

文档版本号：V1.1.0-20170119

**佳都科技**

**视频云+大数据应用平台**

**V1.1.0**

**技战法接口文档**

**佳都新太科技股份有限公司**

**二零一七年一月**

**目录**

[保密申明 2](#_Toc470785313)

[1 前言 3](#_Toc470785314)

[1.1 概述 3](#_Toc470785315)

[1.2 适用范围 3](#_Toc470785316)

[1.3 术语 3](#_Toc470785317)

[1.4 参考文档 3](#_Toc470785318)

[2 API使用说明 4](#_Toc470785319)

[2.1 返回值结构 4](#_Toc470785320)

[2.2 返回码 4](#_Toc470785321)

[3 准备工作 6](#_Toc470785322)

[3.1 环境准备 6](#_Toc470785323)

[3.2 需要的包 6](#_Toc470785324)

[3.3 初始化工作 6](#_Toc470785325)

[3.4 使用接口代码示例 7](#_Toc470785326)

[3.5 已提供接口 8](#_Toc470785327)

[4 车辆技战法 10](#_Toc470785328)

[4.1 首次入城技战法接口 10](#_Toc470785329)

[4.1.1 参数 10](#_Toc470785330)

[4.1.2 返回 11](#_Toc470785331)

[4.2 频繁过车技战法接口 11](#_Toc470785332)

[4.2.1 参数 11](#_Toc470785333)

[4.2.2 返回 12](#_Toc470785334)

[4.3 区域碰撞技战法接口 13](#_Toc470785335)

[4.3.1 参数 13](#_Toc470785336)

[4.3.2 返回 14](#_Toc470785337)

[4.4 同行车分析技战法接口 14](#_Toc470785338)

[4.4.1 参数 14](#_Toc470785339)

[4.4.2 返回 15](#_Toc470785340)

[4.5 轨迹分析技战法接口 16](#_Toc470785341)

[4.5.1 参数 16](#_Toc470785342)

[4.5.2 返回 17](#_Toc470785343)

[4.6 昼伏夜出技战法接口 17](#_Toc470785344)

[4.6.1 参数 17](#_Toc470785345)

[4.6.2 返回 18](#_Toc470785346)

[4.7 一牌多车技战法接口 19](#_Toc470785347)

[4.7.1 参数 19](#_Toc470785348)

[4.7.2 返回 20](#_Toc470785349)

[4.8 落脚点分析技战法接口 20](#_Toc470785350)

[4.8.1 参数 20](#_Toc470785351)

[4.8.2 返回 21](#_Toc470785352)

[4.9 隐匿车辆分析技战法接口 23](#_Toc470785353)

[4.9.1 参数 23](#_Toc470785354)

[4.9.2 返回 23](#_Toc470785355)

[4.10 相似车分析技战法接口 24](#_Toc470785356)

[4.10.1 参数 24](#_Toc470785357)

[4.10.2 返回 25](#_Toc470785358)

[5 人脸检索与分析 26](#_Toc470785359)

[5.1 汽车驾驶员人脸检索接口 26](#_Toc470785360)

[5.1.1 参数 26](#_Toc470785361)

[5.1.2 返回 27](#_Toc470785362)

[5.2 非汽车驾驶员人脸检索接口 27](#_Toc470785363)

[5.2.1 参数 27](#_Toc470785364)

[5.2.2 返回 28](#_Toc470785365)

[5.3 人脸抓拍检索接口 29](#_Toc470785366)

[5.3.1 参数 29](#_Toc470785367)

[5.3.2 返回 30](#_Toc470785368)

[5.4 串并案分析接口 30](#_Toc470785369)

[5.4.1 参数 30](#_Toc470785370)

[5.4.2 返回 31](#_Toc470785371)

[5.5 黑车分析 32](#_Toc470785372)

[5.5.1 参数 32](#_Toc470785373)

[5.5.2 返回 32](#_Toc470785374)

[5.6 人脸1:1比对 33](#_Toc470785375)

[5.6.1 通过人脸图片地址比对 33](#_Toc470785376)

[5.6.1.1 描述 33](#_Toc470785377)

[5.6.1.2 参数 33](#_Toc470785378)

[5.6.1.3 返回 34](#_Toc470785379)

[5.6.1.4 返回示例 34](#_Toc470785380)

[5.6.2 通过人脸特征值比对 34](#_Toc470785381)

[5.6.2.1 描述 34](#_Toc470785382)

[5.6.2.2 参数 34](#_Toc470785383)

[5.6.2.3 返回 34](#_Toc470785384)

[5.6.2.4 返回示例 35](#_Toc470785385)

[5.6.3 通过检索后比对 35](#_Toc470785386)

[5.6.3.1 描述 35](#_Toc470785387)

[5.6.3.2 参数 35](#_Toc470785388)

[5.6.3.3 返回 35](#_Toc470785389)

[5.6.3.4 返回示例 36](#_Toc470785390)

[6 静态小库 37](#_Toc470785391)

[6.1 相关说明 37](#_Toc470785392)

[6.2 创建静态小库 37](#_Toc470785393)

[6.2.1 创建空的静态小库 37](#_Toc470785394)

[6.2.1.1 描述 37](#_Toc470785395)

[6.2.1.2 参数 37](#_Toc470785396)

[6.2.1.3 返回 37](#_Toc470785397)

[6.2.1.4 返回示例 37](#_Toc470785398)

[6.2.2 根据条件创建小库 38](#_Toc470785399)

[6.2.2.1 描述 38](#_Toc470785400)

[6.2.2.2 参数 38](#_Toc470785401)

[6.2.2.3 返回 39](#_Toc470785402)

[6.2.2.4 返回示例 39](#_Toc470785403)

[6.3 判断静态小库是否存在 39](#_Toc470785404)

[6.3.1 描述 39](#_Toc470785405)

[6.3.2 参数 39](#_Toc470785406)

[6.3.3 返回 40](#_Toc470785407)

[6.3.4 返回示例 40](#_Toc470785408)

[6.4 删除静态小库 40](#_Toc470785409)

[6.4.1 描述 40](#_Toc470785410)

[6.4.2 参数 40](#_Toc470785411)

