云从科技人脸 SDK 常见问题说明

First printing, February 2017

前言

本文档对使用算法 SDK 中的一些常见问题进行汇总说明，旨在帮助大家更好的熟悉和使用算法 SDK，避免重复犯错，浪费不必要的时间。

这里的 SDK 主要是指算法前后端库以及模型，平台包括 windows，linux，android。

修改记录

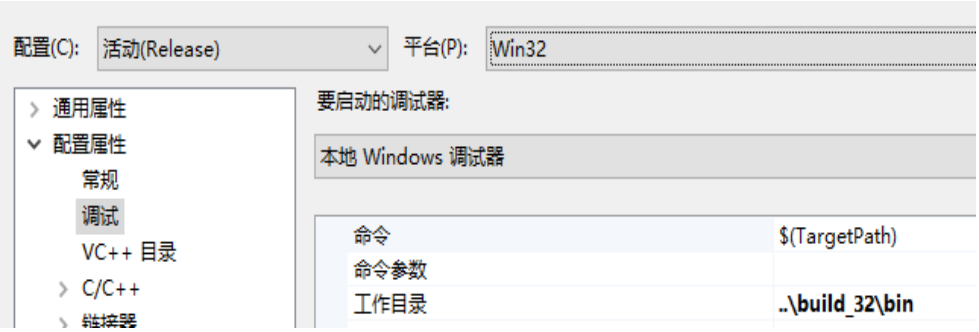
版本号	拟制/修改日期	主要更改内容
5.0.0	2018.09.09	首次撰写

目录

1. 句柄使用

1.1. 句柄创建

创建句柄要注意模型的路径问题，特别是 windows 下 VS 开发，调试时启动程序和直接双击 exe 启动程序的当前路径是不一样的，如果模型是设置的相对路径，会导致找不到模型，句柄创建失败。可以将模型路径设置为相对可执行程序 exe 的路径，然后配置调试路径解决问题，如下图：



1.2. 句柄使用

句柄创建需要加载模型，这个过程会比较耗时间和内存，因此程序中严禁反复创建销毁句柄，建议程序启动时创建句柄，程序退出时销毁句柄。

同一个句柄尽量在一个线程里使用，否则如果不上锁，程序会发生不可预料的结果。建议几个线程就用几个句柄，互不干涉。

2. 多线程

2.1. 问题说明

目前的前后端算法都不支持多线程调用，也就是说，同一个句柄不能同时在多个线程中使用，否则会造成未定义错误甚至应用程序崩溃。

另外，在程序中反复开关线程，或者反复创建销毁句柄，都会造成内存和时间的消耗，影响算法的效果。

2.2. 使用建议

针对以上两个问题，建议如下：

条件允许下，N 个线程使用 N 个句柄，一对一绑定，这是最安全的方式。也可采用句柄池的方式，但要确保一个句柄不能同时在多个线程中使用。

程序启动时创建好线程和句柄，程序退出时再销毁句柄和线程，程序运行中严禁反复开关线程和反复创建销毁句柄。

3. 参数设置

人脸 SDK 的性能标准主要包括两个方面，处理时间和准确率。不同应用场景下，侧重点有所不同。算法中提供了多种参数配置方式，便于客户根据实际需求，控制速度和准确度的平衡。

3.1. 人脸检测

3.1.1. 设置检测参数：

人脸检测参数主要分为两部分，一部分在头文件 cw_det_param_t 结构体中，一部分在模型配置_configs_dl_traditional.xml 文件中，通过 cwSetFaceParam 进行设置。

头文件中的参数如下图：

```
    /***************
    * 接口功能参数
    */
typedef struct cw_det_param
{
    int roiX;                           // roi，默认整帧图像0, 0, 0, 0 若设置为异常值检测阶段将恢复默认
    int roiY;
    int roiWidth;
    int roiHeight;

    int minSize;                        // 检测人脸尺寸范围，  pc端默认[48,600];移动端默认[100,400]
    int maxSize;
    const char* pConfigFile;            // 内部参数配置文件路径，此参数只能设置(set);从句柄内部获取出来的一律无效
} cw_det_param_t;
```

