明白了。我将修改现有的 Python 抓取脚本 `robot\_scraper.py`，以去除无效网页并保留包含有效机器人条目的干净 JSON 文件，确保其中每个机器人都有精准的 URL、描述，以及便于应用你提供的八层林奈分类框架所需的信息。

我将尽快处理并给出具体修改建议和解决方案。

# 机器人数据清洗与爬虫改进方案

在对 `robots\_data.json` 进行清理和完善时，需要确保每条机器人记录都有正确的维基百科页面链接、清晰的描述以及关键特征字段。下面将从爬虫修改、数据清洗规则、JSON结构设计以及与林奈分类框架的兼容性几个方面详细说明改进建议。

## 一、`robot\_scraper.py` 修改建议

为提高数据质量，我们建议对爬虫脚本做如下改动：

\* \*\*避免重复抓取和重复条目\*\*：确保同一维基页面只抓取一次，防止出现名称或URL重复的机器人条目。目前脚本使用 `visited\_urls` 集合去重，但仍出现了如“Robot”条目重复三次的情况。建议改进逻辑：根据\*\*最终URL\*\*或\*\*页面标题\*\*去重。例如，在抓取页面后获取其正规化标题（可能通过响应的重定向后URL或页面 `<title>`），将其用于唯一标识。如果发现新的链接指向已抓取过的最终页面，则跳过插入。此外，可以在抓取完成后对 `robots\_data` 进行一次去重检查，移除\*\*名称相同或URL相同\*\*的重复项，保证输出JSON中每个机器人唯一。

\* \*\*严格过滤无关页面\*\*：目前脚本已排除了一些无关链接（如Category、Help、Template、Portal等前缀）。应继续强化过滤规则：

\* 跳过所有\*\*类别页面\*\*（Category）和\*\*列表页面\*\*（List\\_of\\_\\*\\*）：这些页面往往不含具体机器人信息，只是集合分类。例如，曾错误收录了分类页面“Robots”指向维基共享资源分类。应在解析链接时剔除类似链接，确保不将分类页本身作为机器人条目。可通过检查URL中是否包含“Category:”或“List\\_of”来过滤（脚本已有此列表，但需验证其覆盖全面）。

\* 跳过\*\*消歧义页面\*\*：如果链接标题或URL包含“(disambiguation)”或页面内容含有消歧义模板，应忽略。因为消歧义页并非实际机器人信息页面。这可通过检查页面HTML是否含有提示如“may refer to”或消歧义标识来判定。

\* 跳过\*\*非机器人实体\*\*：确保链接文本或内容确实与机器人相关。显示当前逻辑要求标题包含“robot/bot/android/automaton”等关键词。然而，一些真实机器人名称不含这些词（如 \*\*Ballie\*\* 等)，反而可能被遗漏。为兼顾全面性，我们可以在主要机器人分类页上下文中放宽条件：对于类别页面列出的实体，即使名称不含上述关键词，也可以考虑抓取。例如在处理 \*Category\:Robots\* 子页面列表时，可省略 `\_is\_robot\_related` 或 `\_is\_strong\_robot\_name` 的严格过滤，因为该列表已基本限定为机器人相关条目。或者增加额外关键词（如常见机器人名称特征）以减少遗漏。

\* 增加\*\*异常页面检查\*\*：对HTTP状态非200的页面已在代码中跳过。此外，可检查页面内容是否为空或只有几行无实际内容，若提取不到描述则放弃该条目。这避免保留无效页面。

\* \*\*改进描述提取\*\*：目前采用首个长度大于50字符的段落作为描述，并截断500字符。应保持这一逻辑，以抓取\*\*页面首段\*\*的简介性文字。为了保证描述“有意义”，可略作增强：

\* 跳过维基百科页面顶部常见的维护模板或警告（如“本条目有多个问题”）段落，这些通常以短句或HTML提示框出现。当前通过长度判断已基本避开这些内容。

\* 确保如果首段包含引用标号或额外内容，也一并保留（目前实现已保留引用符号作为纯文本）。如有需要，可以去除描述中的引用标记数字或“needs update”等编辑提示，以提升可读性。