[6.4.3 返回 40](#_Toc470785412)

[6.4.4 返回示例 40](#_Toc470785413)

[6.5 注册人脸到静态小库 41](#_Toc470785414)

[6.5.1 传参方式注册人脸 41](#_Toc470785415)

[6.5.1.1 描述 41](#_Toc470785416)

[6.5.1.2 参数 41](#_Toc470785417)

[6.5.1.3 返回 41](#_Toc470785418)

[6.5.1.4 返回示例 41](#_Toc470785419)

[6.5.2 传条件方式注册人脸 42](#_Toc470785420)

[6.5.2.1 描述 42](#_Toc470785421)

[6.5.2.2 参数 42](#_Toc470785422)

[6.5.2.3 返回 43](#_Toc470785423)

[6.5.2.4 返回示例 43](#_Toc470785424)

[6.6 从静态小库注销人脸 43](#_Toc470785425)

[6.6.1 描述 43](#_Toc470785426)

[6.6.2 参数 44](#_Toc470785427)

[6.6.3 返回 44](#_Toc470785428)

[6.6.4 返回示例 44](#_Toc470785429)

[6.7 静态小库人脸1:N检索 44](#_Toc470785430)

[6.7.1 描述 44](#_Toc470785431)

[6.7.2 参数 44](#_Toc470785432)

[6.7.3 返回 46](#_Toc470785433)

[6.7.4 返回示例 46](#_Toc470785434)

[6.8 静态小库人脸M:N比对 46](#_Toc470785435)

[6.8.1 描述 46](#_Toc470785436)

[6.8.2 参数 46](#_Toc470785437)

[6.8.3 返回 47](#_Toc470785438)

[6.8.4 返回示例 48](#_Toc470785439)

文档控制页

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | 版本 | 时间 | 编写人 |
| 创建 | V1.0.001 | 2016年12月23日 | 欧仕华 |
| 增加通过条件注册人脸 | V1.0.002 | 2016年12月26日 | 欧仕华 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 保密申明

本文档内资料，用于佳都科技内部培训、施工、交流参考指引。未经同意。任何人不得以任何方式向第三方组织、机构、人员直接或间接拷贝、公布信息内容，亦不得将文档信息用于商业目的以外的其他用途，佳都科技保留一切权力。

# 前言

## 概述

本文描述了如何使用封装的技战法、人脸检索和静态小库相关接口。

## 适用范围

本文档适用于视频云+大数据应用平台中技战法、人脸检索、静态小库功能模块的开发。

## 术语

|  |  |  |
| --- | --- | --- |
| **术语** | **英文全称** | **解释** |
| ES | elasticsearch | 基于lucene的全文搜索引擎 |
| alluxio | alluxio | 分布式内存文件系统 |
| spark | spark | 分布式内存计算框架 |
| \_id | 文档id | ES索引中每条记录的唯一id，为字符串类型 |

## 参考文档

① 《视频云+技战法实现逻辑.docx》

② 《视频云+技战法数据字典.docx》

# API使用说明

## 返回值结构

以结果对象的方式返回，结果对象结构如下：

|  |
| --- |
| {  "code": 0  "message": ""  "list":[{...},{...}]  } |

目前使用com.suntek.tactics.common.CollisionResult对象来保存结果，可通过的get方法分别获得code、message、list参数的值，也可通过toJson方法把结果转为JSON字符串。

如无特别说明, 每次请求的返回值中, 都会包含以上三个参数。

每个参数说明如下：

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| code | int | 返回码，0表示成功，其它值表示失败。具体含义可以参考2.2节 |
| message | String | 分析结果描述 |
| list | List<Map<String, Object> | 分析结果列表  注：如接口实现时返回结果格式于此不一致，以接口定义的结构为准 |

## 返回码

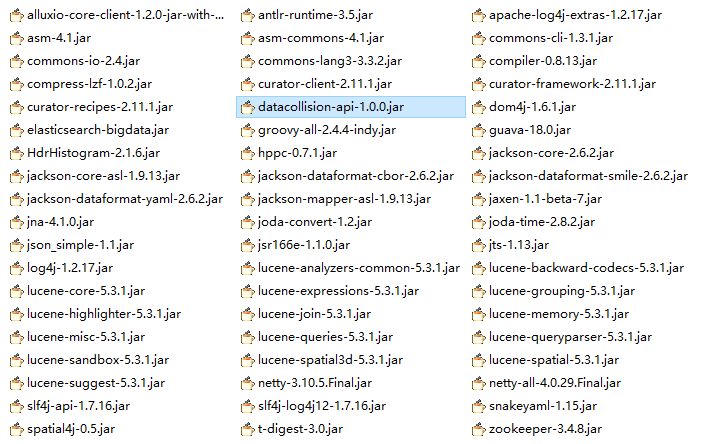
|  |  |
| --- | --- |
| **返回码** | **描述** |
| 0 | 正常分析完成 |
| 10009 | 缺少必要参数，或者参数值格式不正确 |
| 10012 | 传入的ElasticSearch客户端为null |
| 10013 | 查询条数超过最大限制条数 |
| 10014 | 任务执行错误 |
| 10015 | 系统资源繁忙，如新增分析任务时超出任务队列最大长度 |
| 12000 | 创建小库时记录条数超过最大限制条数错误 |
| 12001 | 批量注册人脸时全部人脸特征不合法 |
| 12002 | 注册人脸到小库时记录条数超过最大限制条数错误 |
| 12003 | zookeeper无对应记录错误 |
| 12004 | 更新zookeeper记录失败错误 |
|  |  |
|  |  |
|  |  |

# 准备工作

## 环境准备

首先需要正确安装ElasticSearch、Alluxio、Spark环境并正常启动集群。

## 需要的包



其中：**datacollision-api-1.0.0.jar** 包含了车辆技战法和人脸检索接口的实现。

## 初始化工作

在应用启动时（如tomcat启动时），需调用如下方法进行初始化工作：

AlluxioClient.init("alluxio://10.4.100.143:19998/dc");

