Android V20251221

FaceAI 人脸识别 SDK

API & 说明文档

Email: FaceAISDK.Service@gmail.com

微信WeChat: FaceAISDK

「简要说明」

FaceAI SDK包括人脸识别、活体检测、人脸录入检测以及1:N人脸搜索。可快速集成离线实现Android端侧相关功能。

SDK 可支持Android[5,15] 所有功能都在设备终端离线执行，**SDK本身不用联网，不上传不收集保存任何人脸信息敏感资料更具隐私安全**
动作活体支持张嘴、微笑、眨眼、摇头、点头 随机两种组合验证，支持系统前后摄像头和各种UVC协议双目摄像头。RGB静默活体需配备宽动态值大于105Db成像清晰抗逆光摄像头。

下表统计1万人脸在不同设备的SDK初始化启动速度和一次人脸搜索速度：

设备型号	启动速度	搜索速度(毫秒)
小米 13	79 ms	66 ms
RK3568-SM5	686 ms	520 ms
华为 P8	798ms	678 ms
联想Pad2024	245 ms	197 ms

SDK不限制人脸库容量，但强烈建议分组以减少人脸搜索匹配误差 .
3000测试人脸图链接: https://pan.baidu.com/s/1RfzJlc-TMDb0lQMFKpA-tQ?pwd=Face 提取码: Face

体验Android APK下载地址： https://www.pgyer.com/faceVerify

「集成方式」

目前SDK托管在Maven Central Repository,可通过gradle在线集成：
implementation 'io.github.faceaisdk:Android:VERSION.' **SDK仅需相机权限**

SDK集成演示Demo： https://github.com/FaceAISDK/FaceAISDK_Android

Demo默认AGP 版本 8.13 (非强制)， java17+ , kotlin 1.9.22 +

「人脸搜索-硬件配置要求」

FaceAISDK 对应CPU 配置要求不高，存储和内存大小请根据自身业务需要配置； 为了更好的体验和精度，人脸搜索识别功能对摄像头成像能力，光线环境有一定要求（同时注意摄像头保持物理干净，镜头不污染油污汗渍等）

一. 芯片，储存和内存要求

芯片只要是ARM架构的CPU都支持，4核2.0 GHZ的64位CPU足够，目前不支持X86等其他架构。

二. 摄像头参数建议（人脸搜索识别功能）

1. 低噪点和高感光能力，支持200万像素（1080P）预览画面，(默认使用640*480格式分析数据)
2. 宽动态范围（WDR）与背光补偿（BLC）结合

   室内环境：≥95 dB，室内均匀充足光，不过曝不过暗

   基础要求：≥105dB，可应对日常逆光场景（如门窗旁的逆光等）

   严苛环境：≥120dB，适用于强逆光、户外直射环境（如露天环境等）

人脸搜索识别如果无法使用红外双目摄像头请使用补光感应灯补充脸部光线

3. 低照度性能,彩色模式：≤0.01Lux（F1.2），支持弱光环境清晰成像。
4. 帧率：30fps左右，确保动态人脸无拖影，不要于25fps.

三. 摄像头硬件等级Camera Hardware Level 说明

Hardware Level从宏观上描述Camera设备具备的能力等级，可以使用代码具体判断的，参考集成Demo 开放源码FaceCameraXFragment 。

LEVEL_FULL：完整手动控制，高帧率录制，性能强劲。推荐使用本等级以上

LEVEL_3：YUV重处理，多流联动，FULL的超集，最新旗舰机会配置本类型摄像头

摄像头管理源码*CaneraX*已经暴露在*SDK*集成*Demo* 中，你可以根据定制硬件平台特性自由修改完善，也可以完全自定义。

「API 分类简要说明」

根据相关法规和用户反馈，SDKM从2025.11.24版本开始API大大简化。 我们简单分为人脸录入，人脸识别和人脸搜索。我们建议人脸特征都通过SDK提取更方便管理 使用网络传输，人脸图片在业务上没有需要的可以不再保存。