\* 对于类别页面的描述（如出现“This category has the following X subcategories…”），应识别并跳过，因为这些不是真正的机器人描述。如果检测到描述文本包含 “This category has” 或 “The following pages are in this category” 等模式，则判定当前页面是分类页，不应作为机器人条目保留。

\* \*\*丰富和校准信息提取\*\*：当前脚本从页面信息框（infobox）中提取 \*\*manufacturer\*\*、\*\*year\*\*、\*\*applications\*\* 三类信息。为生成更完善的“特征”字段，建议如下：

\* \*\*厂家/制造商\*\*：保持提取`manufacturer`字段（涵盖制造商或开发者信息），并兼容不同表述，如有的infobox字段可能写作 "developer"、"inventor"、"created\\_by" 等，可增加判断包含这些关键词时也归入制造商字段。

\* \*\*年份\*\*：目前代码抓取包含“year”或“introduced”的字段作为`year`。不同条目可能用不同表述，例如“First launch”、“release date”、“developed (year)”等。可扩展匹配条件，包含`launched`, `released`, `developed`, `built` 等关键词。同时注意，有些年份信息可能不是纯数字（可能是时间段或具体日期），可以视需求决定是保留原文本还是进一步解析为年份。

\* \*\*应用领域\*\*：凡infobox字段含“application”、“use”或类似词汇，都加入`applications`列表。需要注意有的条目可能列出多个用途，通常在HTML中以 `<br>` 分隔。当前实现使用 `get\_text().strip()` 获得所有用途文本，会把多个用途连在一起。建议\*\*拆分多个应用\*\*：例如按换行、逗号或分号分割，使 `applications` 为字符串列表，每项是一个用途领域。这样数据更结构化，利于后续映射。

\* \*\*规格参数\*\*：代码预留了`specifications`字典，但未填充。可考虑从infobox提取更多技术规格（如尺寸、重量、负载能力、电源等)，存入`specifications`。实现上，可针对常见字段（如 "Mass", "Weight", "Dimensions", "Speed", "Power" 等）提取键值对。如果暂不需要详细规格，也可将 `specifications` 字段移除或留空以减少冗余。

\* \*\*类别信息\*\*：目前`category`字段始终为空。如果需要，可用此字段记录机器人\*\*类型或来源类别\*\*。例如，可以记录此机器人来源于哪个维基分类页（如Industrial, Service等），或者记录其在Wiki中所属的重要分类。实现上，可在抓取链接时附带当前处理的`search\_term`标签，或者直接解析页面底部的Category标签，将第一个与机器人类型相关的类别填入。但由于后续将引入自定义的分类框架，此字段也可不再使用，避免混淆。\*\*建议\*\*：暂时移除`category`字段或留空不填，改由后续的林奈分类字段来表示类别归属。

\* \*\*递归抓取完善\*\*（可选）：现有爬虫仅抓取固定的4个页面（总体分类及工业/服务机器人）。这覆盖面有限，可以考虑在这些起始分类下\*\*递归抓取子分类\*\*，扩充机器人列表。例如，从“Category\:Robots”页面获取子分类链接（如“Humanoid robots”、“Medical robots”等），再抓取这些分类页面内的机器人链接。递归时注意控制深度，避免抓取过深导致非机器人内容，以及继续运用上述过滤规则。通过拓展数据来源，可获取更加全面的机器人条目清单。

以上修改将使爬虫抓取的数据更加干净、准确，为后续处理打下基础。

## 二、数据清洗与筛选规则

抓取完成后，需要对 `robots\_data.json` 进行清洗，删除无效或重复的数据。建议采用以下规则：

\* \*\*有效页面校验\*\*：保留HTTP状态码为200且内容正常的条目，删除任何状态异常或解析失败的记录。进一步地，若某条目的`description`为空（说明未提取到正文段落），也应移除，因为这通常意味着该页面不是有效的机器人介绍页（可能是重定向页或空白页）。