VehicleOperationManager.init(4, 4); // 初始化车辆技战法任务队列

FaceOperationManager.init(2, 2); // 初始化人脸检索任务队列

其中第一行的10.4.100.143为Alluxio集群中Master节点的IP，根据实际情况调整；第二行的参数列表意义如下（一个任务会启动一个线程去完成）：

* poolSize

当前运行的最大任务数

* poolQueueSize

在等待队列中最大任务数

## 使用接口代码示例

以“频繁过车技战法”为例子。

|  |
| --- |
| **import com.suntek.tactics.common.CollisionResult;**  **import com.suntek.tactics.manager.VehicleOperationManager;**  **import com.suntek.tactics.api.TacticsEnum;**  public static void main(String[] args) throws Exception {  **// 创建ES客户端，目前车辆技战法使用的es集群名为bigdata，ip为10.4.100.141~144，端口为9300**  **// 人脸检索使用的es集群名为face\_search\_bigdata，ip为10.4.100.145，端口为9301**  Settings settings = Settings.settingsBuilder().put("cluster.name", "bigdata").build();  TransportClient client = TransportClient.builder().settings(settings).build();  client.addTransportAddress(  new InetSocketTransportAddress(InetAddress.getByName(ip), 9300));  // 构造技战法所需参数  Map<String, Object> map = new HashMap<String, Object>();  map.put("beginTime", "2016-01-08 00:00:00");  map.put("endTime", "2016-01-09 00:00:00");  map.put("accessFrequency", 3);  map.put("cross", "44010000001310000046,44010000001310000366");  map.put("vehicleLogo", "0,1,2,3,4,5");  map.put("topN", 20);  // 调用技战法  **CollisionResult result =**  **VehicleOperationManager.runOperation(map, client, TacticsEnum.FREQUENTACCESS);**  // 获取返回结果  int code = **result.getCode();**  String message = **result.getMessage();**  List<Map<String, Object>> resultMap = **result.getList();**  // 获得JSON格式的返回结果  String jsonString = **result.toJson();**  } |

## 已提供接口

其中：TacticsEnum枚举类包含如下技战法：

**1）车辆技战法**

* FIRSTACCESS 首次入城技战法
* FREQUENTACCESS 频繁过车技战法
* REGIONCOLLSION 区域碰撞技战法
* FOLLOWCAR 同行车分析技战法
* DAYHIDENIGHTACTIVE 昼伏夜出技战法
* HIDDENVEHICLE 隐匿车辆分析技战法
* SIMILARVEHICLE 相似车分析技战法
* ONEPLATEMULTIVEHICLE 一牌多车分析技战法
* TRAJECTORYANALYSIS 轨迹分析技战法
* FOOTHOLDANALYSIS 落脚点分析技战法

**2）人脸检索**

* DRIVER\_FACE\_SEARCH 汽车驾驶人人脸检索
* NON\_MOTOR\_FACE\_SEARCH 非汽车驾驶人人脸检索
* FACE\_DETECT\_SEARCH 人脸抓拍人脸检索
* CASE\_RELATE 串并案分析技战法
* ILLEGAL\_VEHICLE 黑车分析技战法
* FACE\_ONE2ONE\_COMPARE 人脸1:1比对
* FACE\_N2N 人脸N:N比对

**3）人脸静态小库**

* FACEDB\_CREATE 创建人脸静态小库
* FACEDB\_ISEXIST 判断人脸静态小库是否存在
* FACEDB\_DELETE 删除人脸静态小库
* FACEDB\_REGISTER 注册人脸到静态小库
* FACEDB\_LOGOUT 在人脸静态小库注销人脸
* FACEDB\_ONE2N\_SEARCH 在人脸静态小库进行1:N检索
* FACEDB\_M2N\_SEARCH 人脸静态小库人脸M:N检索

# 车辆技战法

## 首次入城技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表4.1所示：

表4.1 首次入城技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| backDate | 回溯天数 | int | √ | 取值范围为[1,30] |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表4.2所示：

表4.2 首次入城技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| KKBH | 卡口编号 | String | 卡口id |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |

对于一个车牌在查询时间段内的所有过车记录，取第一次出现的记录返回

## 频繁过车技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表4.3所示：

表4.3 频繁过车技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| accessFrequency | 出入次数 | int | √ | 必须大于1 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表4.4所示：

表4.4 频繁过车技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| ACCESSFREQUENCY | 过车次数 | int | 满足查询条件的该车过车次数 |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |
| IDS | \_id集合 | String | 满足查询条件的该车过车记录的\_id，多个\_id用英文逗号分割，如："110559896386142336,110560454694142028" |

## 区域碰撞技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表4.5所示：

表4.5区域碰撞技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| timeRegionList | 时间段区域对象集合 | List<Map<String, Object>> | √ | List中包含多个“时间段区域对象”， “时间段区域对象”定义参考表4.6 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

表4.6 时间段区域对象定义

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表4.7所示：

表4.7 区域碰撞技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |
| IDS | \_id集合 | String | 满足查询条件的该车过车记录的\_id，多个\_id用英文逗号分割，如："110559896386142336,110560454694142028" |

## 同行车分析技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表7.1所示：

表7.1 同行车分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| plate | 车牌号码 | String | √ | 车牌号码 |
| followTime | 同行次数 | int | √ | 必须大于0 |
| followMinute | 同行分钟数 | int | √ | 必须大于0 |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| cross | 卡口 | String | × | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表7.2所示：

表7.2 同行车分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| TIMES | 同行次数 | int | 同行次数 |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |
| IDS | \_id集合 | String | 满足查询条件的该车过车记录的\_id，多个\_id用英文逗号分割，如："110559896386142336,110560454694142028" |

## 轨迹分析技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表8.1所示：

表8.1 轨迹分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| plate | 车牌号码 | int | √ | 车牌号码 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表8.2所示：

表8.2 轨迹分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| KKBH | 经过卡口 | String | 卡口id |

