算法资料

选手端 • 算法 Agent 网关与返回数据结构说明

一、Agent 网关部署位置（固定）

项	约定
部署位置	**选手本机或可被本机浏览器访问的同一内网地址**（与参赛端 `fetch` 同源策略、CORS 策略一致）
登记地址	队长在平台填写的 **Agent Base URL** 必须为**根地址**，例如 `http://127.0.0.1:8080`（**不要**省略协议；**不要**在末尾多写 `/infer`，由平台拼接）
必须实现的 HTTP 路径	`**GET /health`、`**POST /infer`（路径名固定，大小写敏感）
实现形态	**二选一**：① 网关与算法同一进程实现两接口；② 网关为反向代理，将上述两路径**原样转发**至本机算法服务且**响应体不改写**

二、HTTP 与传输（固定）

项	约定
字符编码	**UTF-8**
`POST /infer` 请求头	`**Content-Type: application/json`**（必须）
`POST /infer` 成功响应	**HTTP 状态码必须为 2xx**；响应体为 **合法 JSON**
`POST /infer` 协议失败	HTTP **≥400** 或 非 JSON 正文 → **当次抽样失败**，整题按平台规则失败
超时	单次推理须在请求 `**meta.infer_T_max_ms`**（毫秒）内完成；浏览器会在此基础上增加少量余量后中断
跨域	竞赛页域名访问本机网关时，**必须**对 `/health`、`/infer`（及浏览器可能发起的 `**OPTIONS` 预检）**返回合规** CORS** 头，否则浏览器无法发请求

三、`GET /health`（固定）

3.1 请求

```
GET {agent_base_url}/health
```

`{agent_base_url}` 为队长登记的 Base URL（去掉末尾 `/` 后由平台拼接 `/health`，若登记项已以 `/health` 结尾则按参赛端逻辑不再重复追加）。

3.2 响应（固定要求）

- **HTTP 状态码**：**200**
- **响应体**：**JSON 对象**
- **字段**：

字段	类型	必须	说明
`status`	string	**是**	平台用于判断在线；**必须为** `"ok"`（小写）表示可接受评测
`supported_tasks`	string[]	**是**	**必须**列出本机已实现的 `task_type`；须为 `classify`、`ocr`、`detect` 的子集，且**须包含本队实际要打的题型**（**顺序与** `app.py` **一致**：`classify` → `ocr` → `detect`）
`service`	string	否	`**app.py` 固定返回**：服务名 `contestant-algo-test-service`
`version`	string	否	`**app.py` 固定返回**：与 `APP_VERSION` 一致
`bridge_mode`	string	否	`**app.py` 固定返回**：`mock-local`

3.3 响应示例（与 `app.py` 中 `health()` 返回一致）

```
{
  "status": "ok",
  "supported_tasks": ["classify", "ocr", "detect"],
  "service": "contestant-algo-test-service",
  "version": "0.2.0",
  "bridge_mode": "mock-local"
}
```

自研网关可省略 `service` / `version` / `bridge_mode`；**与** `app.py` **联调对齐时以上字段以示例为准。**

四、`POST /infer` — 通用请求 / 响应信封（固定）

4.1 请求体 — 所有题型共用的顶层字段（必须出现）

字段	类型	必须	说明
`request_id`	string	**是**	平台生成，**全局唯一**；响应中**必须原样回显**
`session_id`	string	**是**	平台生成；**请求必填。**响应是否回传见 §4.2（`**app.py` 不回传**）
`task_type`	string	**是**	**语义上**仅支持 `ocr`、`classify`、`detect`；`**app.py` 会对请求值 `strip().lower()`**，响应中的 `task_type` 一律为小写

字段	类型	必须	说明
`image`	object	**是**	**必须**同时包含 `format`（string）与 `data`（string）；常见为 `format: "url"` 且 `data` 为题图 URL（`app.py` 中 `format` 缺省为 `"url"`）
`meta`	object	否（`**app.py` 默认 `{}**`）	判分上下文；平台通常会带 `difficulty`、`sample_index`、`sample_k`、`infer_T_max_ms` 等；**请求体省略 `meta` 时 `app.py` 视为空对象**；题型相关字段见下各节