\* \*\*移除重复条目\*\*：通过`name`和`url`共同判定重复。优先依据`url`去重，因为同一个维基页面可能以不同名称出现（如“Robot”与“robots”都指向同一URL）。可以对JSON数组按URL去重，仅保留第一个出现的条目。此外，如发现两个条目URL不同但名称相同（例如大写/小写或同名不同条目），可酌情检查是否实际上是同一机器人，否则也应去重或重命名以区别。总之，确保导出JSON中每个机器人（以URL计）仅一条记录。

\* \*\*剔除非机器人项\*\*：清除任何不是具体机器人的条目，包括：

\* \*\*分类页\*\*：如前述出现的“Robots”分类记录，此类记录的`description`通常包含“subcategories”字样，可作为识别依据，应从结果中删除。

\* \*\*列表页\*\*：若有残留的“List of ...”条目，也应删除，因为这些不是单一机器人。

\* \*\*纯概念或作品\*\*：个别抓取结果可能是机器人相关作品（如纪录片《The Truth About Killer Robots》）或概念（如“March of the Machines”书籍）。它们虽然名字里含Robot，但并非机器人实体，应人工识别后移除。这类页面通常在描述中能看出是电影/书籍等（例如包含“film”或“novel”字样）。

\* \*\*人名或项目\*\*：有些链接带“(project)”等，需判定是否真的机器人。如果只是机器人计划或研究项目而非机器人本体，可以考虑去除或标记。

\* 如果不确定某条是否为机器\\*\\*（例如名字不直观）\\*\\*，可以依据维基页面内容判断：若描述第一句没有表述“是一个机器人”或类似定义，则可能不是实体机器人，应考虑剔除。

\* \*\*字段完整性检查\*\*：对于保留下的每条记录，尽可能\*\*保证关键字段非空\*\*：

\* `name` 和 `url` 必须存在且对应正确；

\* `description` 应该有内容（如果某机器人简介特别短，比如只有一两句话，也应保留，但如果仅有如“X is a robot”一句话，需确认页面是否过于简略）；

\* `manufacturer`, `year`, `applications` 等字段没有的话可为空，但建议在清洗时\*\*统一格式\*\*：例如确保 `applications` 始终是数组类型（即使为空也写作 `[]`），字符串字段未提取到内容则为 `""` 空字符串。统一的数据格式方便后续处理。

\* 可以考虑\*\*删除多余字段\*\*：如果在爬虫修改中决定移除`category`或`specifications`，应从JSON结构中删去这些空字段，使结构更简洁。

\* \*\*内容规范化\*\*：对提取的文本进行必要的规范化处理：

\* 去除描述和其他字段中的参考文献标号（如“\[1]\[2]”）或维基特殊标记（如“\[needs update]”）。这些可以在清洗阶段用正则去掉，以提高可读性。

\* 修正一些常见格式：比如年份字段如果包含额外说明（“circa 2000”或“\~2000年”），可选地转换为标准格式（如仅保留“2000”）。制造商字段如含有括号（公司额外信息）也可酌情精简。

\* \*\*多值拆分\*\*：正如前述，对 `applications` 字段尤其要拆分成列表的，在清洗中可检查字符串中逗号、分号来拆分多个应用项。如果infobox里多个应用挤在一个字符串，可在清洗时进行分割整理。

经过上述清洗筛选，最终保留下来的 JSON 数据将仅包含\*\*真实的机器人\*\*及其主要属性，为下一步分类映射做好准备。

## 三、导出 JSON 结构设计

为了便于下游使用，输出的 JSON 文件应结构清晰、一致。建议采用每个机器人一个JSON对象的形式，字段包括：

\* \*\*name（名称）\*\*：\*字符串\*，机器人名称。例如 `"Atlas (robot)"`。保持与维基百科条目名一致，必要时包含括号中的区分说明。

\* \*\*url（维基链接）\*\*：\*字符串\*，对应维基百科英文页面的完整URL。例如 `"https://en.wikipedia.org/wiki/Atlas\_(robot)"`。应确保链接有效可打开。

\* \*\*description（简介）\*\*：\*字符串\*，来自维基百科页面首段的简要描述。例如：`"Atlas是波士顿动力公司研发的人型仿生机器人，擅长在野外环境行走和操作物体..."`（示例）。该描述提供机器人基本定义和用途，长度适中（几百字以内）。

