

DNNThreadFace3.0 使用指南

目录

- 简单介绍.....2
- 了解 DNNThreadFace3.02
 - 后台系统.....2
 - 本地实时识别系统.....2
 - 远程通讯系统.....3
 - 修改后台服务器技巧.....4
 - 前台人脸采集系统.....5
 - 采集系统的工作步骤如下.....5
 - 第一步，连接服务器.....5
 - 第二步，打开摄像头（可选择前置后置）6
 - 第三步，录制视频.....8
 - 第四步，上传.....9
 - 第五步，管理样本库.....11
 - 第六步，训练人脸模型.....13
 - 前台实时人脸识别系统.....15
- DNNThreadFace3.0 两个前台都是简体中文，如何制作多语言的前台系统.....17
- 后台服务器的部署问题.....18

简单介绍

DNNTThreadFace 集人脸采集、样本管理，模型训练，人脸识别于一体，以傻瓜方式工作的二次开发项目。它不是现产品，但它却是最易用的二次开发工作模板。

了解 DNNTThreadFace3.0

DNNTThreadFace1.0 使用异步流水线的模型，输入视频解码工作于线程中，完成后异步把数据扔给 GPU 线程进行人脸检测，检测结果发现人脸再扔给分类器线程，最后输出识别结果。

DNNTThreadFace2.0 采用了 DNNTThread（多 GPU 跑核心业务负载），做到了其中一个 GPU 在忙碌中，就把识别任务发给另一个 GPU 识别，整个系统与 1.0 一样，都是全异步方式工作，工作效率可以根据硬件数量提升，可以说数倍提升，但是瓶颈仍然受限于 CPU->GPU 的数据流水线。

DNNTThreadFace3.0 对于 CPU->GPU 的整个工作流程，做出了统一优化，简单来说就是站在 2.0 基础上，最优化重构了全部流程。建模更傻瓜，样本上传更快，样本采集更快，识别几乎是立即的，无任何延迟感，3.0 对于二次开发可以说极度友好，可以快速将人脸识别体系应用到项目中。2.0 是技术突破，3.0 是应用突破。

后台系统

后台系统分为本地实时识别系统和远程传输系统，两者相结合。

后台系统不是一蹴而就开发，而是一步一脚印走出来，先有 DNNTThread 原型，再有 FaceQueue 的呈现 demo，然后再做优化，调整，待系统跑起来，再用网络接口数据传输。

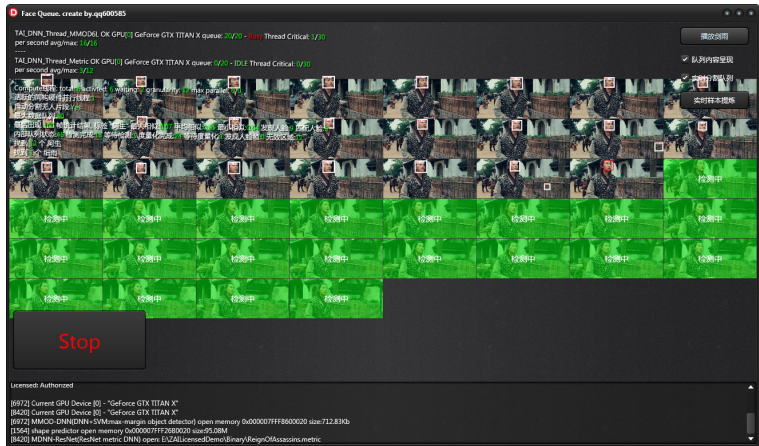
IDC 托管的服务器、云服务器，上行是不限制的，只限制下行，后台不会主动发送推流数据，因此后台希望完全可以放在低成本的云端运行，当然，后台服务器会需要一张 gpu 卡。

本地实时识别系统

实时识别系统由 zAI_VideoFaceQueue.pas 库通过 DNNTThread 内核架起来的自动化识别队列，简单来说是这样的。我们在输入实时视频时，数据是一个队列，也许有好几百帧，也就是好几百张图片光栅，这时候，这些按顺序放在内存中的队列会被多 GPU 并发的执行识别处理流程，GPU 并发技术就是 DNNTThread。

在 GPU 处理多帧的识别结果时，zAI_VideoFaceQueue.pas 库引入了统计学模型，做到了活体识别，以及精确匹配。

在 ZAI 的 demo 中，有一个叫 FaceQueue_DNNThread.exe 的 demo。它将 DNNThread 的内部工作流程基本上都呈现出来了。要了解本地实时识别的处理机制，看这个 demo 就可以了



远程通讯系统

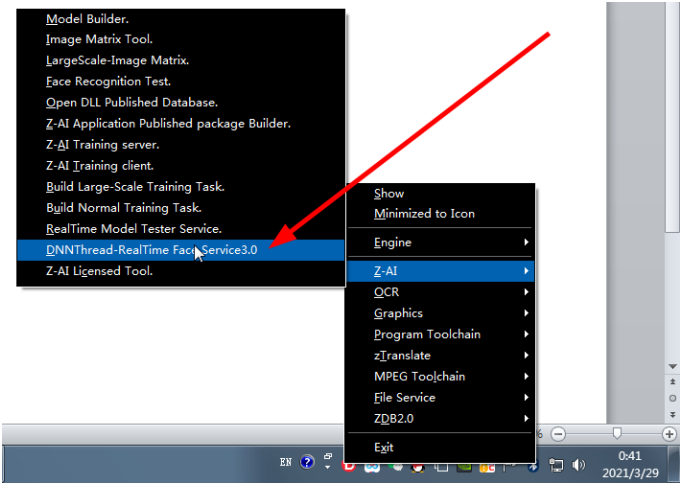
远程通讯系统是架构在 p2pVM 基础上的通讯机制，专门处理复杂会话。简单来说，通讯系统分为，数据采集通讯系统，实时识别通讯系统两个子系统。

数据采集就是让前端把视频实时的发送过来，并且告诉我们，这个视频是谁的人脸。
实时识别使用了双工通讯模型，一个通道负责接收远程的实时推流，一个通道负责发送实时识别的统计学结果。

