**工业识别V1.0.0-算法接口**

**请求URL**

demo:

POST /devicesrecognize

V1.0.0:

POST /iot/device/recognize/image

**参数列表**

|  |  |  |  |
| --- | --- | --- | --- |
| **返回参数名称** | **类型** | **是否必须** | **说明** |
| tags | string | 否 | 标签信息 |
| imgName | string | 若传入url参数，此参数可不传 | 图片名，以后端传入的为主 |
| image | string | 和url二选一 | 图像数据，base64编码，要求base64编码后大小不超过4M，最短边至少50px，最长边最大4096px，支持jpg/png/bmp格式 。**注意：图片需要base64编码、去掉编码头后再进行urlencode。** |
| url | string | 和image二选一 | 图片完整URL，URL长度不超过1024字节，URL对应的图片base64编码后大小不超过4M，最短边至少50px，最长边最大4096px，支持jpg/png/bmp格式，当image字段存在时url字段失效。 |

**请求响应**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **返回参数名称** |  |  |  | **类型** | **是否必须** | **说明** |
| serviceVersion |  |  |  | string | 是 | 识别服务版本号 |
| timestamp |  |  |  | string | 是 | 调用服务时的时间戳，使用python的int(time.time()产生 |
| imgName |  |  |  | string | 是 | 图片名，以后端传入的为主 |
| status |  |  |  | string | 是 | 状态码 |
| message |  |  |  | string | 是 | 状态码信息说明 |
| result |  |  |  | dict | 是 | 返回结果相关 |
|  | imgScore |  |  | float32 | 是 | 图像质量评估得分 |
| recognizeInfo |  |  | list | 是 | 识别结果详细内容 |
|  | deviceCrop |  | base64 | 是 | 检测到的设备/零件目标截取图 |
| box |  | list | 是 | 检测到的设备/零件目标相对于原图的坐标 |
|  | left | int | 是 | 目标框左上角的 x 坐标 |
| top | int | 是 | 目标框左上角的 y 坐标 |
| right | int | 是 | 目标框右下角的 x 坐标 |
| bottom | int | 是 | 目标框右下角的 y 坐标 |
| deviceList |  | list | 是 | 识别设备/零件目标ID的top2结果或者阈值大于 |
|  | deviceType | int | 是 | 设备目标类型，设备整体deviceType=0(如deviceId=0000A)；设备零件deviceType=1(如deviceId=0000A\_0、0000A\_1，0000A\_3.......等等) |
| deviceId | string | 是 | 识别到的设备/零件目标的所属ID |
| score | float32 | 是 | 识别到的设备/零件目标的置信度得分 |

**请求示例**

|  |
| --- |
| SQL # -\*- coding:utf-8 -\*-  import requests import json import argparse  detail\_template = {  "tags": "1.0.0",  "imgName": "2024-05-24\_14:38:50.jpg",  "image": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQE......",  "url": "https://iot-bucket-2024.obs.cn-south-1.myhuaweicloud.com:443/iot/436f1fd9-ea3d-43f8-be8f-af08961a3021.jpeg", }   class RecognizeTest(object):  def \_\_init\_\_(self, base\_url):  self.base\_url = base\_url   def device\_recognize(self, data):  url = self.base\_url + '/devicesrecognize'  header = {"Content-Type": "application/json"}  params = data  response = requests.post(data=json.dumps(params), url=url, headers=header)  try:  print("final result: ", response.json())  except Exception as e:  print("Error", e)  print(response.json())  return response.json()   if \_\_name\_\_ == '\_\_main\_\_':  parser = argparse.ArgumentParser()  url = "http://192.168.101.120:8080"  parser.add\_argument('--url', default=url, required=False, help='')  parser.add\_argument('--detail\_template', default=detail\_template, required=False, help='')  opt = parser.parse\_args()  recognize\_test = RecognizeTest(opt.url)  recognize\_test.device\_recognize(detail\_template) |

**[cURL.txt]**

|  |
| --- |
| SQL # -\*- coding:utf-8 -\*- from flask import Flask, request, Response import json  app = Flask(\_\_name\_\_)      def predict\_image(img\_content):  *"""*  *识别主函数* ***:param*** *img\_content:* ***:return****:*  *"""* data = {  "serviceVersion": "v1.0.0",  "timestamp": "1242536363",  "imgName": "2024-05-24\_14:38:50.jpg",  "status": "A0200",  "message": "runSuccess",  "result": {  "imgScore": 0.6,  "recognizeInfo": [{'deviceCrop': "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQE......",  'box': [{"left": 172,  "top": 402,  "right": 588,  "bottom": 928}],  "deviceList": [{'deviceId': '10001', 'score': 0.87, "deviceType": 0},  {'deviceId': '10002', 'score': 0.76, "deviceType": 0}]},  {'deviceCrop': "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQE......",  'box': {"left": 523,  "top": 161,  "right": 994,  "bottom": 919},  "deviceList": [{'deviceId': '10003\_0', 'score': 0.91, "deviceType": 1},  {'deviceId': '10004\_0', 'score': 0.88, "deviceType": 1}]}  ]  }  }  return data   @app.route('/devicesrecognize', methods=['POST']) def devices\_recognize():  *"""*  *识别入口* ***:return****:*  *"""* request\_data = request.get\_json()  image\_input = request\_data["image"]  predict\_info = predict\_image(image\_input)  print(predict\_info)  return Response(json.dumps(predict\_info))   if \_\_name\_\_ == '\_\_main\_\_':  app.run(host='0.0.0.0', port=8080, processes=True, debug=False) |

**[return.txt]**