4.2 响应体 — 所有题型共用的顶层字段（必须出现）

字段	类型	必须	说明
`request_id`	string	**是**	**必须与请求完全一致**
`session_id`	string	否	`**app.py` 成功/失败响应均不返回该字段**；平台判分未校验响应内 `session_id`；自研网关可回显也可省略
`task_type`	string	**是**	**必须与请求 `task_type` 一致**（平台大小写不敏感比对）；不一致 → **整题失败**
`ok`	boolean	**是**	`true` 表示进入 `result` 结构化校验与判分；`false` 表示本抽样失败（**仍须返回合法 JSON**）
`result`	object	null	**是**
`message`	string	**是**	失败原因；成功时 **必须为** `""`（空字符串）
`elapsed_ms`	number	**是**（与 `app.py` 一致）	整数毫秒；`app.py` 在成功与失败分支均返回

说明：平台从响应中取 `**result` 对象**送判分器；若顶层缺少 `result` 键，会把**根对象**当作 `result`（兼容旧实现）。`**app.py` 始终包含 `result` 键。**

4.3 与 `app.py` 一致的 `meta.extra`（联调可选）

`app.py` 会读取 `**meta.extra`**（若存在且为对象）：

字段	类型	说明
`delay_ms`	number	本次推理前睡眠毫秒数；缺省为环境变量 `MOCK_DELAY_MS` 或 **120**
`force_error`	boolean	为 `true` 时强制返回 `ok: false`（见下失败响应示例文案）

说明：若 `meta.extra` 存在但**不是**对象，`app.py` 视为未配置：延迟用 `MOCK_DELAY_MS` 或 **120**，`force_error` 为 `false`。

五、`task_type: "ocr"` — 请求与返回（固定示例）

5.1 请求体示例（完整 JSON）

与 `app.py` 中 `mock_ocr_result` 一致的行为（占位文本来源）：

- 先取 `meta.expected.text`（当 `meta.expected` 为对象且 `text` 为字符串时写入中间变量）；若存在 `meta.samples[0].expected.text`（`samples` 为非空列表且首元素为含 `expected` 的对象），则用其**覆盖**该中间变量。
- 上述合并结果 `strip()` 后非空，则返回 `{"text": "<该字符串>"}`。
- 否则对 `image.data` 做 SHA-256，取摘要前 **8** 个十六进制字符大写为 token，返回 `{"text": "DEMO-<TOKEN>"}`。

```
{
  "request_id": "eval-1205-1-1",
  "session_id": "team-7",
  "task_type": "ocr",
  "image": {
    "format": "url",
    "data": "https://platform.example.com/files/signed/abc123.jpg"
  },
  "meta": {
    "difficulty": "L2",
    "sample_index": 1,
    "sample_k": 1,
    "infer_T_max_ms": 15000,
    "language_hint": "zh",
    "normalize_rules": {
      "trim_space": true,
      "case_insensitive": false
    },
    "draw_source": "admin-bank"
  }
}
```

`result` 内字段（固定）

字段	必须	说明
`text`	**约定主字段（必须实现）**	识别结果字符串；平台与标准答案做同一套 `normalize_rules` 后 **全等**比对
`content`	否	仅当未提供 `text` 时平台会读取 `content`；**交付时必须有 `text`，不得依赖 `content`**

5.2 响应体示例（与 `app.py` 成功分支字段顺序一致：`request_id` → `task_type` → `ok` → `result` → `elapsed_ms` → `message`）

```
{
  "request_id": "eval-1205-1-1",
  "task_type": "ocr",
  "ok": true,
```

```
  "result": {
    "text": "阀位12.8%"
  },
  "elapsed_ms": 85,
  "message": ""
}
```