## 昼伏夜出技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表9.1所示：

表9.1 昼伏夜出技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| dayBeginTime | 白天开始时间 | String | √ | HH:mm:ss，必须小于dayEndTime |
| dayEndTime | 白天结束时间 | String | √ | HH:mm:ss，必须大于dayBeginTime |
| nightBeginTime | 夜晚开始时间 | String | √ | HH:mm:ss，支持跨凌晨0点的时间段 |
| nightEndTime | 夜晚结束时间 | String | √ | HH:mm:ss，支持跨凌晨0点的时间段 |
| dayNum | 白天出现次数 | int | √ | 出现在白天的最多次数 |
| nightNum | 夜晚出现次数 | int | √ | 出现在夜晚的最少次数 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表9.2所示：

表9.2 昼伏夜出技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| DAYNUM | 白天出现次数 | int | 白天出现次数 |
| NIGHTNUM | 夜晚出现次数 | int | 夜晚出现次数 |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |
| IDS | \_id集合 | String | 满足查询条件的该车过车记录的\_id，多个\_id用英文逗号分割，如："110559896386142336,110560454694142028" |

## 一牌多车技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表10.1所示：

表10.1 一牌多车技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| plate | 车牌号码 | String | √ | 精确车牌号码 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| feature | 特征 | String | × | 用于判断是否同一车辆特征的依据，至少传入一个字段名。若不传，则会根据品牌和子品牌来认定一个车牌的车辆特征，多个字段名用英文逗号分割，如："PPDM,ZPPDM,CLLX"，其中目前CSYS（车身颜色）是默认一定使用的车辆特征。 |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表10.2所示：

表10.2 一牌多车技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| FEATURE | 特征的值 | String | 多个特征的值使用英文逗号连接而成，特征顺序与输入参数中的featureList顺序一致，若featureList为null或空，则该值为：  品牌代号,子品牌代号 |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |
| IDS | \_id集合 | String | 满足查询条件的该车过车记录的\_id，多个\_id用英文逗号分割，如："110559896386142336,110560454694142028" |

## 落脚点分析技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表11.1所示：

表11.1 落脚点分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| plate | 车牌号码集合 | String | √ | 每个精确车牌用英文逗号分割，表示可满足多个值，如："粤A00000,粤B00000" |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表11.2所示：

表11.2 落脚点分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| DATE | 日期 | String | yyyy-MM-dd |
| LINES | 路径列表 | List<Map<String, Object>> | 每个经过的卡口间的路径，如：  [  {  "BEGINPLATE" : "粤A00000",  "BEGINCROSS" : "4401...13101",  "BEGINTIME" : "160101225959",  "ENDPLATE" : "粤A00001",  "ENDCROSS" : "4401...13102",  "ENDTIME" : "160101235959",  "USETIME" : "3600"  }  ……  ] |
| CROSSSET | 卡口集合 | List<String> | 经过次数最多的卡口集合 |
| MAXFREQ | 最大出现次数 | int | 所有卡口出现次数中的最大值，单位：次 |
| LINESET | 路径集合 | List<Map<String, Object>> | 用时最长的路径集合，结构同 "LINES" |
| MAXUSETIME | 最大用时 | int | 所有路径的用时中的最大值，单位：秒 |

页面大概展示效果：

|  |  |  |
| --- | --- | --- |
| 经过次数 最多的卡口 | B | 5次 |
| 停留时间 最长的路径 | B -> A | 40min |
|  |  |  |
| 2016.01.01 | 路径 | 用时 |
| A -> B | 10min |
| B -> A | 40min |
| A -> C | 20min |
| C -> B | 20min |
| 经过次数 最多的卡口 | B | 3次 |
| 停留时间 最长的路径 | B -> A | 40min |
|  |  |  |
| 2016.01.02 | 路径 | 用时 |
| A -> B | 10min |
| B -> C | 30min |
| 经过次数 最多的卡口 | B | 2次 |
| 停留时间 最长的路径 | B -> C | 30min |

## 隐匿车辆分析技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表12.1所示：

表12.1 隐匿车辆分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| crossIn | 案发区域卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| crossOut | 挖掘区域卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表12.2所示：

表12.2 隐匿车辆分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| KKBH | 经过卡口 | String | 卡口id |
| ADDRESS | 卡口地址 | String | 卡口地址 |
| LONGITUDE | 经度 | double | 经度 |
| LATITUDE | 纬度 | double | 纬度 |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |

## 相似车分析技战法接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表13.1所示：

表13.1 相似车分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| plate | 车牌号码 | String | √ | 车牌号码 |
| diffSize | 相差位数 | int | √ | 1或2，如粤A0000和粤A1000相差1位,粤A0000和粤B1000相差2位 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表13.2所示：

表13.2 相似车分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| KKBH | 经过卡口 | String | 卡口id |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |

对于一个车牌在查询时间段内的所有过车记录，取最后出现的记录返回

# 人脸检索与分析

## 汽车驾驶员人脸检索接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.1所示：

表5.1 汽车驾驶员检索API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss，大于等于该时间戳 |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss，小于等于该时间戳 |
| 若设置了以下四个参数，则会使用经过日期和经过时间两个条件来检索数据  此时beginTime和endTime参数无效，可以不传 | | | | |
| beginYMD | 开始年月日 | String | √ | yyyy-MM-dd，大于等于该日期 |
| endYMD | 结束年月日 | String | √ | yyyy-MM-dd，小于等于该日期 |
| beginHMS | 开始时分秒 | String | √ | HH:mm:ss，大于等于该时分秒 |
| endHMS | 结束时分秒 | String | √ | HH:mm:ss，小于等于该时分秒 |
|  | | | | |
| url | 目标人脸图片URL | String | √ | 目标人脸图片URL |
| similarity | 相似度阈值 | int | √ | 取值范围：[0, 100]  只返回相似度大于或等于该值的结果 |
| topN | 返回结果条数 | int | √ | 最多返回结果条数 |
| cross | 卡口 | String | × | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleSubBrand | 子品牌代码 | String | × |
| vehicleColor | 车辆颜色 | String | × |
| plateColor | 车牌颜色 | String | × |
| sex | 性别 | String | × |
| age | 年龄段 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表5.2所示：

表5.2 汽车驾驶员检索API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| SIMILARITY | 相似度 | int | 范围：[0, 100] |
| VEHICLE\_INFO\_ID | 关联ID | long | 本记录关联到的汽车过车记录\_id |
| PIC\_DRIVER | 人脸图片 | String | 图片url |

