人脸核身

API 文档

产品文档

腾讯云

文档目录

API 文档

API 概览

最近更新时间：2018-06-19 13:13:50

为满足广大开发者的开发需求，腾讯云图像智能为您提供了丰富的人脸核身 API 接口。

人脸核身接口

功能	详细说明
用户上传照片身份信息核验	查看文档
活体检测—获取唇语验证码	查看文档
活体检测视频身份信息核验	查看文档
活体检测视频与用户照片的对比	查看文档
人脸静态活体检测	查看文档

错误码说明

最近更新时间：2018-06-06 16:14:52

人脸错误码说明

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 appid/bucket 与操作目标不匹配
9	签名过期
10	appid 不存在
11	secretid 不存在
12	appid 和 secretid 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
107	鉴权服务内部错误
108	鉴权服务不可用
213	内部错误
-1101	人脸检测失败
-1102	图片解码失败

错误码	含义
-1103	特征处理失败
-1104	提取轮廓错误
-1105	提取性别错误
-1106	提取表情错误
-1107	提取年龄错误
-1108	提取姿态错误
-1109	提取眼镜错误
-1200	特征存储错误
-1300	图片为空
-1301	参数为空
-1302	个体已存在
-1303	个体不存在
-1304	参数过长
-1305	人脸不存在
-1306	组不存在
-1307	组列表不存在
-1308	url 图片下载失败
-1309	人脸个数超过限制
-1310	个体个数超过限制
-1311	组个数超过限制
-1312	对个体添加了几乎相同的人脸
-1400	非法的图片格式
-1403	图片下载失败

图片鉴黄错误码说明

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	appid/bucket/url不匹配
7	签名编码失败（内部错误）
8	签名解码失败（内部错误）
9	签名过期
10	appid 不存在
11	secretid 不存在
12	appid 不匹配
13	重放攻击
14	签名失败
15	操作太频繁，触发频控
16	内部错误
17	未知错误
200	内部打包失败
201	内部解包失败
202	内部链接失败
203	内部处理超时
-1300	图片为空
-1308	url 图片下载失败
-1400	非法的图片格式

错误码	含义
-1403	图片下载失败
-1404	图片无法识别
-1505	url 格式不对
-1506	图片下载超时
-1507	无法访问 url 对应的图片服务器
-5062	url 对应的图片已被标注为不良图片，无法访问（专指存储于腾讯云的图片）

OCR识别错误码说明

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	appid/bucket/url不匹配
7	签名编码失败（内部错误）
8	签名解码失败（内部错误）
9	签名过期
10	appid 不存在
11	secretid 不存在
12	appid 不匹配
13	重放攻击
14	签名失败
15	操作太频繁，触发频控
16	Bucket 不存在

错误码	含义
17	url 为空
18	没有图片或 url
19	图片数过多，单次请求最多支持20个 url 或文件
20	图片过大，单个文件最大支持 1MB
21	无效的参数
200	内部打包失败
201	内部解包失败
202	内部链接失败
203	内部处理超时
-1102	图片解码失败
-1300	图片为空
-1301	请求的参数为空
-1308	url 图片下载失败
-1400	非法的图片格式
-1403	图片下载失败
-1404	图片无法识别
-1505	url 格式不对
-1506	图片下载超时
-1507	无法访问 url 对应的图片服务器
-5062	url 对应的图片已被标注为不良图片，无法访问（专指存储于腾讯云的图片）
-5103	OCR 识别失败
-5107	提供的图片不是身份证

人脸核身错误码说明

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 appid/bucket 与操作目标不匹配
9	签名过期
10	appid 不存在
11	secretid 不存在
12	appid 和 secretid 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
24	无权限，未申请服务
107	鉴权服务不可用
108	鉴权服务不可用
213	内部错误
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与Card照片相似度计算失败
-4011	照片解码失败
-4012	照片人脸检测失败

错误码	含义
-4015	自拍照人脸检测失败
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效，上传文件不符合视频要求
-5002	唇语失败
-5005	自拍照解析照片不足，视频里检测到的人脸较少
-5007	视频没有声音
-5008	语音识别失败，视频里的人读错数字
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动检测失败，视频里的人嘴巴未张开或者张开幅度小
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小 270*480
-5016	视频里的人不是活体（翻拍等攻击）
-5801	请求缺少身份证号码或身份证姓名
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效
-5805	用户未输入图像或者 url 下载失败

图片处理

错误码	含义
-5999	参数错误
-5998	签名格式错误
-5997	后端网络错误
-5996	HTTP 请求方法错误
-5995	文件大小错误
-5994	url 参数解析不匹配
-5993	multipart/formdata 参数错误
-5992	请求参数错误
-5991	分片过大
-5990	找不到 filecontent
-5989	上传失败
-5988	cgi 初始化错误
-5987	wup 编码失败
-5986	wup 解码失败
-5985	获取路由失败
-5984	sha1 不匹配
-5983	错误的 session
-5982	建立连接错误
-5981	建立连接错误