5.3 `ok: false` 时响应示例（与 `app.py` 中 `meta.extra.force_error` 分支一致）

```
{
  "request_id": "eval-1205-1-1",
  "task_type": "ocr",
  "ok": false,
  "result": null,
  "elapsed_ms": 128,
  "message": "按 meta.extra.force_error 指令返回模拟失败"
}
```

（`elapsed_ms` 为 `time.perf_counter()` 换算的整数毫秒，**含** `meta.extra.delay_ms`（或默认 `MOCK_DELAY_MS` / 120）睡眠耗时，故通常 ≥ 配置的延迟；上式中 `128` 仅为示意。）

六、`task_type: "classify"` — 请求与返回（固定示例）

6.1 请求体示例（完整 JSON）

```
{
  "request_id": "eval-1205-1-1",
  "session_id": "team-7",
  "task_type": "classify",
  "image": {
    "format": "url",
    "data": "https://platform.example.com/files/signed/def456.jpg"
  },
  "meta": {
    "difficulty": "L1",
    "sample_index": 1,
    "sample_k": 1,
    "infer_T_max_ms": 15000,
    "class_names": ["正常", "告警", "离线"],
    "draw_source": "admin-bank"
  }
}
```

`result` 内字段（固定主写法）

字段	必须	说明
label	**是（固定使用本字段）**	单标签字符串；**必须**落在当次 meta.class_names 内（大小写不敏感），否则该抽样 **0 分**（invalid_label_name）

与 app.py 中 mock_classify_result 一致：若 meta.class_names 为列表，先将其转为字符串列表作为候选；若存在 meta.samples[0].meta.class_names 且为列表，则**用其替换**整个候选列表（与代码顺序一致）。候选非空时，用 stable_random(image.data)（SHA-256 摘要前 16 个十六进制字符转整数为 random.Random 种子）从中选一项返回。候选仍为空时，从固定池 **["normal", "alarm", "offline"]** 中同样方式选一项。§6.1 请求示例中的中文类名仅演示平台下发形状，与无类名时的英文池无关。

平台仍兼容读取的别名字段（非本约定主写法）：label_id、prediction、class；**新开发只输出 label 即可。**

6.2 响应体示例（与 app.py 成功分支一致）

```
{
  "request_id": "eval-1205-1-1",
  "task_type": "classify",
  "ok": true,
  "result": {
    "label": "正常"
  },
  "elapsed_ms": 42,
  "message": ""
}
```

七、task_type: "detect" — 请求与返回（固定示例）

7.1 坐标系（固定）

- **meta.coord_mode 必须为字符串 "pixel"**（在平台下发的 meta 中；根或样本合并后一致）。
- **评委框 expected.boxes[].xyxy 与选手返回的坐标全部为像素**，原点为**题图左上角**，x 向右、y 向下，与**该题图自然宽高**一致。
- **禁止使用 0～1 归一化坐标**；coord_mode 非 pixel 时平台判分 **0 分**或入库校验失败。

7.2 请求体示例（完整 JSON）

```
{
  "request_id": "eval-1205-1-1",
  "session_id": "team-7",
  "task_type": "detect",
  "image": {
    "format": "url",
    "data": "https://platform.example.com/files/signed/ghi789.jpg"
  },
```

```
  "meta": {
    "difficulty": "L2",
    "sample_index": 1,
    "sample_k": 1,
    "infer_T_max_ms": 15000,
    "coord_mode": "pixel",
    "scoring": "center_in_box",
    "class_names": ["defect", "valve"],
    "image_width": 640,
    "image_height": 480,
    "draw_source": "admin-bank"
  }
}
```

说明：`image_width` / `image_height` 为题面参考尺寸（若平台下发）；**判分几何以像素 xyxy / cx,cy 与题库 `expected` 一致为准。**