远程通讯系统因为关系到 p2pVM，架构上的细节，可以通过下列开源项目详细了解

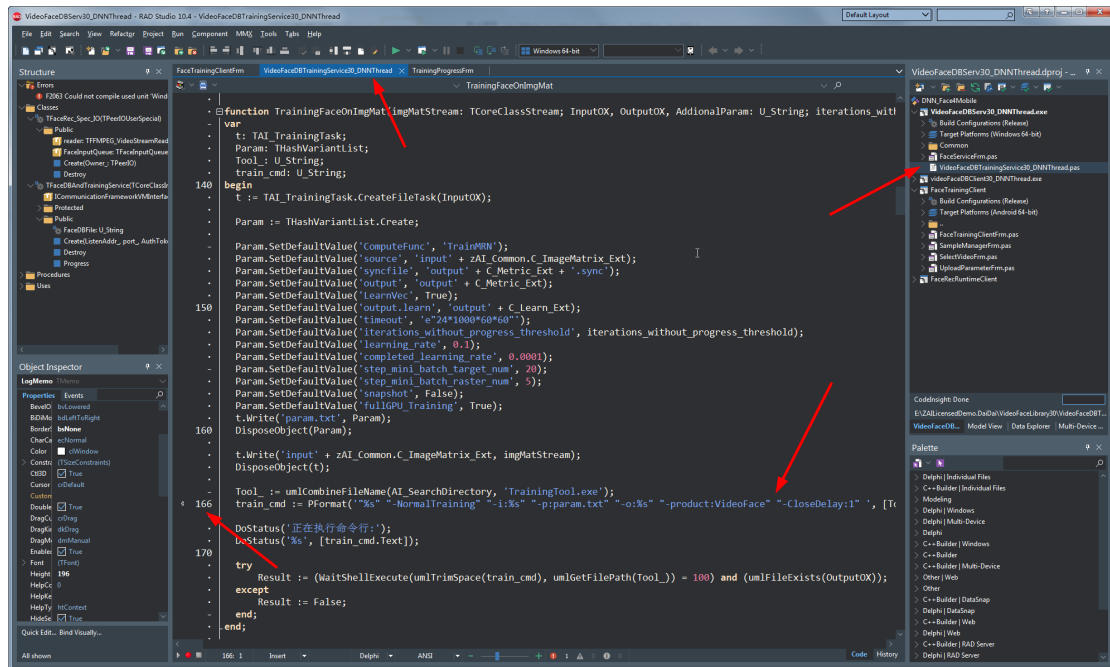
- <https://github.com/PassByYou888/ZServer4D>
- <https://hub.fastgit.org/PassByYou888/ZServer4D>

在 Z-AI1.34 的主工具链中，启动 DNNThreadFace.30 服务器可通过以下方式

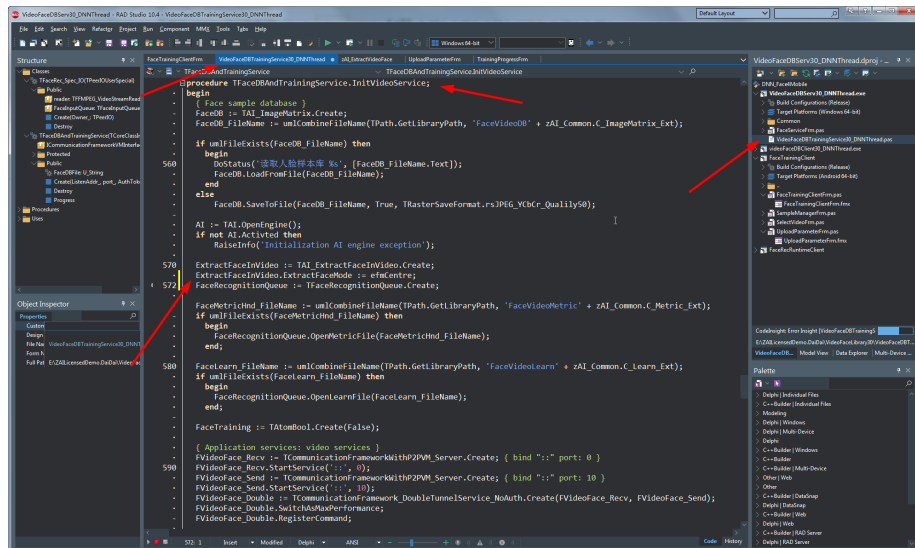


修改后台服务器技巧

- 如果不想让服务器训练时，TrainingTool 弹出来，加个“-Minimized”参数就可以
- 例如
- "-product:VideoFace" "-CloseDelay:1"
- 修改成
- "-product:VideoFace" "-Minimized" "-CloseDelay:1"



- 如果希望服务器的样本库分辨率再高一点，在这里插入两行
- ExtractFaceInVideo.ExtractPictureWidth:=600;
- ExtractFaceInVideo.ExtractPictureHeight:=600;



前台人脸采集系统

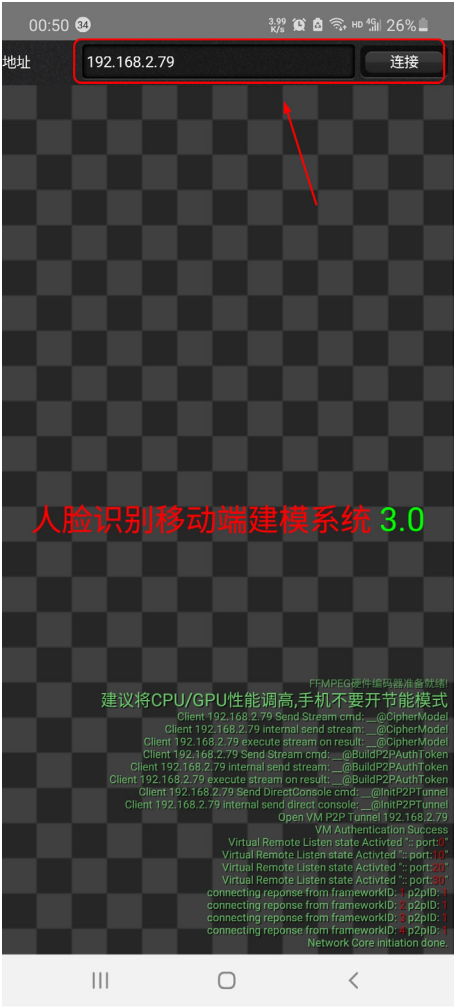
前台人脸系统在 Android10 上面构建，代码标准，没有搞隐晦高端技术，没有离谱命名。
前台系统可以自己改造成苹果 IOS 的东西，标准代码，整个系统就是网络通讯+摄像头采集，不存在技术难题。

