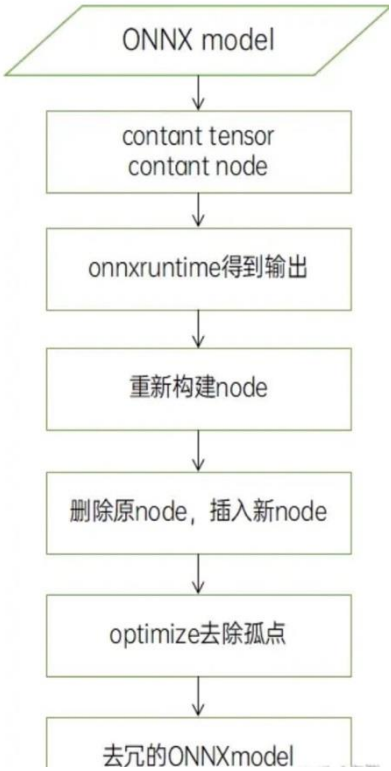
Yolo以及Transformer模型较重，适合于计算机进行运行，我设计了新的轻量化模型算法，通过万能模型onnx为媒介，将将训练的大模型转换为pytorchlite、tflite、NCNN轻量化模型，通过onnx部署，分别可以部署在不同的设备上，在力求减少所需算力的同时尽量保证了精度不下降太多，轻量化模型使得大部分设备能够轻松运行，下图为转换代码和核心思路：

```
from onnx_tf.backend import prepare
import onnx

TF_PATH = "tf_model" # where the representation of tensorflow model will be s
ONNX_PATH = "mobilenet_v2.onnx" # path to my existing ONNX model
onnx_model = onnx.load(ONNX_PATH)  # load onnx model
tf_rep = prepare(onnx_model)  # creating TensorflowRep object
tf_rep.export_graph(TF_PATH)
```

```
input_names = ["input"]
output_names = ["output"]
torch.onnx.export(model,
                  dummy_input,
                  "mobilenet_v2.onnx",
                  verbose=False,
                  input_names=input_names,
                  output_names=output_names)

model = onnx.load("mobilenet_v2.onnx")
ort_session = ort.InferenceSession('mobilenet_v2.onnx')
onnx_outputs = ort_session.run(None, {'input': test_arr})
```



三、使用说明

登陆界面

点击软件图标， 进入软件登陆界面，如上图所示， 显示图标和，输入产品邀请码进入软件， 或者点击购买直接使用。

付款界面

上图为付款页面， 用户付款之后联系客服即可直接使用软件

主界面

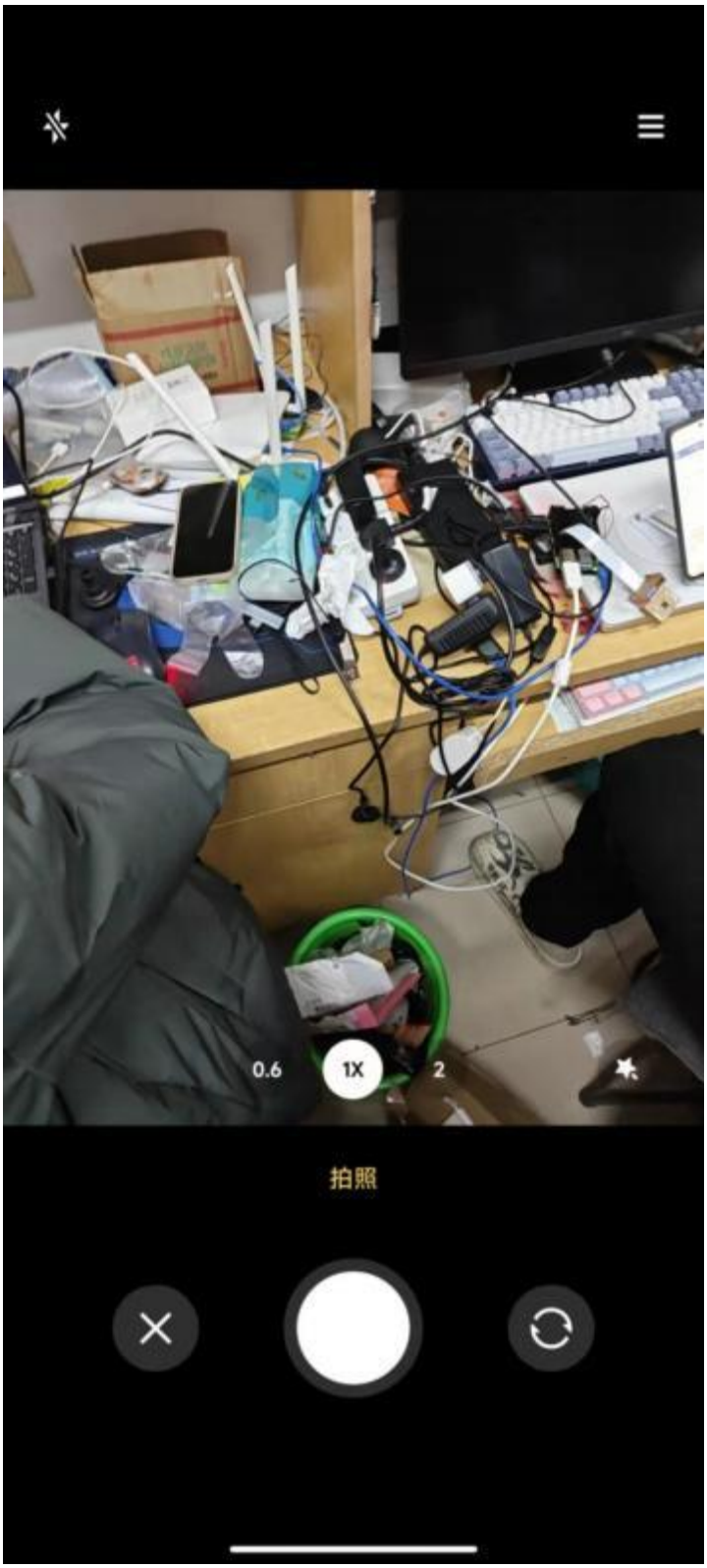
进入软件主界面，如上图所示，显示软件标题和相关功能，主要包含三个部打开相册，相机拍照和实时检测