7.3 `result` 内结构（与 `app.py` 一致：`mock_detect_targets_result`）

固定：`result` 为对象，且含 `**targets` **非空数组。每项**必须**含：

字段	必须	说明
`label`	**是**	字符串；`app.py` 在存在 `meta.class_names` 时从中随机选一，否则为 `"defect"`
`cx`	**是**	浮点数；`app.py` 为 `80～560` 内随机整数转 float（像素 x）
`cy`	**是**	浮点数；`app.py` 为 `60～420` 内随机整数转 float（像素 y）
`score`	否	`app.py` 会附带 `0.9～0.99` 之间三位小数；平台判分不强制校验

说明：竞赛平台后端另接受 `**boxes` / `detections` **等结构（见 `VisualScoringService`）；**本仓库参考实现 `app.py` 仅实现上述 `targets` 形状**，文档以此为准。

7.4 响应体示例（与 `app.py` 成功分支、`mock_detect_targets_result` 一致）

```
{
  "request_id": "eval-1205-1-1",
  "task_type": "detect",
  "ok": true,
  "result": {
    "targets": [
      {
        "label": "defect",
        "cx": 320.0,
        "cy": 240.0,
        "score": 0.942
      }
    ]
```

```
  },
  "elapsed_ms": 118,
  "message": ""
}
```

（cx/cy/label/score 的具体数值由 app.py 依 image.data 与 meta 随机生成，上表仅为形状示例。）

7.5 不支持 task_type 时（与 app.py 分支一致）

```
{
  "request_id": "eval-1205-1-1",
  "task_type": "keypoint",
  "ok": false,
  "result": null,
  "elapsed_ms": 125,
  "message": "不支持的 task_type: keypoint，当前示例仅支持 ['classify', 'ocr', 'detect']"
}
```

（message 与 app.py 在未知 task_type 分支（约第 87–95 行）生成的字符串格式一致；elapsed_ms 同样**含**前置 delay_ms 睡眠，非固定值，上式为示意。）

九、交付自检清单（必须满足）

- GET /health：**200** + JSON，status === "ok"，supported_tasks **包含**本队参赛所需的全部 task_type
- POST /infer：三种题型均能按上文章节与 **app.py **返回**完整信封**且 result 符合表结构
- request_id / task_type：**响应与请求一致**（session_id 响应可省略，与 app.py 一致）
- **classify：result.label 属于**当次 meta.class_names
- **detect：**与 app.py **一致时仅返回 **result.targets（像素 cx/cy）；平台另支持 boxes 等见后端实现
- **ocr**：result.text **存在**且为字符串
- 跨域、超时已在真实浏览器环境下验证通过

文档版本：**JSON 形状与字段以** 初赛竞赛系统/选手算法服务-test/app.py **为准**；平台判分扩展行为见 VisualScoringService、ClientEvaluationResultApplier。

选手端 · 在线答题内容范围和难度说明

选手端 · 在线答题：考查重点与学习建议

在线答题围绕**工业 AI 视觉**相关的基础知识、常用算法与工程落地展开。

当前系统题量与计分：

题型	数量	单题分值	判分方式
单选 L1	24 道	0.8 分	系统自动判分，选项与标准答案一致得满分，否则 0 分
单选 L2	24 道	1.2 分	系统自动判分，选项与标准答案一致得满分，否则 0 分
单选 L3	10 道	2 分	系统自动判分，选项与标准答案一致得满分，否则 0 分
简答题	2 道	16 分	提交后由评委人工评分，可给 0～16 分

在线答题卷面合计 **100 分**，其中单选 **68 分**、简答 **32 分。**系统计入初赛总分时会折算为 **40 分**：

```
在线答题计入分 = 在线答题卷面得分 / 100 × 40
```