采集系统的工作步骤如下

数数 12345，往下走就行了

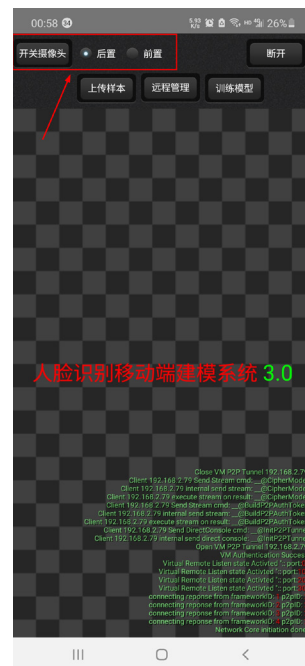
第一步，连接服务器

- 打开 app 时，什么都没有，往里面输入服务器 DNNTThread Face 服务器的地址。
- 这个地址在成功握手以后会自动保存，下次不必反复输入。
- 如果连接不成功会反复重试，一直到成功
- 手机采集端运行打开后不会锁屏
- 不要使用节能模式

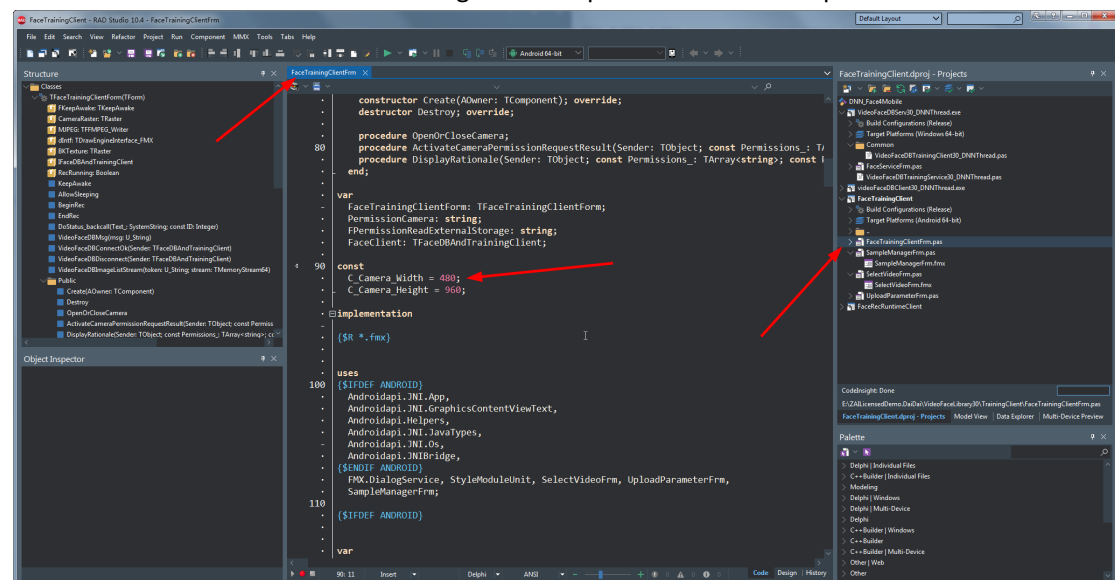


第二步，打开摄像头（可选择前置后置）

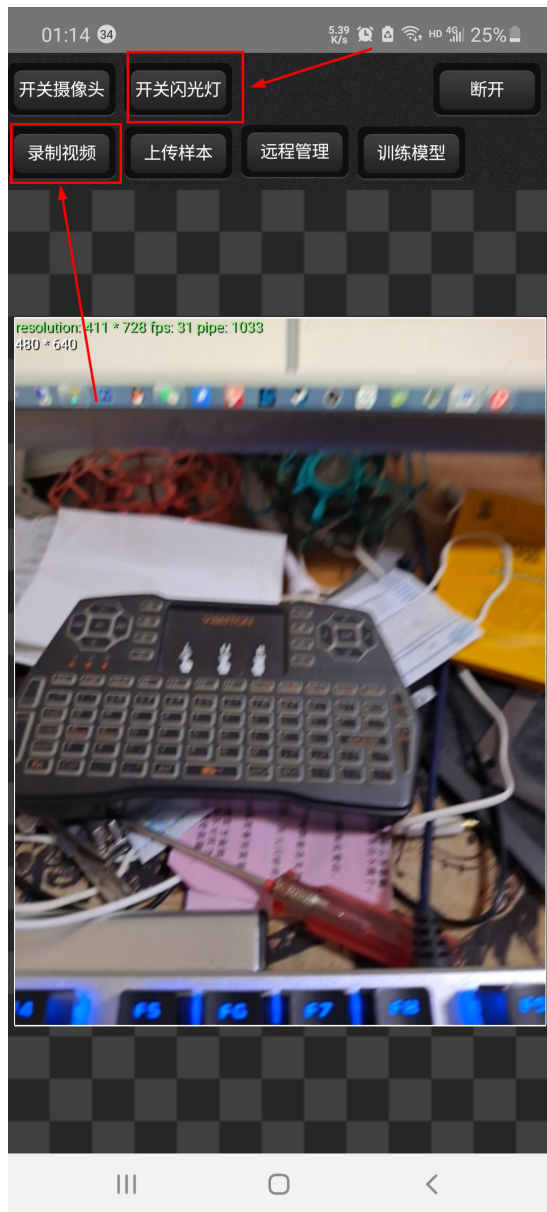
- 登录完一堆绿字，1秒以后就会看到控制面板，这里可以选择手机的前置还是后置，后置摄像头可以开闪光灯加亮，前置分辨率比较低，只能拍自己。
- 如果首次安装 APP 安卓系统会提示打开摄像头控制权限
- 采集和编码视频比较消耗资源，手机应该设置成高性能模式，省电模式可能出现卡顿
- 手机如果 wifi/4G/5G 断线，系统会自动回到第一步让你重新连接



- 不论摄像头是前置还是后置，系统都会选择最接近 C_Camera_定义分辨率。
- 采集分辨率的定义在 FaceTrainingClientFrm.pas 里面。默认 480p，可根据需要修改。