图片下载错误码说明

图片下载失败时，返回包 headers 中的 X-ErrNo 字段错误码说明。

错误码	含义

错误码	含义
-6101	图片不存在、图片没有上传或者图片已经删除
-5062	图片涉嫌违禁
-902	镜像存储功能把请求转发到开发商源站，但没有收到响应，超时了
-100	未知错误，某些场景或者逻辑未定义
-101	存储文件失败
-103	无效的请求；请求报文无法识别
-104	无效的 appid 。Url 中包含的 appid 无效,或者域名没有和 appid 绑定
-105	无效的样式名。 Url 中指定的样式名或者别名没有配置
-106	无效的 URL 。 Url 格式不符合格式要求
-107	无效的 Host 头域
-108	无效的 Referer
-109	无效的样式名 ID 。没有找到该样式名对应的图片。可能是上传该图片后，新增的样式名， 因此图片上传时不能生成该样式名对应的数据
-110	该图片在黑名单中
-111	http 服务器内部错误
-120	回源到源站获取数据时，源站返回的数据有异常，无法正常获取到图片数据
-121	http 服务器内部错误
-122	图片数据没有修改，客户端可以使用缓存数据
-123	图片数据没有修改，客户端可以使用缓存数据
-124	下载偏移错误。 Http 请求的 Range 断点续传偏移量可能设置错误
-129	无法预料的错误
-130	http 服务器内部错误
-140	http 服务器内部错误

图片插件错误码说明

错误码	含义
-2000	服务过载
-1900	框架 handle process 错误
-1899	存储文件失败
-1898	校验 md5 失败
-1897	秒传失败
-1896	编码失败
-1895	解码失败
-1894	业务 ID 错误
-1893	图片数据异常，压缩库无法处理
-1892	上传失败，服务器错误
-1891	解码 biz_req 失败
-1890	编码 biz_req 失败
-1889	存储文件超时
-1888	压缩超时
-1887	校验 sha1 失败
-1886	图片 fileid 已经存在

文件缓存错误码说明

错误码	含义
-300	服务过载
-299	命令字未知
-298	解包失败

错误码	含义
-297	框架 handle process 错误
-296	打包失败
-295	文件数据异常
-294	文件数据异常
-293	通知插件失败
-292	文件缓存服务器错误
-291	编码 session 失败
-290	无效的 session
-289	插件拒绝上传
-288	process 打包失败
-287	process 解包失败
-286	解码 session 失败
-285	文件过大
-284	分片大小不一致
-283	分片过小

cmd错误码说明

错误码	含义
-199	文件移动失败
-198	重定向错误
-197	查无此文件
-196	网络请求失败
-195	网络请求失败

错误码	含义
-194	后端打包失败
-193	返回包打包失败
-192	请求包解析失败
-191	url 参数解析不匹配
-190	文件删除失败
-189	输入参数错误：download_url empty
-187	从 url 中解析参数失败
-186	暂不支持视频文件复制
-185	业务预处理失败
-184	业务后处理失败
-183	获取路由失败
-182	参数检验失败

proxy错误码说明

错误码	含义
-99	proxy 读取配置失败
-98	调用签名服务失败
-97	非法签名
-96	签名过期
-95	消息缺少 session 信息
-94	携带错误 session 信息
-89	proxy 转发 cmd 服务失败
-88	编码 cmd 服务消息失败

错误码	含义
-87	解析 cmd 服务消息失败
-86	proxy 转发 process 服务失败
-85	解析 process 服务应答失败
-84	获取 process L5 失败
-83	签名服务解包失败
-82	不存在此 appid
-81	签名为空
-80	非法的业务 ID
-79	secret id 不存在
-78	SDK 协议不匹配,请升级
-77	单次性签名已不可用
-76	单次签名没有 url
-75	不支持此操作
-74	多次签名-过期时间为 0
-73	单次签名-过期时间不为 0
-72	签名失败
-71	操作太频繁,请稍后再试
-70	appid/userid 与签名不匹配
-69	输入参数错误: download_url empty
-68	ip 直通车打包失败
-67	ip 直通车解包失败
-65	断点续传不支持携带数据包
-64	该业务已经被屏蔽

鉴权签名

最近更新时间：2018-07-10 14:58:45

签名与鉴权

智能图像识别服务通过签名来验证请求的合法性。开发者将签名授权给客户端，使其具备上传下载及管理指定资源的能力。

签名分为两种：

- 多次有效签名：签名中绑定或者不绑定文件 fileid，需要设置大于当前时间的有效期，最长可设置三个月，在此期间内签名可多次使用。
- 单次有效签名：签名中绑定文件 fileid，有效期必须设置为 0，此签名只可使用一次，且只能应用于被绑定的文件。

具体应用参见 签名适用场景。

签名算法

获取签名所需信息

生成签名所需信息必须使用主账号的，包括 APPID、Secret ID 和 Secret Key。

> **注意：**
>
> - 如果您已使用过API 密钥，或在2018年4月1日后接入智能图像服务，请使用 API 密钥 ；
> - 如果您已使用过100、101等开头的项目ID，可以继续使用 项目密钥，但建议使用API 密钥。2018年4月1日后创建的项目ID，不再支持使用项目密钥；
> - 目前仅支持使用主账号的 Secret ID 和 Secret Key，暂不支持子账号的使用，计划后续实现。

拼接签名串

拼接多次有效签名串：
a=[appid]&b=[bucket]&k=[SecretID]&e=[expiredTime]&t=[currentTime]&r=[rand]&f=[fileid]

拼接单次有效签名串：
a=[appid]&b=[bucket]&k=[SecretID]&e=[expiredTime]&t=[currentTime]&r=[rand]&f=[fileid]

注意：

- 多次有效签名串中 fileid 为可选参数；
- fileid 为空，表示不绑定资源，例如上传签名和下载签名；
- fileid 不为空，表示绑定资源，例如绑定资源的下载。