人脸录入

BaseImageDispose： 检测摄像头数据流中人脸角度合适且活体后返回裁剪好的人脸bitmap以便进一步提取人脸特征值（非常推荐使用此方式录入人脸）

FaceAISDKEngine.croppedBitmap2Feature: 将裁剪处理好的人脸bitmap转人脸特征值字符串

FaceAISDKEngine.saveCroppedFaceImage：**(可选)**保存裁剪处理好的人脸图到本地

通过FaceAISDKEngine.croppedBitmap2Feature获取人脸特征值后可供1:1和1:N方式使用

MMKV.encode(faceID, faceFeature)：**1:1** 人脸识别可使用腾讯MMKV保存Key- Value数据，key 为标识个人的唯一ID，Value 为对应人的人脸特征字符串

FaceSearchFeatureManger.insertFaceFeature 1:N人脸搜索特征值保存在内置数据库，支持按照分组和标记搜索，相关介绍见后文详细说明或参考SDK接入Demo

注：

Image2FaceFeature.getFaceFeatureByBitmap 检测一张照片中是否含有人脸并异步返回裁剪矫正好的人脸部分Bitmap和对应的人脸特征向量值字符(不推荐因为没有严格检测人脸合规)(注意和**FaceAISDKEngine**.croppedBitmap2Feature区别)

人脸识别

MMKV.decodeString(faceID) 1:1 人脸识别从MMKV 取出faceID对应人脸特征

FaceProcessBuilder：人脸识别中各种参数设置Builder，详情参考SDK接入Demo

FaceVerifyUtils.goVerifyWithImageProxy: 通过视频预览流参数ImageProxy进行识别

FaceVerifyUtils.goVerifyWithBitmap: 自定义管理摄像头视频帧转bitmap进行识别

FaceVerifyUtils.goVerifyWithNV21Bytes: 自定义摄像头NV21 Byte[]流识别12.8上线

FaceVerifyUtils.goVerifyWithIR: 通过红外(可选)和RGB Bitmap进行人脸识别

FaceVerifyUtils.evaluateFaceSimi：两张静态人脸图中人脸的相似度(开放使用)

人脸搜索

SearchProcessBuilder:人脸搜索中各种参数设置Builder，详情参考SDK接入Demo
FaceSearchEngine.runSearchWithImageProxy 通过摄像头预览流ImageProxy进行识别
FaceSearchEngine.runSearchWithBitmap 自定义管理摄像头视频帧转bitmap进行识别
FaceSearchEngine.runSearchWithIR：通过红外(可选)和RGB Bitmap进行人脸识别
FaceSearchEngine.getFeatureSearcher().search(faceFeature) 判断是否有相似度很高的人脸数据存在
FaceSearchFeatureManger：人脸搜索人脸向量特征库增删改查管理
FaceSearchImagesManger：人脸搜索人脸图片库增删改查管理

「人脸录入」

```
/**
 * BaseImageDispose 本类处理SDK通过摄像头预览流获自动化取角度正常的人脸图
 * context： 上下文context
 * mode: 人脸角度检测匹配模式
 * 2 精确模式 人脸要正对摄像头，严格要求
 * 1 快速模式 允许人脸方位可以有一定的偏移
 * 0 简单模式 允许人脸方位可以「较大」的偏移
 * BaseImageCallBack： 检测匹配成功回调 裁剪矫正好后的人脸图
 */
baseImageDispose(Context mContext, int mode,BaseImageCallBack callBack)
```

```
/**
 * FaceAISDKEngine 将裁剪后的人脸图片直接转换为 Base64 编码的特征字符串。
 *
 * @param croppedBitmap 裁剪好的人脸图片。
 * @return Base64 格式的特征字符串，可用于存储或网络传输。如果转换失败则返回 null。
 */
fun croppedBitmap2Feature(croppedBitmap: Bitmap): String
```

```
/**
 * FaceAISDKEngine 将裁剪后的人脸图片保存到本地文件。
 *
 * @param croppedBitmap 需要保存的人脸图片。
 * @param pathName 存储路径。
 * @param fileName 文件名。
 */
fun saveCroppedFaceImage(croppedBitmap: Bitmap, pathName: String, fileName: String)
```

```
/**
 * FaceSearchFeatureManger 插入或更新单个人脸特征数据。
 * 如果具有相同 faceID 的记录已存在，则会覆盖更新。
 *
 * @param faceID 人脸的唯一标识符。
 * @param faceFeature Base64 编码的人脸特征字符串。
 * @param updateTime 数据更新的时间戳。
 * @param tag 可选的标签信息，用于分类或备注。
 * @param group 可选的分组信息，用于将人脸划分到不同的组。
 */
fun insertFaceFeature(
    faceID: String,
    faceFeature: String,
    updateTime: Long,
    tag: String?,
    group: String?
)
```