### 限制

限制输入条件对应的记录总数不超过100万条，若超过，返回超过限制条数提示。

### 返回示例

## 非汽车驾驶员人脸检索接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.3所示：

表5.3非汽车驾驶员检索API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss，大于等于该时间戳 |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss，小于等于该时间戳 |
| 若设置了以下四个参数，则会使用经过日期和经过时间两个条件来检索数据  此时beginTime和endTime参数无效，可以不传 | | | | |
| beginYMD | 开始年月日 | String | √ | yyyy-MM-dd，大于等于该日期 |
| endYMD | 结束年月日 | String | √ | yyyy-MM-dd，小于等于该日期 |
| beginHMS | 开始时分秒 | String | √ | HH:mm:ss，大于等于该时分秒 |
| endHMS | 结束时分秒 | String | √ | HH:mm:ss，小于等于该时分秒 |
|  | | | | |
| url | 目标人脸图片URL | String | √ | 目标人脸图片URL |
| similarity | 相似度阈值 | int | √ | 取值范围：[0, 100]  只返回相似度大于或等于该值的结果 |
| topN | 返回结果条数 | int | √ | 最多返回结果条数 |
| cross | 卡口 | String | × | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleSubBrand | 子品牌代码 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表5.4所示：

表5.4汽车驾驶员检索API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| SIMILARITY | 相似度 | int | 范围：[0, 100] |
| VEHICLE\_INFO\_ID | 关联ID | long | 本记录关联到的非汽车过车记录\_id |
| PIC\_DRIVER | 人脸图片 | String | 图片url |

### 限制

限制输入条件对应的记录总数不超过100万条，若超过，返回超过限制条数提示。

### 返回示例

## 人脸抓拍检索接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.5所示：

表5.5 人脸抓拍检索API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss，大于等于该时间戳 |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss，小于等于该时间戳 |
| 若设置了以下四个参数，则会使用经过日期和经过时间两个条件来检索数据  此时beginTime和endTime参数无效，可以不传 | | | | |
| beginYMD | 开始年月日 | String | √ | yyyy-MM-dd，大于等于该日期 |
| endYMD | 结束年月日 | String | √ | yyyy-MM-dd，小于等于该日期 |
| beginHMS | 开始时分秒 | String | √ | HH:mm:ss，大于等于该时分秒 |
| endHMS | 结束时分秒 | String | √ | HH:mm:ss，小于等于该时分秒 |
|  | | | | |
| url | 目标人脸图片URL | String | √ | 目标人脸图片URL |
| similarity | 相似度阈值 | int | √ | 取值范围：[0, 100]  只返回相似度大于或等于该值的结果 |
| topN | 返回结果条数 | int | √ | 最多返回结果条数 |
| cross | 卡口 | String | × | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表5.6所示：

表5.6人脸抓拍检索API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| INFO\_ID | 记录ID | long | 获取本信息的记录\_id |
| SIMILARITY | 相似度 | int | 范围：[0, 100] |
| FACE\_INFO\_ID | 关联ID | long | 本记录关联到的过车记录\_id |
| DEVICE\_ID | 设备编号 | String | 人脸卡口设备编号 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| OBJ\_PIC | 人脸全景图 | String | 图片url |
| PIC | 人脸抓拍图 | String | 图片url |

### 限制

限制输入条件对应的记录总数不超过100万条，若超过，返回超过限制条数提示。

### 返回示例

## 串并案分析接口

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.7所示：

表5.7 串并案分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| similarity | 相似度阀值 | int | √ | 相似度大于或等于该值的两个人脸才认为是同一个人，此值为经验值 |
| topN | 返回结果条数 | int | × | 最多返回人数 |
| caseType | 案件类型 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："01050200,01050300, 01050400" |
| criminalMeans | 作案手段 | String | × |
| criminalTargets | 作案目标 | String | × |
| criminalPlace | 作案场所 | String | × |
| criminalTime | 案发时间 | String | × |

### 返回

以List<List<Map<String, Object>>>的结构返回，最外层List保存多个人的分析结果（以各个人对应的案件数降序排序），次外层保存单个人对应的多个案件（以该人对应的多个案件的案发时间升序排序），Map保存单个案件的详细信息。其中Map的Key含义描述如表5.8所示：

表5.8 串并案分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| FILE\_ID | 记录ID | long | 获取本条记录信息的\_id |
| CASE\_NAME | 案件名称 | String | 案件名称 |
| CASE\_TYPE\_ID | 案件类型 | String | 案件类型 |
| BEGINTIME | 案发时间 | long | 案发时间 |
| ENDTIME | 案件结束时间 | long | 案件结束时间 |
| CCLJ | 图片存储路径 | String | 图片存储路径 |
| SLTCCLJ | 缩略图存储路径 | String | 缩略图存储路径 |

## 黑车分析

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.9所示：

表5.9 黑车分析技战法API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| beginTime | 开始时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| endTime | 结束时间 | String | √ | yyyy-MM-dd HH:mm:ss |
| backDate | 回溯天数 | int | √ | 必须大于0 |
| cross | 卡口 | String | √ | 每个卡口id用英文逗号分割，表示可满足多个值，如："44010000001311000001,44010000001311000002,44010000001311000003" |
| topN | topN | int | × | 返回的所有分析结果中，前topN条结果会从ES中获取对应的车辆信息，其余的结果则不会，若topN=0，则全部分析结果都会从ES中获取对应的车辆信息，建议设置为一页显示的条数 |
| vehicleLogo | 品牌代码 | String | × | 每个值用英文逗号分割，表示可满足多个值，如："12,33,45" |
| vehicleType | 车辆类型 | String | × |
| vehicleColor | 车身颜色 | String | × |
| plateType | 车牌类型 | String | × |
| plateColor | 车牌颜色 | String | × |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表5.10所示：