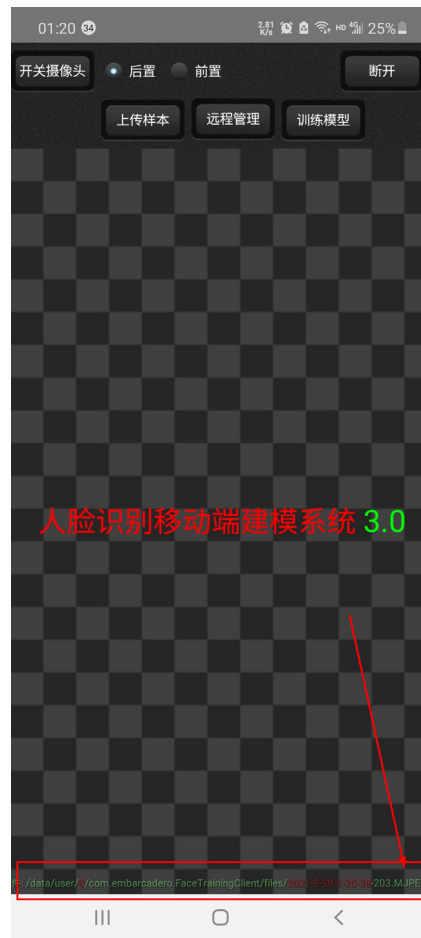


- 只有打开后置摄像头才会看见闪光灯和录制视频按钮
- 如果是打开状态，再次点开关摄像头，会关闭摄像头，重新回到第二步
- 如果 wifi/4G/5G 网络断开了会自动关闭摄像头，并且回到第一步



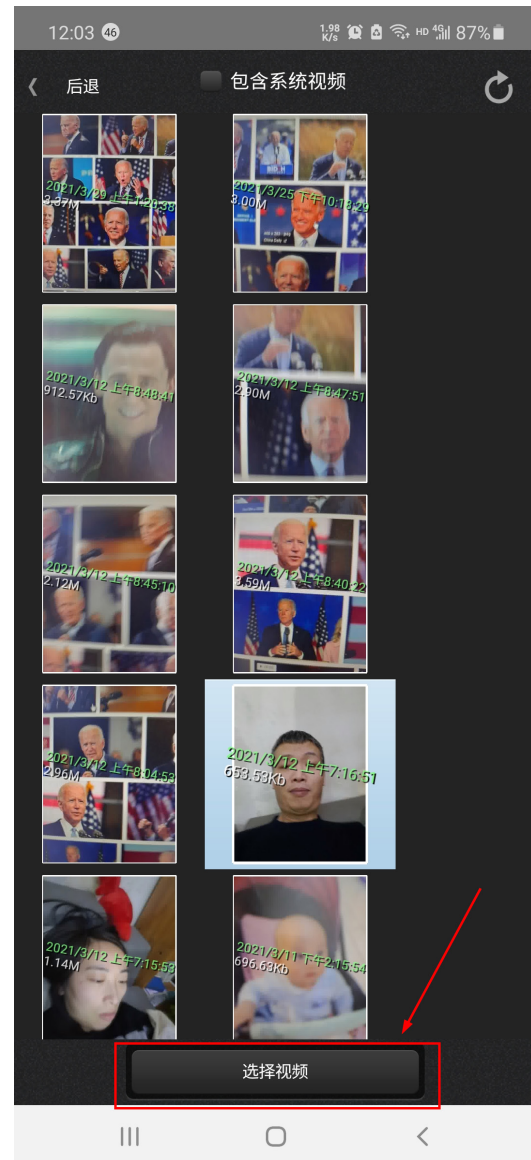
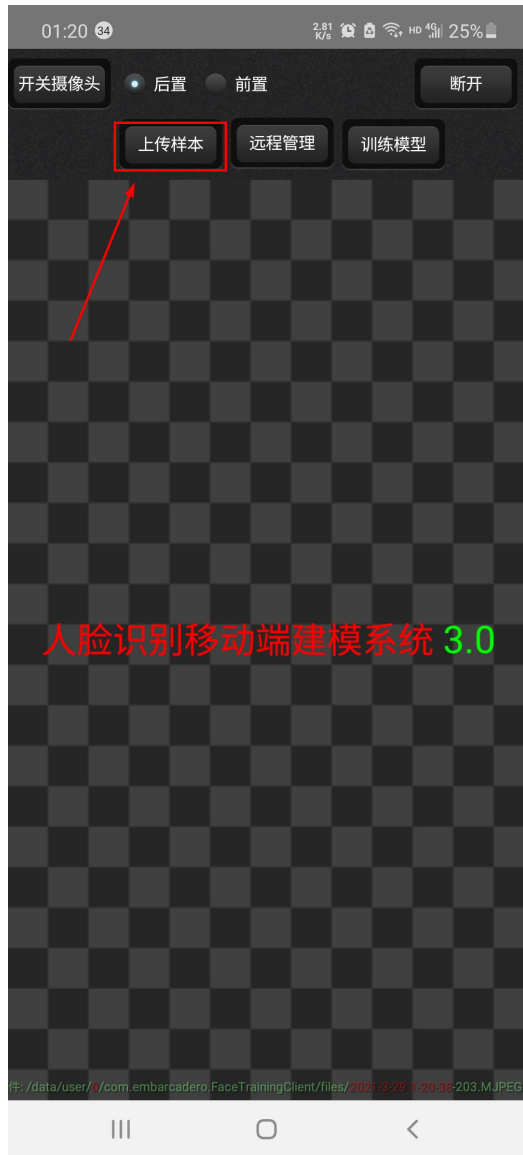
第三步，录制视频

- 点录制视频，这时候按钮上的字会变成停止录制，表示系统正在录制视频
- 在录制过程中，如果点开关摄像头，会自动停止录制
- wifi/4G/5G 网络断开时也会自动停止录制
- 停止录制时在下面会有个提示，视频已经写入到某个 MJPG 码流文件中
- 录制人脸应该多摆日常表情，录制时间不宜过长，凭感觉就行。因为某些高端手机每秒 60fps 采集效率，这样每秒大概产生 30M 的视频资源，视频太大上传费劲，手机空间也消耗快
- 如果视频中出现了很多人，系统会采集视频中央的人，就是一次采集一个人，在做二次开发时就可以画个中央框让用户把摄像头对准



第四步，上传

- 上传样本前注意要先停止录制
- 点上传样本会弹出近期的视频录制列表，这些视频列表会使用手机的多核同屏播放，一般来说，手机配置越好视频越流畅
- 选择视频再点选择视频按钮