打开相册

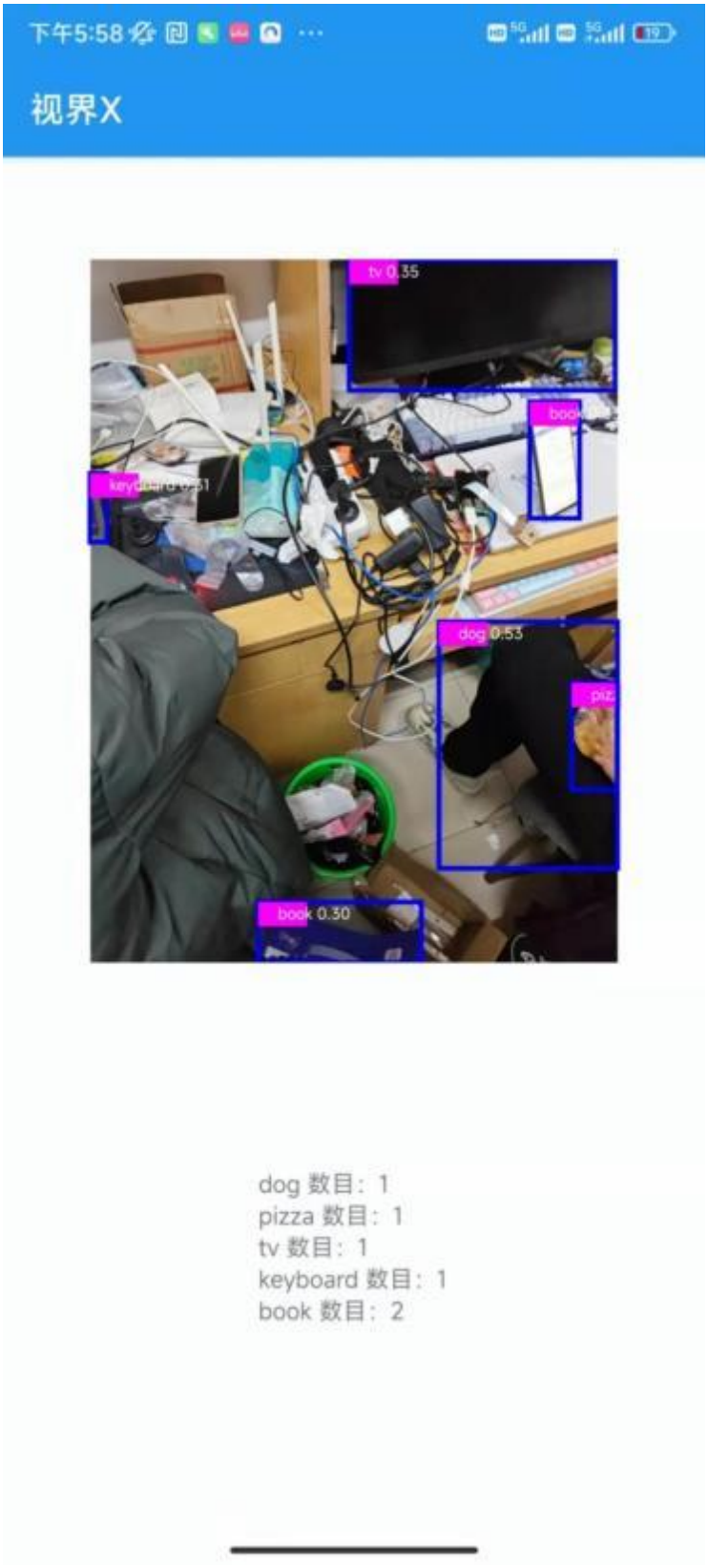
点击“打开相册”，如上图所示， 打开手机相册， 从中选择要检测的图片，可进行目标检测。

上传图片之后软件会自动进行目标检测， 把检测到的物品数量与种类显示在界面下方。

相机拍照



点击“相机拍照”，可调用手机自带摄像头进行照片拍摄，在完成照片拍摄后进行目标检测。


下午5:58
视界X
tv 0.35
dog 0.53
book 0.30
dog 数目：1
pizza 数目：1
tv 数目：1
keyboard 数目：1
book 数目：2

实时检测

打开“实时检测”，详情如上图所示, 打开用户手机的摄像头， 对拍到的目标进行实时检测， 并在界面下方进行实时播报检测到的目标种类于数量