表5.10 黑车分析技战法API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| HPHM | 车牌号码 | String | 车牌号码 |
| CSYS | 车身颜色 | int | 颜色代码 |
| PPDM | 品牌代码 | int | 品牌代码 |
| ZPPDM | 子品牌代码 | int | 子品牌代码 |
| JGSK | 经过时刻 | String | yyyy-MM-dd HH:mm:ss |
| KKBH | 卡口编号 | String | 卡口id |
| XXBH | 过车记录\_id | long | 获取本车辆信息的过车记录\_id |

对于一个车牌在查询时间段内的所有过车记录，取第一次出现的记录返回。

## 人脸1:1比对

### 通过人脸图片地址比对

#### 描述

接口名：FACE\_ONE2ONE\_COMPARE

通过传入的两张图片的路径，返回人脸图片比对后的相似度。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.11所示：

表5.11人脸1:1比对API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| urlFrom | 人脸图片URL | String | √ | 人脸图片URL |
| urlTo | 人脸图片URL | String | √ | 人脸图片URL |

#### 返回

以2.1节的结构返回，以code值判断是否比对成功，message记录成功信息或失败原因，其中List中Map的Key含义描述如表5.12所示：

表5.12 人脸1:1比对API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| SIMILARITY | 相似度 | int | 范围：[0 ,100] |

#### 返回示例

{"code":0,"message":"分析完成","list":[{"SIMILARITY":89}]}

### 通过人脸特征值比对

#### 描述

接口名：FACE\_ONE2ONE\_COMPARE

通过传入的两张图片的特征值，返回人脸图片比对后的相似度。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.13所示：

表5.13人脸1:1比对API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| featureFrom | 人脸特征值1 | String | √ | 通过Base64编码后的值 |
| featureTo | 人脸特征值2 | String | √ | 通过Base64编码后的值 |

#### 返回

以2.1节的结构返回，以code值判断是否比对成功，message记录成功信息或失败原因，其中List中Map的Key含义描述如表5.14所示：

表5.14 人脸1:1比对API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| SIMILARITY | 相似度 | int | 范围：[0 ,100] |

#### 返回示例

{"code":0,"message":"分析完成","list":[{"SIMILARITY":89}]}

### 通过检索后比对

#### 描述

接口名：FACE\_ONE2ONE\_COMPARE

通过传入检索条件来检索两个人脸进行比对，获得人脸的相似度。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.15所示：

表5.15人脸1:1比对API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| rltzFieldNameFrom | 人脸特征字段名 | String | √ | 人脸特征字段名 |
| indexNameFrom | 索引名 | String | √ | 索引名 |
| typeNameFrom | 类型名 | String | √ | 索引类型名 |
| idFrom | 文档ID | String | √ | 文档对应的\_id |
| rltzFieldNameTo | 人脸特征字段名 | String | √ | 人脸特征字段名 |
| indexNameTo | 索引名 | String | √ | 索引名 |
| typeNameTo | 类型名 | String | √ | 索引类型名 |
| idTo | 文档ID | String | √ | 文档对应的\_id |

#### 返回

以2.1节的结构返回，以code值判断是否比对成功，message记录成功信息或失败原因，其中List中Map的Key含义描述如表5.16所示：

表5.16 人脸1:1比对API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| SIMILARITY | 相似度 | int | 范围：[0 ,100] |

#### 返回示例

{"code":0,"message":"分析完成","list":[{"SIMILARITY":89}]}

## 人脸N:N接口

### 描述

接口名：FACE\_N2N

对传入的一组或多组查询条件对应的人脸进行N:N比对，根据设定的相似度阈值分析这些照片中共有几个人，并在返回结果中按人来对记录id进行分组。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表5.17所示：

表5.17 人脸N:N接口输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| indexName | 索引名 | String | √ | 人脸特征所在的索引名，多个索引名用英文逗号分割，如：face\_1701,face\_1702 |
| typeName | type名 | String | √ | 人脸特征所在的索引type名 |
| idFieldName | 主键字段名 | String | √ | 常见有INFO\_ID，XXBH，PERSON\_ID等 |
| rltzFieldName | 人脸特征字段名 | String | √ | 一般为RLTZ |
| similarity | 相似度阀值 | int | √ | 相似度大于或等于该值的两个人脸才认为是同一个人，此值为经验值 |
| querys | 布尔查询对象 | List<BoolQueryBuilder> | √ | 对多组查询条件对应的全部人脸汇总来做N:N比对，返回结果中QUERY\_IDS对应的多个列表顺序与此参数中的布尔查询对象顺序一致 |

### 返回

以2.1节的结构返回，其中List中只有一个Map，其Key含义描述如表5.18所示：

表5.18 人脸N:N接口返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| PERSON\_IDS | 人员记录\_id列表 | List<List<Long>> | 内层每个List代表一个人对应的所有记录\_id，满足查询条件的所有记录\_id都会出现在此列表中，内层List之间根据元素个数倒序排序 |
| QUERY\_IDS | 查询记录\_id列表 | List<List<Long>> | 内层每个List代表一组查询对应的所有记录\_id，满足查询条件的所有记录\_id都会出现在此列表中，内层List之间的顺序与输入参数querys中每个查询的顺序一一对应，当querys中的查询条件只有一组时，不返回此列表 |

### 限制

限制输入参数querys中所有条件对应的全部记录总数不超过10000条，若超过，返回超过限制条数提示。

### 返回示例

1. {"code":0,"message":"分析完成","list":[{"QUERY\_IDS":[[131900000000000000,131900000000000001,131900000000000002],[131900000000000003,131900000000000004,131900000000000005]],"PERSON\_IDS":[[131900000000000000,131900000000000003,131900000000000004],[131900000000000001,131900000000000002,131900000000000005]]}]}
2. {"code":10013,"message":"检索记录条数超过设定的阈值（10000条），请缩小查询条件范围","list":[]}

# 静态小库

## 相关说明

小库id（libraryId）是小库的唯一标识；

记录id（ID）是保存小库记录的索引的\_id值。

## 创建静态小库

### 创建空的静态小库

#### 描述

接口名：FACEDB\_CREATE

创建一个空的静态小库，引擎中小库以libraryId命名。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.1所示：