简答题未完成评审前，平台可能先隐藏或暂不展示最终在线答题总分；评审完成后再形成完整在线答题得分。

主要考什么

整体可分成几块，复习时按块建立知识树即可。

机器视觉与数字图像基础
视觉系统各环节在做什么；像素、分辨率、灰度与常见颜色模型；图像增强、滤波、几何校正；相机标定、畸变、手眼关系等**概念级**理解。

传统图像处理与特征
边缘/角点、霍夫变换、轮廓与模板匹配；形态学基本操作；阈值与分割、纹理与频域的**基本思想**；视频与运动目标相关入门概念。

深度学习与典型视觉任务
卷积网络里常见模块各自解决什么问题；经典检测/分割网络**各自适合什么场景、结构上有何特点**（不必背代码，但要分得清「两阶段 / 单阶段」「语义 / 实例 / 全景」等）；与双目、结构光、线激光、点云、多视图相关的**术语与原理轮廓。**

三维视觉、SLAM 与传感器
视觉里程计、SLAM 常见路线里**传感器与侧重点**的差异；新视角合成、神经辐射场等**了解在解决什么问题即可。**

工业场景与工程落地
缺陷检测、测量、定位引导、读码等**典型应用**；光源与镜头选型思路；嵌入式上的**量化、剪枝、蒸馏、推理加速**等「为什么做、大致怎么做」。

拓展与综合
视觉 Transformer、自监督与小样本、域适应、多模态、开放集检测等**名词与动机**；部分题目会考「下列哪项**不属于**某类方法」——需要对易混概念**正反都能说清。**

简答题倾向
综合论述或方案设计，例如：传统视觉与深度学习的**对比与选型**、缺陷检测**系统构成与流程**、嵌入式部署**优化思路与现场注意点**、工业制造中 AI 视觉的**趋势与挑战**等。要求**条理清楚、能结合场景**，而不是堆砌术语。

中控杯初赛・在线评测算法资料

一、视觉侧总体结构（请按此规划训练）

维度	固定约定
任务类型 `task_type`（本平台）	`classify`（图像分类）·`detect`（目标检测）·`ocr`（文字识别）
难度档位 `difficulty`	`L1`（易）·`L2`（中）·`L3`（难），与题库一一对应
作答单元	每一组 `task_type` + `difficulty` 称为一个**槽位**；当前系统共 **3 类 × 3 档 = 9 个槽位 / 9 道在线评测题**，合计 **60 分**（每槽在规则内可多次抽图重试，取有效成绩）
接口形态	平台经本队 **Agent 网关** 调用你方服务的 `GET /health`、`POST /infer`（JSON）；具体字段以平台《API 与评测说明》为准

健康检查：`GET /health` 的 JSON 中建议包含 `supported_tasks`，须覆盖你队承诺支持的类型，例如：`["ocr","detect","classify"]`。

二、三档难度 `L1` / `L2` / `L3`（所有题型通用）

档位	名称	能力侧重（训练方向）
L1	难度一	题面相对规整：对比度高、透视/遮挡较少；可用**传统图像处理 + 轻量模型**或小型网络达到可用精度。
L2	难度二	**光照、视角、尺度**变化与一定干扰；建议**数据增强 + 经典小模型**（如轻量分类网、小目标检测头）。
L3	难度三	**细粒度、密集、弱对比、类间易混**等；以**深度学习**为主，强调鲁棒性与召回，注意过拟合与误检。

抽题方式：当前系统按固定槽位作答，即 `classify`、`detect`、`ocr` 三类各完成 `L1`、`L2`、`L3` 一题，共 9 个槽位。每个槽位从对应题库随机抽图；同槽位最多可新增随机抽图 2 次，同一张图可重试；最终计分时取该槽位已成功任务中的最高分。

三、任务类型与「题型标签」固定表

下表为选手在 `/infer` **请求体**中会看到的 `task_type` 与中文含义；**请按行实现分支逻辑**，并与 `supported_tasks` 一致。

`task_type`（固定英文，小写）	中文题型	题意摘要	与判分强相关的输出要点
`classify`	工况 / 场景分类	判断整图属于哪一类工况或场景	输出与 `meta.class_names` 一致的**类别标签**
`detect`	工业目标检测	在图中定位指定类别目标	`meta.coord_mode` **固定为** `pixel`：`targets`/`boxes` 的坐标均为**相对题图左上角的像素**（与题图宽高一致）；类别须在 `class_names` 内
`ocr`	表计 / 铭牌文字识别	读出指定区域或整图的规范文本	输出**字符串**；注意与 `meta.normalize_rules` 等一致