「人脸识别」

FaceProcessBuilder：人脸识别中各种参数设置Builder，参数太多，详细见Demo
https://github.com/FaceAISDK/FaceAISDK_Android/blob/publish/faceAILib/src/main/java/com/faceAI/demo/SysCamera/verify/FaceVerificationActivity.java

MMKV.defaultMMKV().decodeString(faceID) 从MMKV取出faceID对应人脸特征

```
 /**
 * 通过视频预览流参数ImageProxy进行识别
 * @param imageProxy CameraX 的暴露的 ImageProxy
 * @param margin 暂时不用，预留参数
 */
public void goVerifyWithImageProxy(ImageProxy imageProxy, int margin)
```

```
/**
 * 通过视频预览流参数ImageProxy进行识别
 * @param imageProxy CameraX 的暴露的 ImageProxy
 */
public void goVerifyWithImageProxy（）
```

```
 /**
 *  FaceAISDKEngine 比较两张裁剪好的人脸图片的相似度。
 *
 * @param croppedBitmap1 第一张人脸图片（Bitmap 格式）。
 * @param croppedBitmap2 第二张人脸图片（Bitmap 格式）。
 * @return 返回一个浮点数，值域为 [0.0, 1.0]，表示两张人脸的相似度。值越大，表示越相似。
 */
fun evaluateFaceSimiByBitmap(croppedBitmap1: Bitmap, croppedBitmap2: Bitmap): Float
```