ROI：设置检测 ROI 区域，区域越小，速度越快，区域外的人脸无法检测

minSize：检测人脸最小尺寸，值越大，速度越快，小于该尺寸的人脸无法检测

maxSize：检测人脸最大尺寸，值越小，速度越快，大于该尺寸的人脸无法检测

模型配置中的参数如下图：

```
<Face_DetTrack_Params>
  <!-- 每帧最大人脸数，默认20 -->
  <Faces_Num_Max_Frame>20</Faces_Num_Max_Frame>
  <!-- 检出性能,范围1-10，越大检出率越低，但误检越小．默认2 -->
  <Neighbors_Num_Min>2</Neighbors_Num_Min>
  <!-- 检测精细度，范围1.1 ~ 1.80，越大速度越快，但越不精细．默认1.20 -->
  <Scale_Up_Ratio>1.2500000476837158e+000</Scale_Up_Ratio>
  <!-- 检测精细度，范围0.1 ~ 0.25，越大速度越快，但越不精细．默认0.13 -->
  <Win_Step_Ratio>1.2999999523162842e-001</Win_Step_Ratio>
  <!-- 局部检测子图扩展比例，范围[1.4,3.0]，默认1.80 -->
  <Local_ImgSize_Expand_Ratio>1.7999999523162842e+000</Local_ImgSize_Expand_Ratio>
  <!-- 局部检测人脸扩展比，范围[1.2~1.5]，默认1.30 -->
  <Local_FaceSize_Range_Ratio>1.2999999523162842e+000</Local_FaceSize_Range_Ratio>
  <!-- 跟踪开关下，后续计算（关键点或对齐或质量分）开关打开时的隔帧参数,范围[0,2]，0-所有检测人脸都进行后续计算，1-仅全局检测人脸不计算，2-每隔一帧进行后续计算．默认2 -->
  <Post_Detect_Frequency>0</Post_Detect_Frequency>
  <!-- 没有检测到的人脸是否进行"串联+跟踪".0关闭，非0开启，默认1开启 -->
  <Lost_Track>1</Lost_Track>
  <!-- 预跟踪帧数，默认2 -->
  <Frame_Num_For_New>2</Frame_Num_For_New>
  <!-- 检测性能水平，相当于检测图缩小比例，范围(1~6).数字越小缩小比例越大,反之亦然.默认1，如果性能不好建议6(不做任何缩放)，如果检测超大图可设置1~5．-->
  <Perfmon_Level>1</Perfmon_Level>
  <!-- 图像预处理方式，0-不做任何处理，1-高斯模糊预处理，其他-不错处理，默认0不做预处理．-->
  <Image_Preprocess_Mode>0</Image_Preprocess_Mode>
  <!-- 对齐人脸图颜色空间类型:1-灰度图;2-双通道图;3-彩色图 -->
  <ImgAligned_Color_Mode>1</ImgAligned_Color_Mode>
  <!-- 输出人脸排序类型：0-不排序，1-按人脸大小从大到小排序，2-按id从小到大排序；3-从左到右排序；4-从上到下排序． 默认0． -->
  <Face_Order_Type>1</Face_Order_Type>
  <!-- 光照分开关 -->
  <Light_Score_SWITCH>1</Light_Score_SWITCH>
  <!-- 模糊分开关 -->
  <Blur_Score_SWITCH>1</Blur_Score_SWITCH>
  <!-- 对称分开关 -->
  <Symmetry_Score_SWITCH>1</Symmetry_Score_SWITCH>
  <!-- 嘴巴分开关 -->
  <Mouth_Score_SWITCH>0</Mouth_Score_SWITCH>
  <!-- 眼镜分开关 -->
  <Glass_Score_SWITCH>0</Glass_Score_SWITCH>
  <!-- 眼睛分开关 -->
  <Eye_Score_SWITCH>0</Eye_Score_SWITCH>
  <!-- 肤色分开关 -->
  <Skin_Score_SWITCH>1</Skin_Score_SWITCH>
  <!-- 墨镜分开关 -->
  <Sunglass_Score_SWITCH>0</Sunglass_Score_SWITCH>
  <!-- 所有质量开关综合 -->
  <All_Score_SWITCH>0</All_Score_SWITCH>
  <!-- Face_DetTrack_Params -->
  </Face_DetTrack_Params>
```