签名串中各字段含义如下：

字段	解释
a	开发者的 APPID，接入智能图像时由系统生成
b	Bucket，图片资源的组织管理单元，历史遗留字段，可不填
k	Secret ID
e	签名的有效期，是一个符合 UNIX Epoch 时间戳规范的数值，单位为秒；单次签名时，e 必须设置为 0
t	当前时间戳，是一个符合 UNIX Epoch 时间戳规范的数值，单位为秒，多次签名时，e 应大于 t
r	随机串，无符号 10 进制整数，用户需自行生成，最长 10 位
f	资源存储的唯一标识，单次签名必填；多次签名选填，如填写则会验证与当前操作的文件路径是否一致。

注意：

- 拼接单次有效签名串时，有效期e必须设置为 0，以保证此签名只能针对固定资源使用一次；
- 删除和复制文件必须使用单次有效签名，上传必须使用多次有效签名；
- 具体应用参见 签名适用场景。

生成签名

1. 使用 HMAC-SHA1 算法对请求进行加密（ SHA1算法加密后的输出必须是原始的二进制数据，否则签名失败）；

2. 对 orignal 使用 HMAC-SHA1 算法进行签名，然后将 orignal 附加到签名结果的末尾，再进行 Base64 编码，得到最终的 sign；

3. 生成签名的公式如下：
   SignTmp = HMAC-SHA1(SecretKey, orignal)

Sign = Base64(SignTmp.orignal)

注意：

- 此处使用的是标准的 Base64 编码，不是 urlsafe 的 Base64 编码；
- SecretKey 为 API 密钥，orignal 为 2.2 节中拼接好的签名串。

PHP 签名示例

本节介绍生成签名的算法实例，实例中使用 PHP 语言，如果开发者使用其他与开发，请使用对应的算法。

获取签名所需信息

获取得到的签名所需信息如下：

APPID：YOUR APPID_ID
Bucket：tencentyun（可不填）
Secret ID： YOUR SECRET_ID
Secret Key： YOUR SECRET_KEY

拼接签名串

```
$appid = "YOUR APPID_ID";
$bucket = "tencentyun";
$secret_id = "YOUR SECRET_ID";
$secret_key = "YOUR SECRET_KEY";
$expired = time() + 2592000;
$onceExpired = 0;
$current = time();
$rdm = rand();
$userid = "0";
$fileid = "tencentyunSignTest";

$srcStr = 'a='.$appid.'&b='.$bucket.'&k='.$secret_id.'&e='.$expired.'&t='.$current.'&r='.$rdm.'&f=';

$srcWithFile = 'a='.$appid.'&b='.$bucket.'&k='.$secret_id.'&e='.$expired.'&t='.$current.'&r='.$rdm.'&
f='.$fileid;

$srcStrOnce= 'a='.$appid.'&b='.$bucket.'&k='.$secret_id.'&e='.$onceExpired .'&t='.$current.'&r='.$rd
m
.'&f='.$fileid;
```

生成签名

SHA1 算法加密后的输出必须是原始的二进制数据，否则签名失败：

```
$signStr = base64_encode(hash_hmac('SHA1', $srcStr, $secret_key, true).$srcStr);

$srcWithFile = base64_encode(hash_hmac('SHA1', $srcWithFile , $secret_key, true).$srcWithFile );

$signStrOnce = base64_encode(hash_hmac('SHA1',$srcStrOnce,$secret_key, true).$srcStrOnce);

echo $signStr."\n";

echo $srcWithFile ."\n";

echo $signStrOnce."\n";
```

JAVA 签名示例

```
/*
* Copyright 2017, Tencent Inc
* All rights reserved.
*
* Created on 2017年9月12日
*/
package sign;

import java.util.Base64;
import java.util.Random;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class Sign {

/**
* 生成 Authorization 签名字段
*
* @param appId
* @param secretId
* @param secretKey
* @param bucketName
* @param expired
```

```
 * @return
 * @throws Exception
 */
public static String appSign(long appId, String secretId, String secretKey, String bucketName,
long expired) throws Exception {
long now = System.currentTimeMillis() / 1000;
int rdm = Math.abs(new Random().nextInt());
String plainText = String.format("a=%d&b=%s&k=%s&t=%d&e=%d&r=%d", appId, bucketName,
secretId, now, now + expired, rdm);
byte[] hmacDigest = HmacSha1(plainText, secretKey);
byte[] signContent = new byte[hmacDigest.length + plainText.getBytes().length];
System.arraycopy(hmacDigest, 0, signContent, 0, hmacDigest.length);
System.arraycopy(plainText.getBytes(), 0, signContent, hmacDigest.length,
plainText.getBytes().length);
return Base64Encode(signContent);
}

/**
 * 生成 base64 编码
 *
 * @param binaryData
 * @return
 */
public static String Base64Encode(byte[] binaryData) {
String encodedstr = Base64.getEncoder().encodeToString(binaryData);
return encodedstr;
}

/**
 * 生成 hmacsha1 签名
 *
 * @param binaryData
 * @param key
 * @return
 * @throws Exception
 */
public static byte[] HmacSha1(byte[] binaryData, String key) throws Exception {
Mac mac = Mac.getInstance("HmacSHA1");
SecretKeySpec secretKey = new SecretKeySpec(key.getBytes(), "HmacSHA1");
mac.init(secretKey);
byte[] HmacSha1Digest = mac.doFinal(binaryData);
return HmacSha1Digest;
}

/**
```

```
 * 生成 hmacsha1 签名
 *
 * @param plainText
 * @param key
 * @return
 * @throws Exception
 */
public static byte[] HmacSha1(String plainText, String key) throws Exception {
return HmacSha1(plainText.getBytes(), key);
}

}
```

