

基于 MNN 和 MTCNN 的人脸检测

目前安卓端调用 mnn 的 demo 是将应用层（如 loadmodel, create session）逻辑写入了 java 层，对于不熟悉 java 的同学开发不是很方便，本文提供一个版本将应用层写到 c++ 的 demo 演示。这样做的好处是核心算法逻辑和 mnn 实现都可以在 c++ 层完成，java 层只需要传图和显示结果即可，而且只需要一个 jni 接口交互，避免了在 java 和 c++ 对算法混合开发的过程。

调用的结构是这样的：

1. 最上层是 java 层，功能为传图给 c++ 层，显示图像，不涉及 mnn 的逻辑
2. 中间层：jni 层，java 和 c++ 的接口层，只负责传递数据
3. 底层：c++ 层，负责图像的计算功能，例如本实例中的人脸检测功能

如何使用？

1. 使用 android studio 编译安装文件夹 android 下的工程
 - a) 在以下路径下载 opencv 的 android 最新版本
<https://opencv.org/releases/>,
 - b) 将下载的 opencv 压缩包解压到与 cMakeLists.txt 相同的路径下，位置如下图。（注：也可以修改 cMakeLists.txt 中（第 10 行）的 opencv 路径）

名称	修改日期
.externalNativeBuild	2019/8/29 9:13
build	2019/8/29 9:13
includes	2019/8/29 9:13
libs	2019/8/29 9:13
OpenCV-android-sdk	2019/8/29 9:13
src	2019/8/29 9:13
.gitignore	2019/6/24 10:08
app.iml	2019/8/28 15:07
build.gradle	2019/7/7 14:12
CMakeLists.txt	2019/8/28 15:18
proguard-rules.pro	2019/6/24 10:08

Figure 1 opecv 解压位置

2. 将 models 中的 3 个模型文件拷贝至手机上，手机路径位于“内部存储\1”

- a) 如果在手机端找不到“内部存储\1 文件夹”，请断点调试 native-lib.cpp 中第 44 行查看完整路径

C++代码层次划分：

- b) 本文聚焦于 c++ 实现部分，阅读此实现的起点为 native-lib.cpp 中的 “deal(imgData, testfilePath);”，此接口为 c++ 实现的入口，它的输入为当前的图像帧，模型存储的路径，功能为实时检测人脸框，并将人脸框画在 imgData 上，返回 imgData；
- c) deal 函数内部调用 imgProcess 的 detFace 接口，实现人脸检测功能
- d) detFace 内部调用 MTCNN 类，MTCNN 类对 mtcnn 人脸检测算法进行封装，实现于 mtcnn.cpp 中
- e) MTCNN 在底层调用了 Inference_engine 类实现前传功能
- f) Inference_engine 对 mnn 库的外部接口进行封装，统一为 load 和 infer 两个接口，并且在这一层封装了 mnn 的逻辑功能