Neighbors_Num_Min：检出性能，范围 1-10，越大检出率越低，但误检越小

Scale_Up_Ratio：检测精细度，范围 1.1 ~ 1.80，越大速度越快，但越不精细

Post_Detect_Frequency：设置为 0，每帧都会进行后续计算（关键点，质量分与人脸对齐操作），速度慢，设置为 2，隔帧进行后续检测，速度快

Perfmon_Level：检测性能水平，设置越大，速度越慢，检测效果越好

Mouth_Score_SWITCH：是否戴眼镜分开关，设置为 1，可以获取该分数，检测耗时增加，设置为 0，不做眼镜检测，无法获取该分数，耗时减少。所有质量分开关，均与此类似

3.1.2. 设置算法功能开关：

用什么功能，就开什么开关。例如只做检测，就只打开 CW_OP_DET，不要开对齐，质量分的开关。

如果后续需要提特征，就必须打开对齐开关 CW_OP_ALIGN，获取对齐人脸，但该开关也是最耗时间的。

如果需要获取质量分，需要打开质量开关 CW_OP_QUALITY，打开该开关后，只能获取质量总分，需要获取其他质量开关，例如光照，嘴巴，眼睛等分数，需要设置模型配置文件中的对应参数。相应的耗时也会增加。

3.1.3. 控制图片大小

图片越小，速度越快。当图片尺寸过大，可以先按比例压缩，再进行人脸检测。但不建议压缩到 640*480 尺寸以下，会造成人脸过小而检测不到的情况。

3.1.4. 替换模型

对于检测模型，一般情况下，模型越小，速度越快，效果越差，对于关键点模型，关键点数量越少，速度越快，可以根据实际场景替换不同的模型。

3.2. 人脸识别

从算法上，选择质量分更高的人脸来做识别，可以大大提高识别准确率。

从模型上，一是替换模型，随着训练数据量的日益扩大，算法会持续提供效果更好的模型。算法和模型是分开的，替换模型不需要重新编译程序，但是原有的特征将不能再使用，需要用新的模型重新提特征；二是配置 xml 文件中的参数“Flip”，设置为“false”要比设置为“true”快一倍。该参数如果设置为“true”，提特征时要将图片水平翻转后再提一次特征，因此时间翻倍。

4. 人脸比对识别流程

人脸检测—>关键点提取—>人脸对齐—>特征提取—>人脸识别比对

4.1. 人证比对流程

图片 1 人脸检测对齐 (cwFaceDetection) → 提特征 1 (cwGetFaceFeature)

图片 2 人脸检测对齐 (cwFaceDetection) → 提特征 2 (cwGetFaceFeature)

比对特征 1 与特征 2 (cwComputeMatchScore)，获取分数

若比对分数大于阈值 0.7，认为是同一个人，否则不是同一个人

4.2. 人脸识别流程

建模 → 加载 → 识别

1）建模流程

底库 N 个图片提取 N 个特征：cwFaceDetection → cwGetFaceFeature

N 个特征以二进制格式保存到本地文件，或者保存数据库

2）加载流程

将 N 个底库特征加载到内存

```
char *feaFiledAll = new char[N * iFeatureLen];
```

3）识别流程

待识别图片提取特征：cwFaceDetection → cwGetFaceFeature

1 个特征与 N 个特征比对：cwComputeMatchScore

获取 N 个分数，顺序与特征顺序一致，按照分数高度排序，获取 Top1

Top1 的分数大于阈值 0.85，则认为识别成功，否则识别失败

5. Q&A 常见问题回答

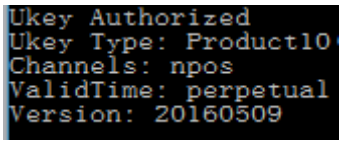
5.1. 常见错误码

所有的错误码均可以在头文件和开发文档中找到对应的错误解释，以下针对一些常见的错误进行说明。

5.1.1. 20015

CW_UNAUTHORIZED_ERR：20015，未安装商用 SDK 专用授权。使用云从商用 SDK，需要安装云从授权，PC 是安装 hasp 锁的方式，安卓是使用 cw 授权码的方式。