\* \*\*manufacturer（制造商）\*\*：\*字符串\*，机器人主要制造商、研发机构或设计者。如 `"Boston Dynamics"`。若无明确制造商（如学术原型或概念），可为空字符串或标记为 `"Unknown"`。

\* \*\*year（年份）\*\*：\*字符串/数字\*，机器人推出或制造年份。建议统一为字符串格式保存，以包含可能的说明。例如 `"2013"` 或 `"2013–2017"`，`"约2010年代"` 等。如果没有年份信息则为空。

\* \*\*applications（应用领域）\*\*：\*数组\*，元素为字符串，每项是机器人用途或应用场景。例如 `["灾难救援", "侦察"]`。这些信息源自infobox的应用字段或页面内容，可多值。若无明确应用则给空数组。通过数组存储便于一台机器人对应多个用途标签。

\* \*\*specifications（规格参数）\*\*：\*对象/字典\*，包含可选的技术规格键值对。例如：`{"weight": "75 kg", "height": "1.5 m", "power": "Hydraulic"}` 等。只有在能够提取到具体参数时才加入该字段，没有则可以省略或为空对象 `{}`。这一字段主要供将来需要更详细技术比较时使用。

此外，为了支持后续的林奈八层分类映射，可以考虑在JSON结构中预留或直接包含分类字段。例如，可在每条记录增加一个嵌套的 `taxonomy` 对象，或直接增加多个平级字段：

\* \*\*domain（域）\*\*：机器人存在的主要环境域，可能值：`"Physical"`（物理）、`"Virtual"`（虚拟）、`"Hybrid"`（混合）。

\* \*\*kingdom（界）\*\*：机器人的应用领域类别，如工业、医疗、服务、军用等。例如 `"Industrial"`、`"Medical"` 等。

\* \*\*morpho\\_motion\\_class（纲：形态-运动类别）\*\*：机器人的形体结构和运动方式，如`"Wheeled-Mobile"`（轮式）、`"Legged-Humanoid"`（人形双足）、`"Flying-Drone"`（飞行器）等。

\* \*\*order（目：自主性等级）\*\*：控制方式/自主程度，如`"Manual"`（手动）、`"Autonomous"`（自主）、`"Teleoperated"`（遥控）等。

\* \*\*sensing\\_family（科：主要传感类型）\*\*：机器人主要感知方式，例如`"Vision-Based"`（视觉）、`"LiDAR-Based"`（激光雷达）等。

\* \*\*actuation\\_genus（属：驱动方式）\*\*：主要驱动/执行机构类型，如`"Electric"`（电动）、`"Hydraulic"`（液压）、`"Pneumatic"`（气动）等。

\* \*\*cognition\\_class（类：智能水平）\*\*：机器人的智能/计算类型，例如`"Rule-Based"`（基于规则）、`"AI-Powered"`（人工智能驱动）等。

\* \*\*application\\_species（种：具体用途）\*\*：机器人执行的具体任务功能，支持多标签，如 `["Exploration", "Mapping"]` 等。这通常与前述applications字段相关联，但这里会映射到预定义的标准类别名称。

上述分类字段建议以\*\*空值或占位符\*\*初始化在JSON结构中，或者至少为每条记录保留这些字段。最终导出的JSON可以采用如下嵌套结构示例：

```json

{

"name": "Atlas (robot)",

"url": "https://en.wikipedia.org/wiki/Atlas\_(robot)",

"description": "Atlas是波士顿动力公司研发的人型仿生机器人...",

"manufacturer": "Boston Dynamics",

"year": "2013",

"applications": ["灾难救援", "军事后勤"],

"specifications": {

"height": "1.8 m",

"weight": "80 kg"

},

```

或者不使用嵌套，直接以平级字段列出分类（视使用方便性决定）。无论哪种方式，JSON结构都应清晰表达机器人基本信息和分类信息，并且易于程序读取和操作。

## 四、未来与林奈分类框架的对接兼容

为将清洗后的机器人数据集映射到用户提供的八层林奈分类框架，需要做好以下准备和考虑：

\* \*\*预留分类字段，方便映射填充\*\*：正如上节所述，在JSON结构中直接包含 `domain`、`kingdom` 等八层分类字段（哪怕初始为空），有助于后续直接填充分类结果，而不需要对数据结构再做大改动。一旦确定分类结果，这些字段即可就地更新。这种设计与分类框架的JSON模板相匹配，确保兼容。