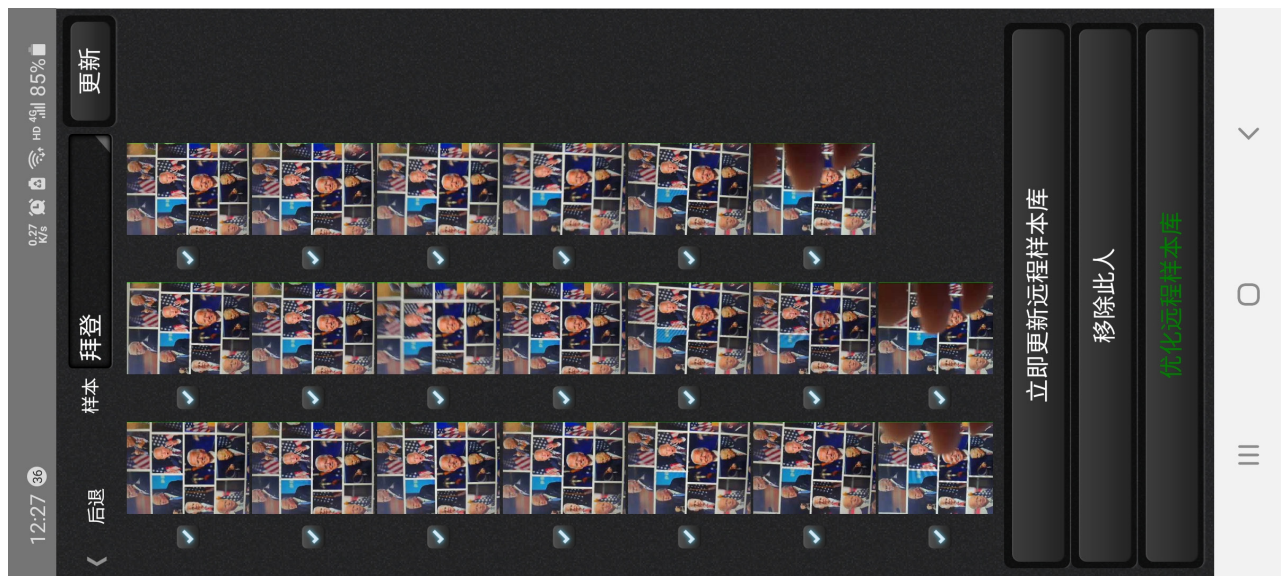


- 人脸标签必须要给出
- 限制视频帧是指让服务器不必解析过大的视频，200 帧，大概 5 秒左右，如果我们录制了 5 分钟，那也只会采集前面的 200 帧
- 限制样本数表示从本次上传的最大人脸，这个参数需要说明一下，如果视频中找到 200 张人脸，这里的 20 是从 200 张人脸中平均提炼出 20 张，不是检测到 20 张就停止
- 点开始上传，大概 2-10 秒内服务器会给出提示，从本次视频中提取了几张人脸，当服务器给出提示以后，点远程样本管理可在管理器中立即看到本次上传的样本
- 在上传样本中，只会等待几秒，建议不要关闭或则切换 app，容易断线
- 如果上传过程断线，不会把服务器的样本库弄坏，只需要重新选择视频再上传一次即可

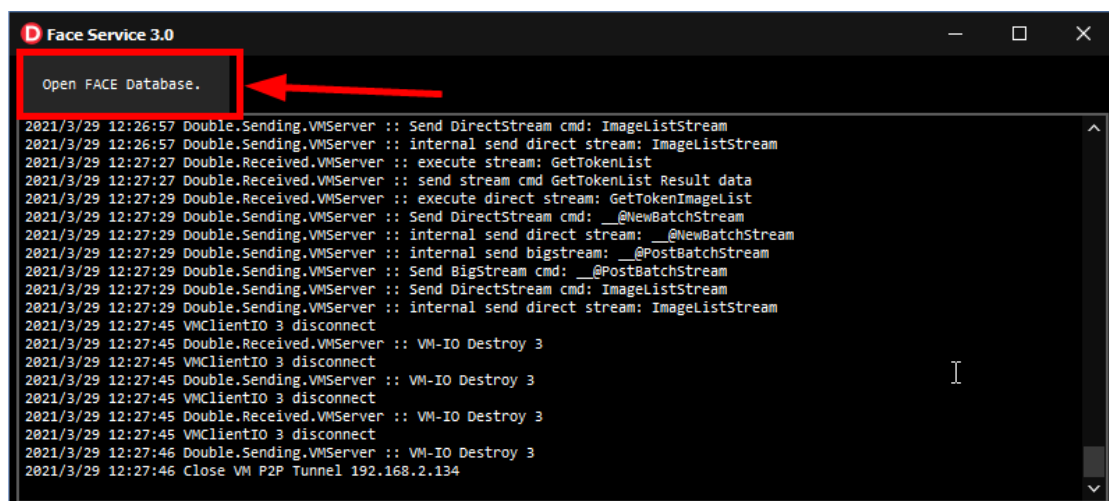


第五步，管理样本库

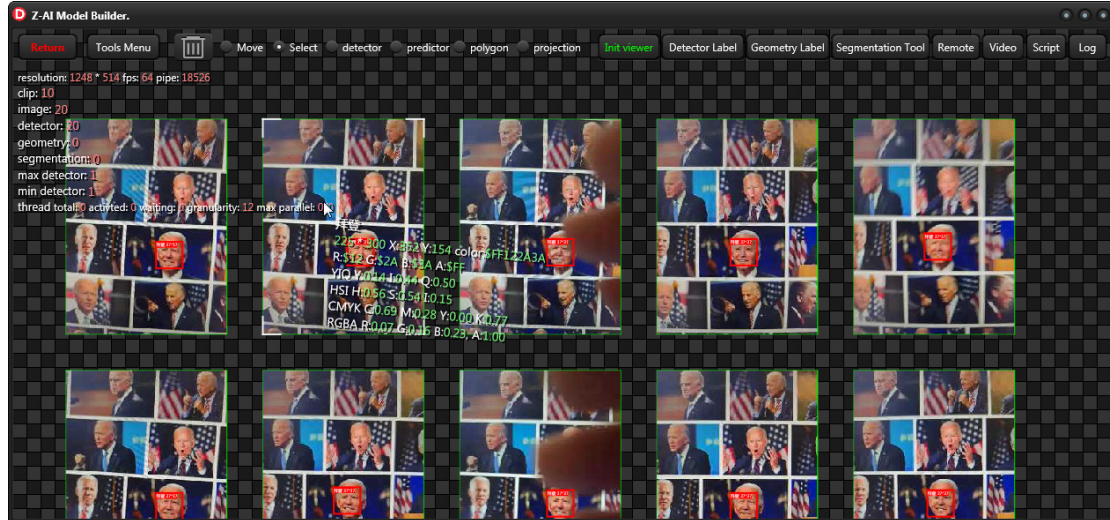
- 样本管理的作用在于维护远程人脸数据库
- 立即更新远程样本库按钮是把没有勾上的样本从服务器删除，某些样本如果觉得不够清晰，变形，可以去勾，然后点立即更新远程样本库
- 移除此人按钮，是把当前选择拜登全部从样本库移除
- 优化远程样本库按钮是个重要设置，默认会给每个人保留最多 200 张样本，多余的都会移除掉。这个功能也可以维护样本库，因为服务器的人脸样本库工作于内存中，过多样本库占用内存。一般来说，1000 的样本库人会额外用掉 5GB 内存，使用时需要注意。
- 内存条便宜，一般跑 GPU 的服务器，随便上 16G