Node JS 签名示例

```
var crypto = require('crypto');

var secretId = 'YOUR SECRET_ID',
secretKey = 'YOUR SECRET_KEY',
appid = 'APPID',
pexpired = 86400,
userid = 0;

var now = parseInt(Date.now() / 1000),
rdm = parseInt(Math.random() * Math.pow(2, 32)),
plainText = 'a=' + appid + '&k=' + secretId + '&e=' + (now+pexpired) + '&t=' + now + '&r=' + rdm
+ userid + '&f=',
data = new Buffer(plainText,'utf8'),
res = crypto.createHmac('sha1',secretKey).update(data).digest(),
bin = Buffer.concat([res,data]);

var sign = bin.toString('base64');
```

C++ 签名示例

```
//g++ -g sign_sample.cpp -o sign -lcrypto

#include <stdio.h>
#include <stdlib.h> /* srand, rand */
```

```
#include <time.h> /* time */
#include <openssl/hmac.h>
#include <openssl/pem.h>
#include <openssl/bio.h>
#include <openssl/evp.h>
#include <string>
#include <vector>
#include <sstream>

#define HMAC_LENGTH 20

std::vector<unsigned char> hmac_sha1(std::string& data, std::string& key)
{
unsigned char* result;
unsigned int len = HMAC_LENGTH;

result = (unsigned char*)malloc(sizeof(char) * len);

HMAC_CTX ctx;
HMAC_CTX_init(&ctx);

HMAC_Init_ex(&ctx, key.c_str(), key.length(), EVP_sha1(), NULL);
HMAC_Update(&ctx, (unsigned char*)data.c_str(), data.length());
HMAC_Final(&ctx, result, &len);
HMAC_CTX_cleanup(&ctx);

std::vector<unsigned char> sha1;
for (int i = 0; i < len; i++){
sha1.push_back(result[i]);
}

free(result);
return sha1;
}

std::string base64_encode(const std::string& src){
BUF_MEM * bptr = NULL;
BIO* b64 = BIO_new(BIO_f_base64());
BIO_set_flags(b64, BIO_FLAGS_BASE64_NO_NL);
BIO* bmem = BIO_new(BIO_s_mem());
if(NULL == b64 || NULL == bmem){
return "";
}

bmem = BIO_push(b64, bmem);
```

```
int ret = BIO_write(bmem, src.data(), src.length());
if(ret <= 0){
return "";
}

ret = BIO_flush(bmem);
BIO_get_mem_ptr(bmem, &bptr);
std::string res(bptr->data, bptr->length);
BIO_free_all(bmem);
return res;

}


int main() {
std::string appid = "1000001";
std::string secret_id = "YOUR SECRETID";
std::string secret_key = "YOUR SECRETKEY";
time_t now = time(NULL);
long expired = (long)now + 2592000;
long onceExpired = 0;
long current = (long)now;
int rdm = rand();
std::string userid = "0";

std::stringstream raw_stream;
raw_stream << "a=" << appid << "&k=" << secret_id << "&e=" << expired << "&t=" << current
<< "&r=" << rdm;
std::string raw = raw_stream.str();

std::vector<unsigned char> sha1 = hmac_sha1(raw, secret_key);

std::stringstream data_stream;
for (int i = 0; i != sha1.size(); i++)
data_stream << sha1[i];
data_stream << raw;
std::string data = data_stream.str();

std::string sign = base64_encode(data);

printf("%s\n", sign.c_str());

return 0;
}
```

签名适用场景

签名的适用场景有如下限制：

场景	适用签名
智能鉴黄	多次有效签名
图片标签	多次有效签名
OCR识别	多次有效签名
人脸识别	多次有效签名
人脸核身	多次有效签名
人脸融合	多次有效签名

用户上传照片身份信息核验

最近更新时间：2018-12-18 15:59:33

接口描述

接口请求域名： https://recognition.image.myqcloud.com/face/idcardcompare

本接口（idcardcompare）用于判断给定一张照片与身份证号和姓名对应的登记照的人脸相似度，即判断给定照片中的人与身份证上的人是否为同一人。

⚠ 注意：

- 本接口支持 HTTPS 协议，如果您现在使用的是 HTTP 协议，为了保障您的数据安全，请切换至 HTTPS。
- 如果开发者使用的是原域名（service.image.myqcloud.com），为获得更好的体验，请及时切换到以上域名。

请求头 header

所有请求都要求含有以下头部信息：

参数名	必选	值	描述
host	是	recognition.image.myqcloud.com	腾讯云人脸核身服务器域名。
content-length	否	包体总长度	每个请求的包体大小限制为6MB，不支持.gif 类型的动图。
content-type	是	application/json 或 multipart/form-data	根据不同接口选择： 1. 使用 application/json 格式，参数为 url，其值为图片的 url。 2. 使用 multipart/form-data 格式，参数为 image，其值为图片的 base64。
authorization	是	鉴权签名	用于 **鉴权** 的签名。

⚠ 注意：

如选择 multipart/form-data，请使用 HTTP 框架/库推荐的方式设置请求的 content-type，不推荐直接调用 setheader 等方法设置，否则可能导致 boundary 缺失引起请求失败。

输入参数

使用 application/json 格式，参数选择 url ；使用 multipart/form-data 格式，参数选择 image。

参数名	必选	类型	说明
appid	是	String	接入项目的唯一标识，可在 账号信息 或 云 API 密钥 中查看。
idcard_number	是	String	用户身份证号码。
idcard_name	是	String	用户身份证姓名 （ 中文，请使用 UTF-8 编码 ）。
image	否	Binary	image 和 url 只提供一个即可。
url	否	String	image 和 url 只提供一个即可；如果都提供，只使用 url。
session_id	否	String	相应请求的 session 标识符，可用于结果查询。