说明：同一 `task_type` 下，**L1 / L2 / L3** 通过**不同子题库**体现难度差异（成像复杂度、类别数、干扰等），**不改变**你在 `infer` 里应返回的**字段结构大类**（分类仍是类别、检测仍是框/点、OCR 仍是文本）。

四、各题型按难度的命题主轴（训练目标）

4.1 文字识别 `ocr`

档位	命题主轴（建议训练数据形态）
L1	单行、规整印刷体或数码管类读数；高对比、少粘连
L2	多行、轻透视或阴影；需分行或整体拼接策略
L3	强反光、污渍、小字密集；宜「检测文本行 + 识别」或端到端鲁棒模型

4.2 目标检测 `detect`

本届在线评测平台的**工业目标检测**固定为下列 **9 类目标。**参赛队训练检测模型时，至少需要覆盖这 9 个目标类别；`/infer` 返回的 `targets[].label` 或 `boxes[].label` 必须与下方中文标签一致。

序号	检测类别
1	人
2	汽车
3	自行车
4	手机
5	水杯
6	笔记本电脑
7	台灯
8	沙发

序号	检测类别
9	狗

档位	命题主轴（建议训练数据形态）
L1	单类或少实例、朝向较正；框尺度适中
L2	多类多实例、尺度变化与部分遮挡；含一定比例小目标
L3	密集、弱对比、类间易混；需控制误检与漏检

坐标系（固定）：**仅像素。**`meta.coord_mode` 恒为 `"pixel"`；题库中评委参考框 `expected.boxes[].xyxy` 与选手返回的 `targets[].cx` / `cy` **或** `boxes[].xyxy` 均使用**同一幅题图**的像素坐标系（原点左上，x 向右、y 向下）。训练与推理前请先读题图实际宽高，保证输出与图像像素对齐。

判分语义（大白话）：平台不是拿你的框和标准框算 IoU，也不要求你的框大小和标准框完全一样。它主要看两件事：

1. **类别写对**：比如标准目标是 `笔记本电脑`，你也必须返回 `笔记本电脑`，写成别的类别不算。
2. **中心点落进去**：你返回的目标中心点 `(cx, cy)` 必须落在对应的评委参考框里面。

每个标准目标最多匹配一个预测目标，每个预测目标也最多用一次，不能用同一个点重复拿多个目标的分。

得分按命中比例算：

```
原始得分 = 命中的标准目标数 / 标准目标总数 × 100
最终槽位得分 = 原始得分 / 100 × 当前题型难度满分
```

举例：一张图里标准答案有 10 个目标，你的输出有 6 个目标同时满足“类别对 + 中心点落进对应框”，这次原始得分就是 60 分；如果该槽位满分是 10.2 分，最终就是 6.12 分。

推荐返回方式：优先返回 `targets`，直接给中心点，减少框格式理解错误：

```
{
  "ok": true,
  "result": {
    "targets": [
      { "label": "笔记本电脑", "cx": 980.5, "cy": 769 },
      { "label": "手机", "cx": 376, "cy": 994.5 }
    ]
  },
  "message": "ok"
}
```

也可以返回 `boxes[].xyxy`，平台会自动取框中心点参与判分；但无论哪种方式，坐标都必须是**原图像素坐标**，不要返回 0～1 归一化坐标。

4.3 图像分类 `classify`

本届在线评测平台的**场景分类**固定为下列 **8 类场景**。参赛队训练分类模型时，至少需要覆盖这 8 个场景类别；`/infer` 返回的分类标签必须与下方中文标签一致。