- 在服务器端，点 Open FACE Database 可以打开查看当前人脸库。因为在服务器中，外部编辑器编辑过什么东西，服务器并不知道，所以，服务器只能查看拍错，并不是用于管理和维护人脸库。管理人脸库，用手机采集端即可。

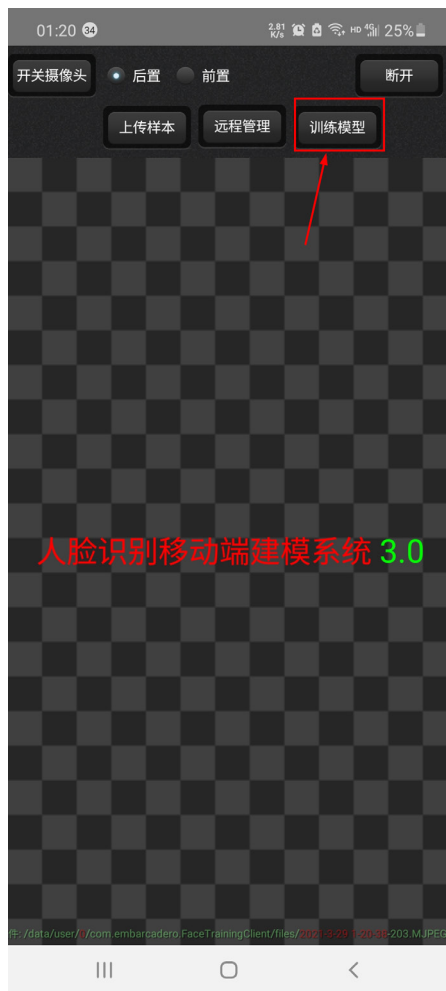


- 服务器端保存的样本库为优化尺度人脸库
- 例如手机采集的是 720p，实际保存在样本库中的数据，分辨率不会超过 300
- 这些样本库是按神经网络反向生成的优化样本库，可以在小内存开销下，堆到非常大，例如上千人只消耗 5GB 的内存

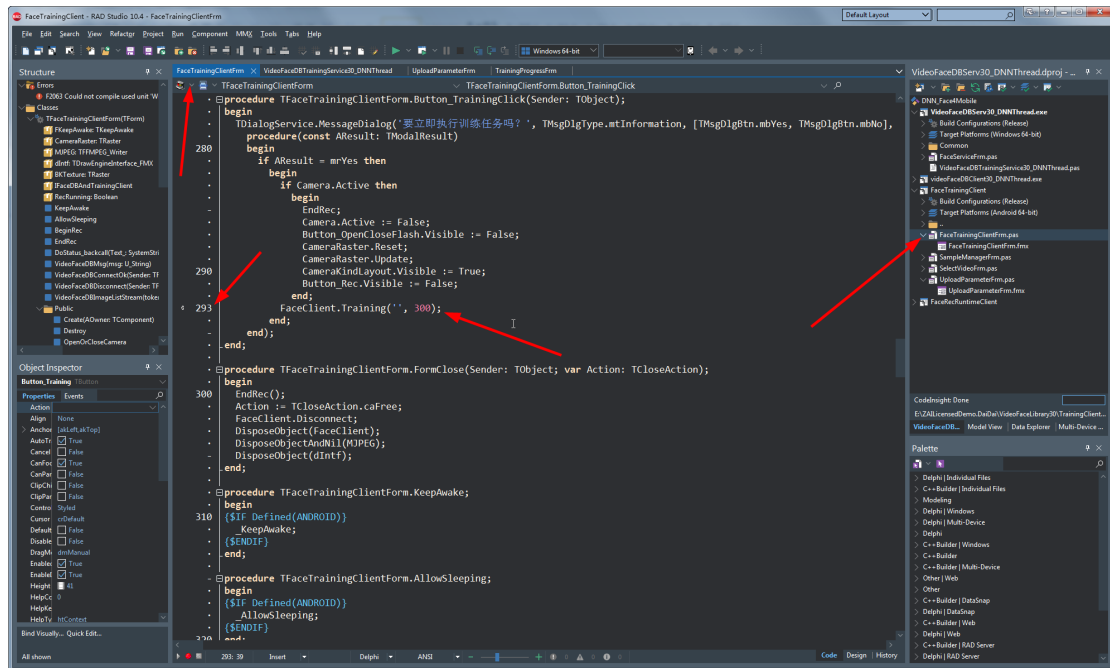


第六步，训练人脸模型

- 样本库中至少有 2 个人才能训练人脸，否则训练会失败
- 训练时会自动结束录制视频
- 训练只需要在主界面点训练模型，进入傻瓜训练流程，什么都不需要管，等就行了
- 服务器启动训练，完成训练，都会给出提示
- 服务器如果正在训练中，仍然可以提交样本，管理样本库，实时识别，不影响服务器工作
- 训练时会在服务器打开 TrainingTool 训练器，通过观察服务器可以了解到训练进度
- TrainingTool 训练器可以支持多 gpu 加速训练，多 GPU 训练只能在样本非常多的时候才有用，一般来说，人脸 Metric 度量化模型，一张普通 GPU 都足够了
- 考虑到模型精度问题，训练过程大约维持在 2-5 分钟，如果需要加速，可以自己二次开发修改代码
- 训练完成后，就可以用识别端运行看效果了