表6.1创建静态小库API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 需要创建的小库id |

#### 返回

以2.1节的结构返回，以code值判断是否创建小库成功，message记录成功信息或失败原因，list为空列表。

#### 返回示例

{"code":0,"message":"静态小库创建成功","list":[]}

### 根据条件创建小库

#### 描述

接口名：FACEDB\_CREATE

根据查询条件从一个索引中查询数据，并以此数据构建一个静态小库，引擎中小库以libraryId命名。

目前创建小库最大支持数据条数为10000条。

#### 参数

以Map<String, Object>对象作为参数传入API，，每个Key含义描述如表6.2所示：

表6.2创建静态小库API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库名 | String | √ | 需要创建的小库id |
| indexName | 索引名 | String | √ | 人脸特征所在的索引名，多个索引名用英文逗号分割，如：face\_1701,face\_1702 |
| typeName | type名 | String | √ | 人脸特征所在的索引type名 |
| idFieldName | 主键字段名 | String | √ | 常见有INFO\_ID，XXBH，PERSON\_ID等 |
| rltzFieldName | 人脸特征字段名 | String | √ | 一般为RLTZ |
| query | 布尔查询对象 | BoolQueryBuilder | √ | 只获取符合该查询条件的记录 |

query参数使用对象org.elasticsearch.index.query.BoolQueryBuilder

**注意：**

目前本接口query参数只支持如下查询条件：

① termsQuery

多值匹配查询。

② rangeQuery

范围查询，目前只支持大于等于（gte）和小于等于（lte）查询，并且gte和lte必须同时出现，如对于integer类型的字段，未知gte值时，可设置为Integer.MAX\_VALUE，如是long类型，则可设置为Long.MAX\_VALUE，lte同理。

**示例：**

BoolQueryBuilder query = QueryBuilders.boolQuery();

QueryBuilder q1 = QueryBuilders.termsQuery("FILE\_ID",

new long[]{121632433832460288L, 121632433832460288L});

QueryBuilder q0 = QueryBuilders.rangeQuery("JGSK")

.gte(160101000000L).lte(160102000000L);

query.must(q0);

query.must(q1);

#### 返回

以2.1节的结构返回，以code值判断是否创建小库成功，message记录成功信息或失败原因，list为空列表。

#### 返回示例

① {"code":0,"message":"静态小库创建成功","list":[]}

② {"code":12000,"message":"用于创建小库的记录条数超过设定的阈值（10000条），请缩小查询条件范围","list":[]}

## 判断静态小库是否存在

### 描述

接口名：FACEDB\_ISEXIST

根据小库的唯一标识，判断小库是否已在引擎端创建。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.3所示：

表6.3判断静态小库是否存在API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 小库id |

### 返回

以2.1节的结构返回，以code值判断是否创建小库成功，message记录成功信息或失败原因，list保存一个boolean类型值，为true时表示小库存在，false时表示不存在。

### 返回示例

{"code":0,"message":"成功判断静态小库是否存在","list":[true]}

## 删除静态小库

### 描述

接口名：FACEDB\_DELETE

根据单个小库唯一标识，删除在引擎端对应的小库。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.4所示：

表6.4删除静态小库API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 要删除的小库唯一标识 |

### 返回

以2.1节的结构返回，以code值判断是否删除小库成功，message记录成功信息或失败原因，list为空列表。

### 返回示例

{"code":0,"message":"删除静态小库成功","list":[]}

## 注册人脸到静态小库

### 传参方式注册人脸

#### 描述

接口名：FACEDB\_REGISTER

把多个人脸特征注册到引擎端的对应小库中，允许重复注册。

目前通过参数方式注册人脸到小库单次最大支持数据条数为1000条。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.5所示：

表6.5注册人脸到静态小库API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 需要注册人脸的小库id |
| features | 多组人脸特征 | Map<Long, String> | √ | 由Map保存的多个记录id和人脸特征的组合，其中key值为记录id，value值为使用BASE64编码的人脸特征字符串 |

#### 返回

以2.1节的结构返回，以code值判断是否注册人脸到小库成功，message记录成功信息或失败原因，list值为List<Long>，若存在部分注册失败的人脸记录，则List中每个元素为注册失败的人脸特征记录id列表。

#### 返回示例

① {"code":0,"message":"注册全部人脸到小库成功","list":[]}

② {"code":0,"message":"注册多个人脸到小库成功，部分由于人脸特征不合法导致注册失败","list":[121632433874403321]}

③ {"code":12000,"message":"全部人脸特征不合法","list":[121632433874403322,121632433874403321]}

### 传条件方式注册人脸

#### 描述

接口名：FACEDB\_REGISTER

通过传入条件从某个索引查询到多个人脸特征，并注册到引擎端的对应小库中，允许重复注册。

目前通过条件方式注册人脸到小库单次最大支持数据条数为10000条。

#### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.5所示：

表6.5注册人脸到静态小库API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 需要注册人脸的小库id |
| indexName | 索引名 | String | √ | 人脸特征所在的索引名，多个索引名用英文逗号分割，如：face\_1701,face\_1702 |
| typeName | type名 | String | √ | 人脸特征所在的索引type名 |
| idFieldName | 主键字段名 | String | √ | 常见有INFO\_ID，XXBH，PERSON\_ID等 |
| rltzFieldName | 人脸特征字段名 | String | √ | 一般为RLTZ |
| query | 布尔查询对象 | BoolQueryBuilder | √ | 只获取符合该查询条件的记录 |

query参数使用对象org.elasticsearch.index.query.BoolQueryBuilder

**注意：**

目前本接口query参数只支持如下查询条件：

① termsQuery

多值匹配查询。

② rangeQuery

范围查询，目前只支持大于等于（gte）和小于等于（lte）查询，并且gte和lte必须同时出现，如对于integer类型的字段，未知gte值时，可设置为Integer.MAX\_VALUE，如是long类型，则可设置为Long.MAX\_VALUE，lte同理。

**示例：**

BoolQueryBuilder query = QueryBuilders.boolQuery();

QueryBuilder q1 = QueryBuilders.termsQuery("FILE\_ID",

new long[]{121632433832460288L, 121632433832460288L});

QueryBuilder q0 = QueryBuilders.rangeQuery("JGSK")

.gte(160101000000L).lte(160102000000L);

query.must(q0);

query.must(q1);