输出参数

字段	类型	说明
data.session_id	String	相应请求的 session 标识符。
data.similarity	Float	用户上传的图像与身份证登记照的人脸相似度，取值范围[0,100]，推荐相似度大于75时可判断为同一人，可根据具体场景自行调整阈值。
code	Int	错误码。
message	String	错误描述。

示例

输入示例

使用 URL 的请求包

```
POST /face/idcardcompare HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: service.image.myqcloud.com
Content-Length: 187
Content-Type: application/json
```

```
{
"appid":"123456",
"idcard_number":"110110199909090909",
"idcard_name":"张三",
"url":"http://test-123456.image.myqcloud.com/test.jpg"
}
```

使用 image 的请求包

```
POST /face/idcardcompare HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: service.image.myqcloud.com
Content-Length: 735
Content-Type: multipart/form-data;boundary=----------acebdf13572468

----------acebdf13572468
Content-Disposition: form-data; name="appid";

123456
----------acebdf13572468
Content-Disposition: form-data; name="idcard_number";

110110199909090909
----------acebdf13572468
Content-Disposition: form-data; name="idcard_name";

张三
----------acebdf13572468
Content-Disposition: form-data; name="image"; filename="test.jpg"
Content-Type: image/jpeg

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------acebdf13572468--
```

输出示例

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 204
Content-Type: application/json

{
"data":{
```

```
"similarity":100.0,
"session_id":"",
},
"code":0,
"message":"OK"
}
```


错误码

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 APPID/Bucket 与操作目标不匹配
9	签名过期
10	APPID 不存在
11	SecretId 不存在
12	APPID 和 SecretId 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
24	无权限，未申请服务
107	鉴权服务不可用
108	鉴权服务不可用
213	内部错误

错误码	含义
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与Card照片相似度计算失败
-4011	照片解码失败
-4012	照片人脸检测失败
-4015	自拍照人脸检测失败
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效，上传文件不符合视频要求
-5002	唇语失败
-5005	自拍照解析照片不足，视频里检测到的人脸较少
-5007	视频没有声音
-5008	语音识别失败，视频里的人读错数字
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动检测失败，视频里的人嘴巴未张开或者张开幅度小
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小 270 * 480
-5016	视频里的人不是活体（翻拍等攻击）
-5801	请求缺少身份证号码或身份证姓名

错误码	含义
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效（因户口变动，极少数正常用户也会返回这个结果，此类用户无法使用本服务）
-5805	用户未输入图像或者 url 下载失败
-5809	用户身份证登记照无效（一般是派出所上传照片时出错，重新拍照片到身份证所属派出所提交可解决。
-5817	证件照质量较差，请到户籍所在地派出所进行核实

更多其他 API 错误码请查看 **错误码说明** 。

人脸静态活体检测

最近更新时间：2018-12-24 14:56:33

接口描述

接口请求域名： https://recognition.image.myqcloud.com/face/livedetectpicture

本接口（livedetectpicture）用于对用户上传的静态照片进行人脸活体检测。与动态活体检测的区别是：静态活体检测中，用户不需要通过唇语或摇头眨眼等动作来识别。

静态活体检测仅用于对防攻击要求不高的场景，如果对活体检测有更高安全性要求，请使用 唇语活体检测。

⚠ 注意：

- 本接口支持 HTTPS 协议，如果您现在使用的是 HTTP 协议，为了保障您的数据安全，请切换至 HTTPS。
- 如果开发者使用的是原域名（service.image.myqcloud.com），为获得更好的体验，请及时切换到以上域名。

请求头 header

所有请求都要求含有下表列出的头部信息：

参数名	必选	值	描述
host	是	recognition.image.myqcloud.com	腾讯云人脸识别服务器域名。
content-length	否	包体总长度	图片大小建议小于1280 * 720px；每个请求的包体大小限制为6MB；不支持 .gif 类型的动图。
content-type	是	application/json 或 multipart/form-data	据不同接口选择： 1. 使用 application/json 格式，参数为 url，其值为图片的 url。 2. 使用 multipart/form-data 格式，参数为 image，其值为图片的二进制内容。
authorization	是	鉴权签名	多次有效签名，用于鉴权，生成方式见 鉴权签名方法。

注意：
如选择 multipart/form-data，请使用 HTTP 框架/库推荐的方式设置请求的 content-type，不推荐直接调用 setheader 等方法设置，否则可能导致 boundary 缺失引起请求失败。

输入参数

使用 application/json 格式，参数选择 url；使用 multipart/form-data 格式，参数选择 image。

参数名	必选	类型	说明
appid	是	String	接入项目的唯一标识，可在 账号信息 或 云 API 密钥 中查看
image	否	Binary	图片内容（图片的宽高比请接近3：4，不符合宽高比的图片返回的分值不具备参考意义）
url	否	String	图片的 image 和 url 只提供一个即可，如果都提供，只使用 url（图片的宽高比请接近3：4，不符合宽高比的图片返回的分值不具备参考意义）

⚠ 注意：
图片的宽高比请接近3：4，不符合宽高比的图片返回的分值不具备参考意义。

输出参数

字段	类型	说明
data.score	Int	活体打分，取值范围 [0,100]，分数一般落于[80, 100]区间内，0分也为常见值。推荐值大于 87 时可判断为活体。可根据具体场景自行调整阈值。
code	Int	错误码。
message	String	错误描述。

示例

输入示例

使用 url