序号	场景类别
1	办公室
2	公园
3	街道
4	商场
5	厨房
6	卧室
7	图书馆
8	体育馆

档位	命题主轴（建议训练数据形态）
L1	3～5 类，类间差异大
L2	更多类、细粒度（相似外观子类）
L3	多标签或含干扰类；需拒识或多头输出策略

五、在线评测单槽满分与判分逻辑（当前系统口径）

在线评测共 **9 道题 / 9 个槽位**，总分 **60 分**。每个槽位先由题型判分器得到 **0～100 的原始得分率**，再乘以该槽位满分。各槽满分如下：

`task_type`	L1 满分	L2 满分	L3 满分
`classify`	4.8	6	7.2
`ocr`	4.8	6	7.2
`detect`	6	7.8	10.2

合计：`classify` 18 分，`ocr` 18 分，`detect` 24 分，总计 **60 分**。

```
单槽得分 = 原始得分 / 100 × 该槽满分
在线评测得分 = 9 个槽位计分成绩之和，最高 60 分
```

同一个 `task_type + difficulty` 槽位如果有多次成功评测，系统取该槽位的**最高得分**计入在线评测总分。

三类题型判分逻辑：

- `classify`：返回的 `result.label` 与标准类别一致得 100，否则 0；标签不在合法类别中也按错误处理。

- `ocr`：平台按题目 `meta.normalize_rules` 对标准文本和模型输出做同样规范化后，完全一致得 100，否则 0。
- `detect`：按“类别一致 + 目标中心点落入对应评委参考框”统计命中比例，不按 IoU 算框重叠率；原始得分为 `命中目标数 / 标准目标数 × 100`。

初赛总分合成：在线答题计入 40 分，在线评测计入 60 分，二者相加为初赛总分。

六、请求里的 `meta` 与「标签」怎么读（实现必读）

平台会在 `/infer` **的** `meta`（及必要时 `draw` 等字段）中给出本题可用的类别名、坐标模式、语言提示、OCR 规范化规则等。选手侧应：

1. **分类** `classify`：使用 `meta.class_names`（字符串数组）作为**合法类别全集**；当前平台固定候选为 **办公室、公园、街道、商场、厨房、卧室、图书馆、体育馆。**返回值中的标签必须落在该集合内（或与平台约定的大小写规则一致），否则可能判为无效标签得 0 分。
2. **检测** `detect`：必须出现 `"coord_mode": "pixel"`（在根 `meta` 或样本 `meta` 中，与平台下发一致）。使用 `meta.class_names`；当前平台固定候选为 **人、汽车、自行车、手机、水杯、笔记本电脑、台灯、沙发、狗**，共 **9 类目标。** `targets` **的** `cx, cy` **或** `boxes` **的** `xyxy` **均为像素**，与当幅题图宽高一致；**不得使用 0～1 归一化坐标。**判分只看“类别是否一致、中心点是否落入对应标准框”，不按 IoU 算框重叠率。
3. **文字** `ocr`：关注 `language_hint`、`normalize_rules`（如是否去空格、英文是否忽略大小写）等；**先对预测串做同一套规范化**再与标准比对。

注意：`request_id`、`session_id`、`task_type`、`image` 为每题必带字段；须在约定超时内返回 JSON，且 `ok` **/** `result` **/** `message` 等顶层结构符合平台《API 与评测说明》。

七、备赛与联调检查清单

- [ ] 三套分支：`classify` / `detect` / `ocr` 均能根据 `meta` 解析类别与坐标规则
- [ ] `detect`：确认请求里 `coord_mode === "pixel"`，且输出坐标与**题图像素尺寸**一致；优先返回 `targets[].label/cx/cy`，中心点落入对应标准框才算命中
- [ ] `GET /health` 中 `supported_tasks` 包含上述三类
- [ ] 为 **L1、L2、L3** 分别准备代表性样本做离线评测，避免只拟合单一难度
- [ ] 控制单张图端到端时延，满足平台 `infer_T_max_ms`（每题可能不同，以请求为准）
- [ ] 异常路径：超时、非 JSON、`ok: false` 时仍能返回可解析结构，便于平台记录原因