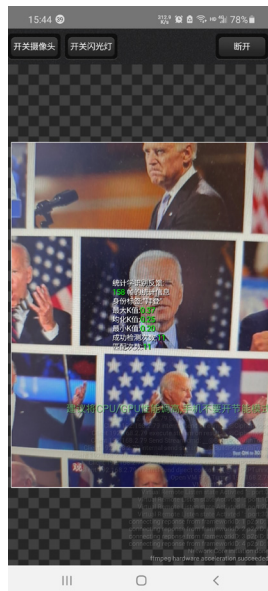


- 通过代码修改失效步数可以让训练更快，一般来说，我建议在 200-500 间

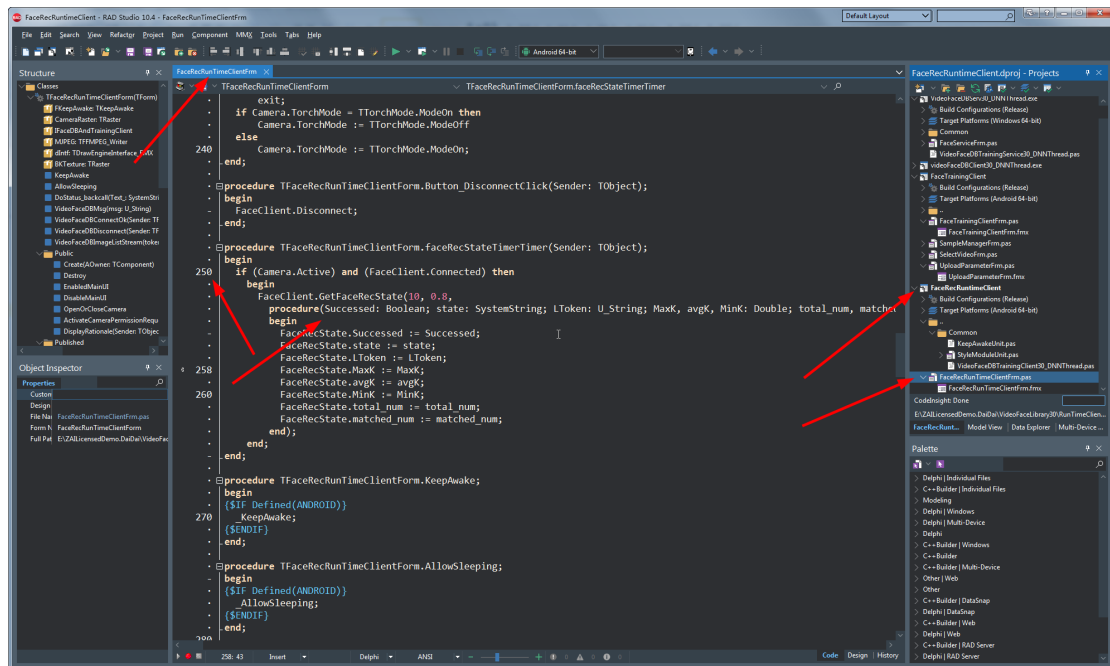


前台实时人脸识别系统

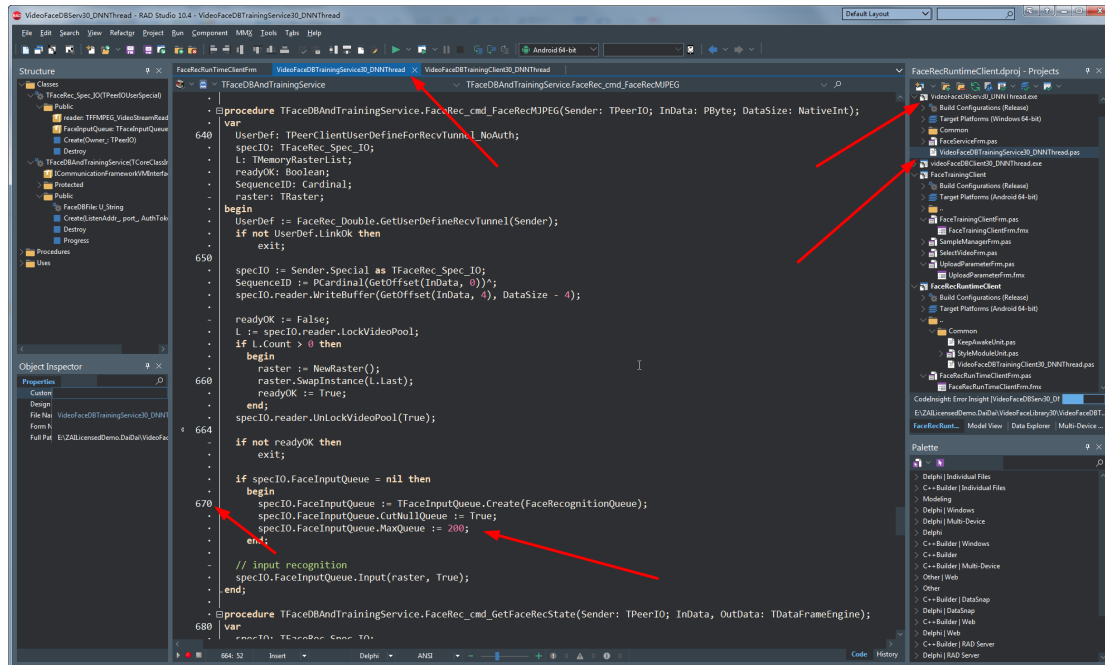
- 前台实时识别是一个独立的子系统，独立安装，如果需要合并的主工程，直接修改代码。
- 前台实时识别是实时往后台发摄像头的推流，找到合适的人脸时会提示识别结果
- 连接网络+打开摄像头的操作方式与采集端一致



- 二次开发时，通过 `GetFaceRecState` 方式，获取远程服务器的统计学计算结果
- 这些结果是根据多帧识别统计出来的结果
- `Successed` 表示是否找到了人脸，如果是自动化场景，`successed` 为 `true` 表示有人，否则可以当成是无人，具体的使用可以去参考 `GetFaceRecState` 的方法原型



- 实时识别的最大默认统计计算队列为 200，也就是说如果一直对某人实时识别，最大计算队列就是 200 帧，大约 3-10 秒，视手机采集效率而定，某些高端手机，每秒可以采集发送 60fps 给服务器，那就 3 秒，某些中低端手机，每秒只能采集 20 帧，那么就 10 秒。大概就这意思。
- 这个参数给的有点大，如果我们要修改 200 参数，按下图指引，修改完成服务器，编译重启