\* \*\*建立映射策略\*\*：自动或半自动地将每个机器人条目映射到相应的分类：

\* 可以利用\*\*维基百科的类别标签\*\*作为线索。许多机器人页面底部会有类别如“Industrial robots”、“Medical robots”、“Humanoid robots”等。比如，若页面属于“Industrial robots”类别，则其 `kingdom` 映射为“Industrial”；属于“Humanoid robots”类别，则 `morpho\_motion\_class` 可标记为“Legged-Humanoid”。脚本可抓取每个页面的Categories列表，并依据关键词匹配到框架术语。

\* 利用\*\*已提取的信息\*\*：如infobox的应用字段（applications）可以帮助映射`application\_species`。例如，如果某机器人applications包含“surgery”或“手术”，可推断其 `application\_species` 包含 “Surgery”；如果包含“bomb disposal”或“排爆”，可映射为 “Rescue”。可以制定一个关键词映射表，将常见用途词汇归类到框架定义的类别上。

\* 制造商或使用场景有时也能提供线索。例如制造商是农业设备公司的，可能 `kingdom` 为 “Agriculture”；服务型机器人的产品通常 `kingdom` 为 “Service”。这些关联可用于辅助判断，但需要人工校验以避免偏差。

\* \*\*默认值和人工校正\*\*：并非所有分类都能由现有信息自动推断，例如`domain` 几乎总是“Physical”（绝大多数机器人有物理形态，除非明确是纯软件机器人），可以默认标注为 Physical；`cognition\_class`（认知等级）通常需要根据机器人是否有自主智能、是否搭载AI等决定，维基资料可能未明确，需要后续人工补充。对于难以自动判定的层级，建议预留为 `"Unknown"` 或空，在人工审核时填入。

\* \*\*确保多标签处理\*\*：分类框架的第八层“应用种”允许多重归类，这在JSON中已使用数组表示。映射时要支持一台机器人属于多个应用类别。例如一个送货机器人可能既用于物流运输又用于餐饮服务，应在 `application\_species` 列出 `["Transport", "Service"]` 等。我们的数据结构和映射逻辑需能处理这种一对多关系。

\* \*\*分类一致性和层级约束\*\*：林奈式框架层层递进，每层分类应与相邻层逻辑相关。例如 `kingdom: Industrial` 的机器人，其 `application\_species` 不应标记为“Companionship”（陪伴）等完全无关的类别。在批量映射时要注意这种一致性，可能通过预设规则或人工检查实现。确保最终每条机器人的八层分类组合在逻辑上是合理的（框架本身没有硬性约束组合，但从常识出发要匹配）。

\* \*\*数据更新维护\*\*：林奈分类框架可能随研究深入调整定义或增加类别。我们的JSON设计应能\*\*兼容扩展\*\*——例如如果以后扩展了第九层、第十层分类，也可以方便地在每条记录中增加字段而不破坏结构。目前采用明显的字段名可以减小歧义。如果框架调整某些名称（比如改叫 “Mode” 而非 “Domain”），只需相应更新字段名即可。在生成最终JSON时，建议在文件头部或文档中注明所采用的分类版本，以方便日后维护同步。

通过上述措施，清洗后的机器人数据将能顺利地映射到“机器人生命之树”八层分类框架中。这样，每条机器人记录不仅有基础信息，还对应一个完整的分类标签集合，方便用于\*\*机器人分类展示、聚类分析\*\*以及\*\*构建大型机器人分类本体\*\*等应用。

综上所述，我们通过修改爬虫确保数据源干净准确，应用严格规则清洗无效条目，设计合理的数据结构，以及提前布局分类字段，以实现对接林奈式八层分类体系的目标。这将极大提升数据集的质量和可用性，为后续的机器人谱系研究和可视化奠定坚实基础。