```
POST /face/livedetectpicture HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: recognition.image.myqcloud.com
Content-Length: 117
Content-Type: "application/json"

{"appid":"1000001","url":"http://test-123456.image.myqcloud.com/test.jpg"}
```

使用 image

```
POST /face/livedetectpicture HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: recognition.image.myqcloud.com
Content-Length: 641
Content-Type: multipart/form-data;boundary=----------acebdf13572468

----------acebdf13572468
Content-Disposition: form-data; name="appid";

123456----------acebdf13572468
Content-Disposition: form-data; name="image"; filename="face.jpg"
Content-Type: image/jpeg

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------acebdf13572468--
```

输出示例

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 168
Content-Type: application/json
{
"data":{
"score":30,
},
"code":0,
"message":"OK"
}
```

错误码

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 APPID/Bucket 与操作目标不匹配
9	签名过期
10	APPID 不存在
11	SecretId 不存在
12	APPID 和 SecretId 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
107	鉴权服务不可用
108	鉴权服务不可用
213	内部错误
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与Card照片相似度计算失败
-4015	自拍照人脸检测失败

错误码	含义
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效
-5002	唇语失败
-5005	自拍照解析照片不足
-5007	视频没有声音
-5008	声音识别失败
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动失败
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小320 * 480
-5801	请求缺少身份证号码或身份证姓名
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效
-5805	用户未输入图像或者 url 下载失败
-5806	身份证号码或者身份证姓名格式错误
-5807	查询身份证信息错误

更多其他 API 错误码请查看 **错误码说明** 。

活体检测—获取唇语验证码

最近更新时间：2019-01-21 15:43:18

接口描述

接口请求域名： https://recognition.image.myqcloud.com/face/livegetfour
本接口（livegetfour）用于获取一个唇语验证字符串，用于录制视频，进行活体检测。

⚠ 注意：

- 本接口支持 HTTPS 协议，如果您现在使用的是 HTTP 协议，为了保障您的数据安全，请切换至 HTTPS。
- 如果开发者使用的是原域名（service.image.myqcloud.com），为获得更好的体验，请及时切换到以上域名。

请求头 header

参数名	必选	值	描述
host	是	recognition.image.myqcloud.com	腾讯云人脸核身服务器域名。
content-length	否	包体总长度	建议视频的大小限制为3MB 以下。
content-type	是	application/json 或 multipart/form-data	据不同接口选择： 1. 使用 application/json 格式，参数为 url，其值为图片的 url。 2. 使用 multipart/form-data 格式，参数为 image，其值为图片的 base64 。
authorization	是	鉴权签名	用于 **鉴权** 的签名。

⚠ 注意：

如选择 multipart/form-data，请使用 HTTP 框架/库推荐的方式设置请求的 content-type，不推荐直接调用 setHeader 等方法设置，否则可能导致 boundary 缺失引起请求失败。

输入参数

使用 application/json 格式：

参数名	必选	类型	说明
appid	是	String	接入项目的唯一标识，可在 账号信息 或 云 API 密钥 中查看。
seq	否	String	标识请求序列号。

输出参数

字段	类型	说明
data.validate_data	String	唇语验证字符串
code	Int	错误码
message	String	错误描述

示例

输入示例

```
POST /face/livegetfour HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: recognition.image.myqcloud.com
Content-Length: 66
Content-Type: application/json

{
"appid":"123456"
}
```

输出示例

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 114
Content-Type: application/json
```