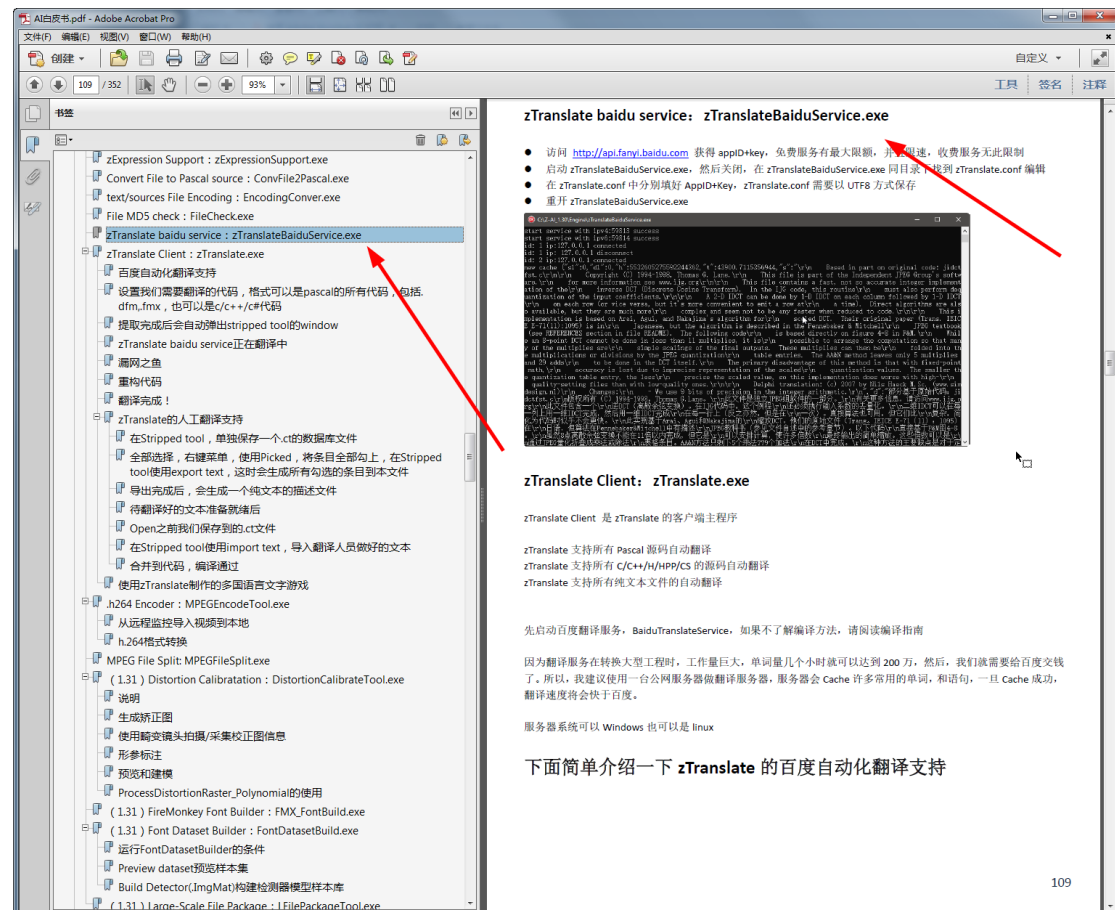
DNNTreadFace3.0 两个前台都是简体中文，如何制作多语言的前台系统

先在百度翻译注册个账号，DNNTreadFace3.0 翻译量很小，用免费账号就可以
<http://api.fanyi.baidu.com>

如果是海外用户，百度翻译不支持全球访问，需要自备翻墙到国内访问百度翻译

然后参考 AI 白皮书中关于 zTranslate 使用指南即可

一般来说，可以用自动翻译，也可以用人工翻译

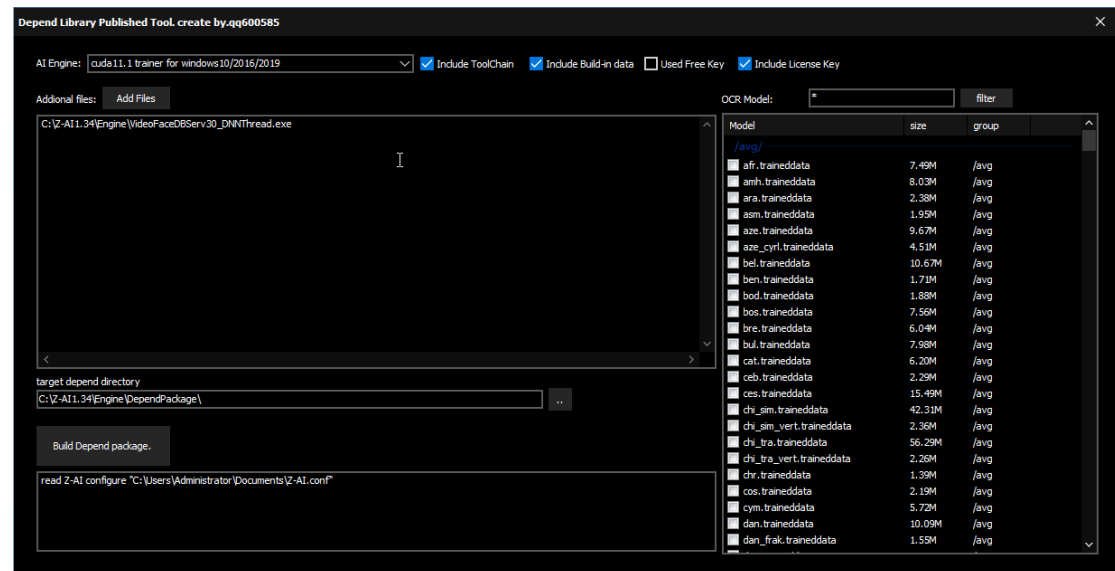


后台服务器的部署问题

后台服务器需要安装 cuda，如果服务器没有 gpu 那是不可以使用的，即时使用 MKL 那也无济于事。

后台服务器建议采用 16GB 内存+cpu4 核+任意 gpu

后台服务器的部署，只需要使用部署工具就可以完成打包分发，具体操作去看白皮书。



完

by.qq600585

2021/3/29