如果安装了授权仍报这个错，有可能是没有安装商用 SDK 专用 key，可使用 UkeyCheck 查看授权情况，如下图：



第二行，Ukey Type 的值为 Product10 或者为 Master 才表示 SDK 的 key 安装成功。

5.1.2. 20022

CW_EXCEEDMAXHANDLE_ERR：20022，超过授权最大句柄数。这个错与 Key 相关，能够创建个句柄数量是从 key 中读取，例如 key 中能够创建的句柄数母为 1，当 SDK 创建第二个检测句柄时，就会报这个错，客户需要与业务联系，从商务上解决该问题。

5.1.3. 20012

CW_UNINITIALIZED_ERR：20012，尚未初始化。如果是提特征接口，检查是否对齐人脸数据为空；如果是其他接口，检查是否句柄为空。

5.1.4. 20019

CW_FILE_UNAVAILABLE：20019，文件不存在。检查创建句柄传的模型配置文件 xml 是否存在，特别是传的路径为相对路径时，容易造成路径错误的

问题。

5.1.5. 20023

CW_RECOG_FEATURE_MODEL_ERR：20023，加载特征识别模型失败。检查创建识别句柄传的模型配置文件 xml 是否存在，与 20019 错误类似，程序找不到传入的模型配置文件。

5.2. 人脸图片标准

由于目前人脸识别算法是基于人脸二维图片进行的识别，算法系统对于人脸图片有相应的基本要求。而视频又可以通过编解码技术转化为人脸图片帧进行处理，以下要求仅以图片要求提出：

支持格式	所有图片和视频帧格式
角度	旋转角<45°，俯仰角<30°，偏航角<30°。
光照	光照均匀，背光与曝光不严重均可做识别。
模糊	人脸轮廓较清楚，五官位置可辨认。
整体性	必须保证完整人脸，人脸轮廓全部在图片内；
人脸像素	瞳间距 15 像素以上可检测；瞳间距 30 像素以上可识别。
图片像素	系统支持 30*30 像素至 5000*4000 像素图片输入处理。

5.3. 获取高质量人脸

人脸的质量与识别效果关系很大，人脸质量过低，会导致很高的误识率。因此在人脸检测时，要在满足需求的前提下，尽可能获取高质量的人脸。

如下图，一般判断人脸质量高低通过判断人脸质量总分，大于 0.65，则认为人脸质量比较高。阈值一般建议 0.3-0.5 左右，设置太高，容易导致抓不到人脸，设置太低，会导致误识率增高，根据实际需求调整。

要获取人脸质量总分，需要在检测接口 cwFaceDetection 的最后一个参数中加上质量分开关 CW_OP_QUALITY，否则该值为 0。

5.4. 人脸比对识别阈值

一般 1:1 的阈值建议设置为 0.7，大于 0.7，认为是同一个人，否则认为不是同一个人。可根据实际需求调整，但建议不超过 0.75。

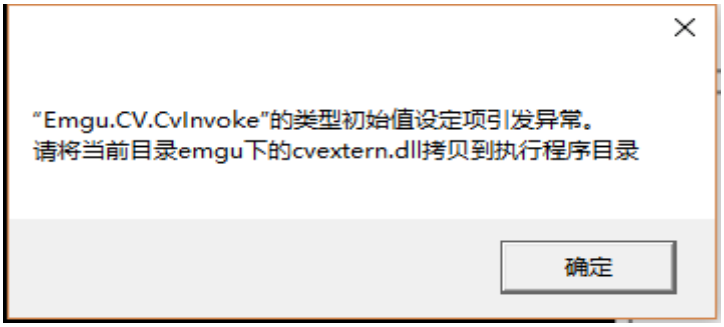
一般 1:N 的阈值建议设置为 0.8-0.9 之间，底库越大，阈值相应要设置大一些。根据实际场景需求，保证识别率和误识率的平衡。