```
{
"data":{
"validate_data": "9532",
},
"code":0,
"message":"OK"
}
```


错误码

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 APPID/Bucket 与操作目标不匹配
9	签名过期
10	APPID 不存在
11	SecretId 不存在
12	APPID 和 SecretId 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
24	无权限，未申请服务
107	鉴权服务不可用
108	鉴权服务不可用

错误码	含义
213	内部错误
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与Card照片相似度计算失败
-4011	照片解码失败
-4012	照片人脸检测失败
-4015	自拍照人脸检测失败
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效，上传文件不符合视频要求
-5002	唇语失败
-5005	自拍照解析照片不足，视频里检测到的人脸较少
-5007	视频没有声音
-5008	语音识别失败，视频里的人读错数字
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动检测失败，视频里的人嘴巴未张开或者张开幅度小
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小 270 * 480
-5016	视频里的人不是活体（翻拍等攻击）

错误码	含义
-5801	请求缺少身份证号码或身份证姓名
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效
-5805	用户未输入图像或者 url 下载失败

更多其他 API 错误码请查看 **错误码说明** 。

唇语活体检测视频身份信息核验

最近更新时间：2018-12-18 15:57:27

接口描述

接口请求域名： https://recognition.image.myqcloud.com/face/idcardlivedetectfour
本接口（idcardlivedetectfour）用于判断录制的唇语视频中人物是否为真人（活体检测），同时判断唇语视频中的人脸与身份证号和姓名对应的登记照的人脸相似度，即判断视频中的人与身份证上的人是否为同一人。

⚠ 注意：

- 本接口支持 HTTPS 协议，如果您现在使用的是 HTTP 协议，为了保障您的数据安全，请切换至 HTTPS。
- 如果开发者使用的是原域名（service.image.myqcloud.com），为获得更好的体验，请及时切换到以上域名。

请求头 header

参数名	必选	值	描述
host	是	recognition.image.myqcloud.com	腾讯云人脸核身服务器域名。
content-length	否	包体总长度	建议视频的大小为3MB 以下。
content-type	是	multipart/form-data	仅支持上传本地视频文件。
authorization	是	鉴权签名	用于 **鉴权** 的签名。

⚠ 注意：

选择 multipart/form-data，请使用 HTTP 框架/库推荐的方式设置请求的 content-type，不推荐直接调用 setHeader 等方法设置，否则可能导致 boundary 缺失引起请求失败。

输入参数

使用 multipart/form-data 格式：

参数名	必选	类型	说明
appid	是	String	接入项目的唯一标识，可在 账号信息 或 云 API 密钥 中查看。
validate_data	是	String	livegetfour 得到的唇语验证数据。
video	是	Binary	录制的视频（建议对视频进行压缩处理，减少视频大小有助于减少上传时间和节约流量）。
idcard_number	是	String	用户身份证号码。
idcard_name	是	String	用户身份证姓名（中文，请注意使用 UTF-8编码）。
seq	否	String	标识请求的序列号。

⚠ 注意：

录制的视频里，用户须清晰念出 livegetfour 得到的唇语验证数字。

输出参数

字段	类型	说明
data.live_status	Int	活体检测错误码，非 0 值为出错。
data.live_msg	String	活体检测错误描述。
data.compare_status	Int	人脸对比检测错误码，非 0 值为出错。
data.compare_msg	String	人脸对比错误描述。
data.sim	Int	相似度，取值范围 [0, 100]，推荐相似度大于70时可判断为同一人，可根据具体场景自行调整阈值（阈值 70 的误通过率为千分之一，阈值 80 的误通过率是万分之一）。
data.video_photo	String	视频中的一张 sim 值最大的图像，base64 编码。
code	Int	错误码。
message	String	错误描述。

⚠ 注意：

此处分为两个流程：活体检测和人脸比对。其中 live_status 表示活体检测成功与否，compare_status 表示对比成功与否，只有 compare_status为0时，才有 sim 和 video_photo 字段。
您在使用本接口时，需要先判断活体检测是否成功（ live_status ），再判断人脸比对是否成功（ compare_status ），最后判断 sim 是否大于阈值（如 sim > 75 ）。

示例

输入示例

```
POST /face/idcardlivedetectfour HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: service.image.myqcloud.com
Content-Length: 641
Content-Type: multipart/form-data;boundary=----------acebdf13572468

----------acebdf13572468
Content-Disposition: form-data; name="appid";

123456
----------acebdf13572468
Content-Disposition: form-data; name="validate_data";

9532
----------acebdf13572468
Content-Disposition: form-data; name="idcard_number";

110110199909090909
----------acebdf13572468
Content-Disposition: form-data; name="idcard_name";

张三
----------acebdf13572468
Content-Disposition: form-data; name="video"; filename="video.flv"

yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy
----------acebdf13572468--
```

输出示例

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 168
```

```
Content-Type: application/json

{
"data":{
"live_status":0,
"live_msg":"OK",
"compare_status":0,
"compare_msg":"OK",
"sim":90,
"video_photo":"xxxxxxxxxxxxxx",
},
"code":0,
"message":"OK"
}
```

错误码

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 APPID/Bucket 与操作目标不匹配
9	签名过期
10	APPID 不存在
11	SecretId 不存在
12	APPID 和 SecretId 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数

错误码	含义
23	请求包体过大
24	无权限，未申请服务
107	鉴权服务不可用
108	鉴权服务不可用
213	内部错误
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与 Card 照片相似度计算失败
-4011	照片解码失败
-4012	照片人脸检测失败
-4015	自拍照人脸检测失败
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效，上传文件不符合视频要求
-5002	唇语失败
-5005	自拍照解析照片不足，视频里检测到的人脸较少
-5007	视频没有声音
-5008	语音识别失败，视频里的人读错数字
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动检测失败，视频里的人嘴巴未张开或者张开幅度小
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大

错误码	含义
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小 270 * 480
-5016	视频里的人不是活体（翻拍等攻击）
-5801	请求缺少身份证号码或身份证姓名
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效（因户口变动，极少数正常用户也会返回这个结果，此类用户无法使用本服务）
-5805	用户未输入图像或者 url 下载失败
-5809	用户身份证登记照无效（一般是派出所上传照片时出错，重新拍照片到身份证所属派出所提交可解决）
-5817	证件照质量较差，请到户籍所在地派出所进行核实

更多其他 API 错误码请查看 **错误码说明** 。

活体检测视频与用户照片的对比

最近更新时间：2018-12-14 17:31:18

接口描述

接口请求域名： https://recognition.image.myqcloud.com/face/livedetectfour
本接口（livedetectfour）用于判断录制的唇语视频中人物是否为真人（活体检测），同时判断唇语视频中的人脸与给定的一张人脸照片的人脸相似度，即判断视频中的人与给定一张照片的人是否为同一人。

⚠ 注意：

- 本接口支持 HTTPS 协议，如果您现在使用的是 HTTP 协议，为了保障您的数据安全，请切换至 HTTPS。
- 如果开发者使用的是原域名（service.image.myqcloud.com），为获得更好的体验，请及时切换到以上域名。

请求头 header

所有请求都要求含有下表列出的头部信息：

参数名	必选	值	描述
host	是	recognition.image.myqcloud.com	腾讯云人脸核身服务器域名。
content-length	否	包体总长度	建议视频的大小为3MB 以下。
content-type	是	multipart/form-data	仅支持上传本地视频文件。
authorization	是	鉴权签名	用于 **鉴权** 的签名。