#### 返回

以2.1节的结构返回，以code值判断是否注册成功，message记录成功信息或失败原因，list为空列表。

#### 返回示例

① {"code":0,"message":"注册人脸到小库成功","list":[]}

② {"code":12000,"message":"单次注册的人脸记录条数超过设定的阈值（10000条），请缩小查询条件范围","list":[]}

## 从静态小库注销人脸

### 描述

接口名：FACEDB\_LOGOUT

从引擎端的小库注销对应id的人脸。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.6所示：

表6.6从静态小库注销人脸API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryId | 小库id | String | √ | 需要注销人脸的小库id |
| ids | 多个记录id | String | √ | 需要注销的人脸记录id，多个用英文逗号分割，如："110559896386142336,110560454694142028" |

### 返回

以2.1节的结构返回，以code值判断是否注销人脸到小库成功，message记录成功信息或失败原因，list为空列表。当注销失败时，接口调用方应认为全部传入的id都注销失败。

### 返回示例

{"code":0,"message":"注销人脸成功","list":[]}

## 静态小库人脸1:N检索

### 描述

接口名：FACEDB\_ONE2N\_SEARCH

在一个小库中检索与传入图片相似度最高的一批人脸。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.7所示：

表6.7静态小库人脸1:N检索API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| indexName | 索引名 | String | √ | 小库所在的索引名，多个索引名用英文逗号分割，如：face\_1701,face\_1702 |
| typeName | type名 | String | √ | 人脸特征所在的索引type名 |
| libraryId | 小库名 | String | √ | 想要查询的静态小库 |
| idFieldName | 主键字段名 | String | √ | 静态小库的主键字段名称，常见有INFO\_ID，XXBH，PERSON\_ID等 |
| similarity | 相似度阈值 | int | √ | 取值范围：[0, 100]，只返回相似度大于等于该值的结果 |
| url | 人脸图片URL | String | √ | 目标人脸图片URL |
| topN | topN | int | √ | 最多返回结果条数 |
| query | 布尔查询对象 | BoolQueryBuilder | × | 只返回符合该查询条件的记录\_id，若不传该参数，则默认检索当前静态小库的全部人脸 |

query参数使用对象org.elasticsearch.index.query.BoolQueryBuilder

**注意：**

目前本接口query参数只支持如下查询条件：

① termsQuery

多值匹配查询。

② rangeQuery

范围查询，目前只支持大于等于（gte）和小于等于（lte）查询，并且gte和lte必须同时出现，如对于integer类型的字段，未知gte值时，可设置为Integer.MAX\_VALUE，如是long类型，则可设置为Long.MAX\_VALUE，lte同理。

**示例：**

BoolQueryBuilder query = QueryBuilders.boolQuery();

QueryBuilder q1 = QueryBuilders.termsQuery("FILE\_ID",

new long[]{121632433832460288L, 121632433832460288L});

QueryBuilder q0 = QueryBuilders.rangeQuery("JGSK")

.gte(160101000000L).lte(160102000000L);

query.must(q0);

query.must(q1);

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表6.8所示：

表6.8静态小库人脸1:N检索API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| ID | 静态小库ID字段值 | String | 静态小库ID字段值 |
| SIMILARITY | 相似度 | int | 取值范围：[0, 100]，表示ID对应的人脸记录与提交的人脸之间的相似度 |

以SIMILARITY降序排序。

### 返回示例

{"code":0,"message":"分析完成","list":[{"ID":128230829339148418,"SIMILARITY":69},{"ID":128230944942555270,"SIMILARITY":69},{"ID":128230916056383616,"SIMILARITY":67}]}

## 静态小库人脸M:N比对

### 描述

接口名：FACEDB\_M2N\_SEARCH

对多个小库进行M:N比对，找出在这些小库两两之间相似度最高的一批人。

### 参数

以Map<String, Object>对象作为参数传入API，每个Key含义描述如表6.9所示：

表6.9静态小库人脸M:N比对API输入参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **是否必要** | **描述** |
| libraryIds | 多个小库id | String | √ | 想要查询的多个小库，使用英文逗号分割，如："library1,library2"，至少传入两个小库id，第一个小库id为基准库，其他小库id都为检索库。  注意：目前只支持两个小库进行M:N比对，请不要传入两个以上的小库 |
| similarity | 相似度阈值 | int | √ | 取值范围：[0, 100]，只返回相似度大于等于该值的结果 |
| topN | topN | int | √ | 最多返回结果条数 |
| eachTopN | eachTopN | int | √ | 对每个人脸返回topN个得分最高的记录 |

### 返回

以2.1节的结构返回，其中List中每个Map的Key含义描述如表6.10所示：

表6.10 静态小库人脸M:N比对API返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| ID | 基准静态小库的记录id | String | 基准静态小库的记录id |
| LIST | 相似度列表 | List | ID对应人脸与检索库的人脸之间的相似度列表，每个元素为一个Map，参照表6.11 |

以LIST中第一个相似度降序排序。

表6.11 相似度Map

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **名称** | **数据类型** | **描述** |
| ID | 检索库的记录id | String | 检索库的记录id |
| SIMILARITY | 相似度 | int | 两个人脸之间的相似度 |

以SIMILARITY降序排序。

### 返回示例

{"code":0,"message":"分析完成","list":[{"ID":121632433589190657,"LIST":[{"ID":121633523588136963,"SIMILARITY":89},{"ID":121636481260322818,"SIMILARITY":89}]},{"ID":121632433589190658,"LIST":[{"ID":121632433589190658,"SIMILARITY":89},{"ID":121633523592331264,"SIMILARITY":89}]},{"ID":121632433593384962,"LIST":[{"ID":121636481889468422,"SIMILARITY":89},{"ID":121636482178875398,"SIMILARITY":89}]},{"ID":121632433597579264,"LIST":[{"ID":121632433597579264,"SIMILARITY":89},{"ID":121633523596525570,"SIMILARITY":89}]},{"ID":121632433610162176,"LIST":[{"ID":121636481889468422,"SIMILARITY":89},{"ID":121636482178875398,"SIMILARITY":89}]}]}