5.5. Demo_CSharp

5.5.1. 说明

Demo_CSharp 采用 WPF 开发，实现了视频实时检测，显示，以及 1:1 比对功能。

5.5.2. Emgu 异常



由于 SDK 只提供了 Release 版本，因此 VS 默认的配置环境为 Release，建议客户打开工程后采取该默认配置。使用 x86 平台调用 32 位 SDK，会将可执行程序编译到 SDK 目录下的“build_32/bin”中，然后将“Demo_CSharp/emgu/x86”中的“cvextern.dll”库也拷贝到该运行程序目录中。使用 x64 平台调用 64 位 SDK，会将可执行程序编译到 SDK 目录下的“build_64/bin”中。然后将“Demo_CSharp/emgu/x64”中的“cvextern.dll”库也拷贝到对应的运行程序目录中。该 Demo 使用第三方库“cvextern.dll”，依赖 vc2013 运行库，主要用来做视频解码，因此要打开摄像头，需要安装 vc2013 运行库。

5.5.3. 找不到 CWFaceSDK.dll

SDK 依赖 vc2012 运行库，需要安装 vc2012 运行库。由于算法 SDK 采用

VS2012 开发，为了算法运行的稳定和效果，建议客户采用 VS2012 进行开发。

5.6. Demo_Java

5.6.1. 说明

Demo_Java 为 Eclipse 工程，使用的 JDK 版本为“jdk1.8.0”。通过一个控制台程序，实现了人脸检测，人脸质量，人脸 1:1 比对，1:N 识别，以及人脸属性等功能。

该 Demo 并不直接调用 SDK，而是调用在 SDK 基础上封装的 jni 库“CWFaceSDKJni”。Jni 库依赖 SDK，需要将所有库拷贝到一个目录中，然后设置环境变量，让程序能够 load 这些库。

5.6.2. 怎样配置环境变量

Windows 系统，jni 库位于“win_x64_jni”，依赖上层目录“build_64\bin”中的 SDK 库。将这些 jni 库和 SDK 库拷贝到一起后设置环境变量。最方便的设置方法就是把这些库全部拷贝到 jdk 目录下的 bin 目录。

Linux 系统，jni 库位于“linux_x64”，依赖上层目录“lib64”中的 SDK 库。同样，将这些库拷到到一起后，需要设置环境变量。配置环境变量的一种方法：打开“/etc/profile”配置文件，最后一行加上“export LD_LIBRARY_PATH=jni 库的路径”，执行“source /etc/profile”，或者重启电脑，让配置生效。

5.7. Demo_Android

5.7.1. 说明

Demo_Android 为 Eclipse 工程，实现了人脸图片检测，人脸视频实时检测，人脸关键点，人脸 1:1 比对，1:N 识别，以及人脸属性等功能。

5.7.2. 怎样使用授权码

创建句柄时，需要传入有效授权码，否则创建失败。Demo 中，需要在“ConStant.java”文件中，对应的位置输入正确授权码：

```
// 授权码 ，由云从科技提供，也可调用网络授权接口cwGetLicence获取
public static String sLicence="";
```

5.7.3. 怎样在实时检测功能的基础上做 1:1

如果只使用一个句柄：在检测接口的组后一个参数加上对齐开关，检测到人脸后，获取对齐数据，然后提特征，与证件照的特征进行比对。

如果使用两个句柄：实时检测线程使用一个句柄，检测到人脸就传输到提特征线程，提特征线程获取到人脸后再做一个检测，加上对齐开关，获取对齐人脸数据，然后提特征，比对。这种方式的好处是，基本不会丢帧，实时检测视频界面不会卡顿。