⚠ 注意：

选择 multipart/form-data，请使用 HTTP 框架/库推荐的方式设置请求的 content-type，不推荐直接调用 setHeader 等方法设置，否则可能导致 boundary 缺失引起请求失败。

输入参数

使用 multipart/form-data 格式：

参数名	必选	类型	说明
appid	是	String	接入项目的唯一标识，可在 账号信息 或 云 API 密钥 中查看。
validate_data	是	String	livegetfour 得到的唇语验证数据。
video	是	Binary	录制的视频（建议对视频进行压缩处理，减少视频大小有助于减少上传时间和节约流量）。
compare_flag	是	Bool	video 中的照片和 Card 做对比： 1. 在设置 form-data 数据时设置为字符串"true"、"false"。 2. 参数为 True 时做对比，card 是对比的照片，会返回活体检测和对比的结果。 3. 参数为 False 时只做活体检测，card 字段可不填，只返回活体检测的结果。
card	否	Binary	需要检测的另一张照片。
seq	否	String	标识请求的序列号。

⚠ 注意：

录制的视频里，用户须缓慢清晰念出 livegetfour 得到的唇语验证数字。

输出参数

字段	类型	说明
data.live_status	Int	返回活体检测错误码，非0值为出错。
data.live_msg	String	返回错误描述。
data.compare_status	Int	返回人脸对比检测错误码，非0值为出错(compare_flag 是 True 的时候才返回)。
data.compare_msg	String	返回错误描述(compare_flag 是 True 的时候才返回)。
data.sim	Int	相似度，取值范围 [0, 100]，推荐相似度大于70时可判断为同一人，可根据具体场景自行调整阈值（阈值70的误通过率为千分之一，阈值80的误通过率是万分之一）(compare_flag 是 True 并且识别成功的时候才返回)。

字段	类型	说明
data.photo	String	返回相似度最高的 video 中的一张图像(compare_flag 是 True 并且识别成功的时候才返回， base64编码)。
code	Int	错误码。
message	String	错误描述。

注意：
此处分为两个流程：活体检测和人脸比对。其中live_status表示活体检测成功与否，compare_status表示对比成功与否，只有 compare_status为 0 时，才有 sim 和 photo 字段。您在使用本接口时，需要先判断活体检测是否成功（ live_status ），再判断人脸比对是否成功（ compare_status ），最后判断 sim 是否大于阈值（如 sim > 75 ）。

示例

输入示例

```
POST /face/livedetectfour HTTP/1.1
Authorization: FCHXdPTEwMDAwMzc5Jms9QUtJRGVRZDBrRU1yM2J4ZjhRckJi==
Host: service.image.myqcloud.com
Content-Length: 641
Content-Type: multipart/form-data;boundary=----------acebdf13572468

----------acebdf13572468
Content-Disposition: form-data; name="appid";

123456
----------acebdf13572468
Content-Disposition: form-data; name="validate_data";

9532
----------acebdf13572468
Content-Disposition: form-data; name="compare_flag";

true
----------acebdf13572468
Content-Disposition: form-data; name="video"; filename="video.flv"
```

```
yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy
----------acebdf13572468
Content-Disposition: form-data; name="card"; filename="face.jpg"
Content-Type: image/jpeg

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------acebdf13572468--
```

输出示例

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 168
Content-Type: application/json

{
"data":{
"live_status":0,
"live_msg":"OK",
"compare_status":0,
"compare_msg":"OK",
"sim":90,
"photo":"xxxxxxxxxxxxxx",
},
"code":0,
"message":"OK"
}
```

错误码

错误码	含义
3	错误的请求；其中 message:account abnormal,errorno is:2 为账号欠费停服
4	签名为空
5	签名串错误
6	签名中的 APPID/Bucket 与操作目标不匹配
9	签名过期
10	APPID 不存在

错误码	含义
11	SecretId 不存在
12	APPID 和 SecretId 不匹配
13	重放攻击
14	签名校验失败
15	操作太频繁，触发频控
16	Bucket 不存在
21	无效参数
23	请求包体过大
24	无权限，未申请服务
107	鉴权服务不可用
108	鉴权服务不可用
213	内部错误
-4006	视频中自拍照特征提取失败
-4007	视频中自拍照之间对比失败
-4009	Card 照片提取特征失败
-4010	自拍照与 Card 照片相似度计算失败
-4011	照片解码失败
-4012	照片人脸检测失败
-4015	自拍照人脸检测失败
-4016	自拍照解码失败
-4017	Card 照片人脸检测失败
-4018	Card 照片解码失败
-5001	视频无效，上传文件不符合视频要求
-5002	唇语失败

错误码	含义
-5005	自拍照解析照片不足，视频里检测到的人脸较少
-5007	视频没有声音
-5008	语音识别失败，视频里的人读错数字
-5009	视频人脸检测失败，没有嘴或者脸
-5010	唇动检测失败，视频里的人嘴巴未张开或者张开幅度小
-5011	活体检测失败(活体其他错误都归类到里面)
-5012	视频中噪声太大
-5013	视频里的声音太小
-5014	活体检测 level 参数无效
-5015	视频像素太低，最小 270 * 480
-5016	视频里的人不是活体（翻拍等攻击）
-5801	请求缺少身份证号码或身份证姓名
-5802	服务器内部错误，服务暂时不可用
-5803	身份证姓名与身份证号码不一致
-5804	身份证号码无效（因户口变动，极少数正常用户也会返回这个结果，此类用户无法使用本服务）
-5805	用户未输入图像或者 url 下载失败

更多其他 API 错误码请查看 **错误码说明** 。