「人脸搜索」

SearchProcessBuilder:人脸搜索中各种参数设置Builder，参数太多详见
https://github.com/FaceAISDK/FaceAISDK_Android/blob/publish/faceAILib/src/main/java/com/faceAI/demo/SysCamera/search/FaceSearch1NActivity.java

```
/**
 * FaceSearchEngine 类中摄像头预览处理搜索
 *
 * @param imageProxy 实时采集的数据流 CameraX 的暴露的 ImageProxy
 * @param margin 预留参数
 */
fun runSearchWithImageProxy(imageProxy: ImageProxy, margin: Int)
```

```
/**
 * FaceSearchEngine类中摄像头预览处理搜索
 *
 * @param rgbBitmap 摄像头实时采集的数据流转为Bitmap，用于人脸搜索，活体检测（可选）
 */
fun runSearchWithBitmap(rgbBitmap: Bitmap)
```

```
/**
 * FaceSearchEngine类中摄像头预览处理搜索
 * @param irBitmap  实时IR摄像头采集的数据流转为Bitmap，用户IR活体检测
 * @param rgbBitmap 实时采集的数据流转为Bitmap，用于人脸搜索
 */
fun runSearchWithIR(irBitmap: Bitmap,rgbBitmap: Bitmap)
```

```
/**
 * 搜索人脸特征数据库中相似度最高的face以及对应ID
 *  FaceSearchEngine.getInstance().getFeatureSearcher(this).search(faceFeature)
 * @param faceFeature
 */
override fun search(faceFeature: String): FeatureSearchResult
```

```
/**
 * FaceSearchFeatureManger
* 插入或更新单个人脸特征数据。
 * 如果具有相同 faceID 的记录已存在，则会覆盖更新。
 *
 * @param faceID 人脸的唯一标识符。
 * @param faceFeature Base64 编码的人脸特征字符串。
 * @param updateTime 数据更新的时间戳。
 * @param tag 可选的标签信息，用于分类或备注。
 * @param group 可选的分组信息，用于将人脸划分到不同的组。
 */
fun insertFaceFeature(
    faceID: String,
    faceFeature: String,
    updateTime: Long,
    tag: String?,
    group: String?
)
```

```
/**
 * FaceSearchFeatureManger
 * 保存name(unique id)对应的人脸搜索特征向量编码到人脸数据库
 * @param name    人脸FaceID String
 * @param faceEmbedding 人脸特征向量编码
 */
fun insertOrUpdateFaceEmbedding(name:String,faceEmbedding: FloatArray)
```

```
/**
 *  FaceSearchFeatureManger批量插入或更新人脸搜索特征数据到本地数据库。
 * 此操作在后台协程中执行，不会阻塞主线程。
 *
 * @param list 包含多个 SearchEmbedding 对象的列表。
 */
fun insertOrUpdateFaceFeatures(list:List<SearchEmbedding>)
```

```
/**
 *  FaceSearchFeatureManger清空数据库中所有的人脸特征数据。
 * 此操作在后台协程中执行。
 */
fun clearAllFaceFaceFeature()
```

```
/**
 * FaceSearchFeatureManger 根据 faceID 查询单个人脸的特征数据。
 * 这是一个同步操作，会直接返回查询结果。
 *
 * @param faceID 要查询的人脸的唯一标识符。
 * @return 如果找到，则返回对应的 SearchEmbedding 对象；否则返回 null。
 */
fun queryFaceFeatureByID(faceID:String):SearchEmbedding
```

```
/**
 * FaceSearchFeatureManger 查询数据库中所有的人脸特征数据。
 * 这是一个同步操作，会直接返回查询结果。
 *
 * @return 包含数据库中所有人脸数据的 SearchEmbedding 列表。如果数据库为空，则返回一个空列表。
 */
fun queryAllFaceFaceFeature():List<SearchEmbedding>
```

```
/**
 * 匹配到的大于 Threshold的所有结果，如有多个很相似的人场景允许的话可以弹框让用户选择
 * SearchProcessBuilder setCallBackAllMatch(true) 才有数据返回 否则默认是空
 */
 @Override
 public void onFaceMatched(List<FaceSearchResult> matchedResults, Bitmap searchBitmap) {
    //已经按照降序排列，可以弹出一个列表框
    Log.d("onFaceMatched",matchedResults.toString());
 }
```

```
/**
 * 最相似的人脸搜索识别结果，得分最高
 * @param faceID  人脸ID
 * @param score   相似度值
 * @param bitmap  场景图，可以用来做使用记录log
 */
@Override
public void onMostSimilar(String faceID, float score, Bitmap bitmap)
```

「常见问答说明」

1.如何提升 1:N人脸识别的准确率?

参考专题文章 https://mp.weixin.qq.com/s/G2dvFQraw-TAzDRFIgdobA ,SDK不可用于金融，高安全级别场景，某些场景我们提供以下方法供业务侧避免搜索匹配错误无可选处理路径

```
/**
 * 匹配到的大于 Threshold的所有结果，如有多个结果场景允许的话可弹框让用户选择
 * SearchProcessBuilder setCallBackAllMatch(true) 才有数据返回 否则默认是空
 */
 @Override
 public void onFaceMatched(List<FaceSearchResult> matchedResults, Bitmap searchBitmap) {
    //已经按照相似度降序排列
    Log.d("onFaceMatched",matchedResults.toString());
}
```

2 .uniApp 原生插件支持

参考UTS demo项目集成 https://github.com/FaceAISDK/FaceAISDK_uniapp_UTS
目前已经支持iOS，Android 1:1人脸识别和人脸录入等，细节可以修改原生部分代码重新打包实现。

3.人脸识别的阈值设置说明

人脸识别和搜索都有相应的API设置阈值setThreshold(0.88f) //阈值范围限 [0.75 , 0.95]

阈值设置越大结果可信度越高，比如你设置为0.75可能和你同性别弟弟妹妹就能通过你的人脸识别但是越不是越高越好，设置越高需要你录入的人脸品质高以及摄像头成像能力也高（宽动态>105 Db可抗逆光）

4.FaceAI SDK 版权说明

FaceAI SDK 使用开源+自研封装实现，非虹软(试用每年要重新激活)，Face++，商汤 商业方案二次包装。

所有功能都可充分体验验证，更多请咨询： **FaceAISDK.Service@gmail.com**