📚 SayCan 论文精读笔记

项目	内容
论文标题	Do As I Can, Not As I Say: Grounding Language in Robotic Affordances
作者	Michael Ahn, Anthony Brohan, Noah Brown 等 (Google Robotics)
发表时间	2022年 (arXiv: 2204.01691)
阅读日期	2025年1月
阅读目的	理解如何将 LLM 的语言理解能力与机器人实际执行能力结合

📌 一句话总结

SayCan 通过将 **LLM 的语义理解能力**（知道"应该做什么"）与 **机器人的 Affordance**（知道"能做什么"）**相乘**，实现了将抽象自然语言指令分解为可执行的具体动作序列，是具身智能领域 **Grounding（落地）** 的开创性工作。

🔑 核心公式（全文只有这一个！）

$$\pi^* = \arg\max_{\pi \in \Pi} \underbrace{p(c_\pi | s, \ell_\pi)}_{\text{Affordance（能做到）}} \times \underbrace{p(\ell_\pi | i)}_{\text{LLM（该做啥）}}$$

符号解释表

符号	含义	机械臂场景举例
i	用户输入的高层指令	"把那个脏的零件挪开"
Π	机器人会的所有技能集合	{抓取, 放置, 移动到A点, ...}
ℓ_π	某个技能的语言描述	"pick up the dirty part"
s	当前环境状态（如摄像头画面）	桌面上有3个零件，1个脏
$p(\ell_\pi \| i)$	LLM：这个技能"对不对"（语义匹配度）	LLM觉得"抓脏零件"符合指令 → 0.9
$p(c_\pi \| s, \ell_\pi)$	Affordance：机器人"能不能"做到	脏零件在臂展内 → 0.8
最终得分	两者相乘	0.9 × 0.8 = 0.72 ✓ 选中

💡 核心思想：Say × Can

问题：LLM 知道很多常识，但不知道机器人"能不能做到"

例子：用户说"我打翻饮料了，帮帮我"

方法	回答	问题
GPT-3	"You could try using a vacuum cleaner"	机器人没有吸尘器
LaMDA	"Do you want me to find a cleaner?"	只是聊天，不会行动
FLAN	"I'm sorry, I didn't mean to spill it"	完全跑偏
SayCan	1.find sponge → 2.pick up → 3.bring to you → done	✅ 可执行的步骤

📐 方法详解：SayCan 的工作流程

工作流程（对应论文 Figure 3 和 Algorithm 1）

步骤	操作	说明
1	接收高层指令 i	用户说："I spilled my drink, can you help?"
2	遍历所有技能 Π	机器人会的技能：{find X, pick X, go to Y, put down X, done}
3	LLM 打分 $p(\ell_\pi \| i)$	对每个技能，问 LLM：这个动作对解决问题有帮助吗？
4	Affordance 打分 $p(c_\pi \| s, \ell_\pi)$	用 Value Function 评估：当前状态下这个动作能成功吗？
5	相乘选最优	选择 Say × Can 得分最高的技能执行
6	执行并迭代	执行选中的技能，将其加入历史，回到步骤3，直到选中 done

🎯 关键概念：Affordance（可供性）

来源：心理学家 Gibson 提出的概念——环境"允许"你做什么

生活例子：

- 椅子的 affordance 是"可以坐"
- 门把手的 affordance 是"可以转动"

- 杯子的 affordance 是"可以抓握、可以装水"

在 SayCan 中：用 Value Function（价值函数） 来量化 Affordance

Affordance 计算方式

技能类型	Affordance 计算方式	说明
Pick（抓取）	RL 训练的 Value Function	归一化到 [0,1]，见论文 Appendix D.2
Go to（导航）	基于距离的启发式	距离越近，affordance 越高
Place（放置）	固定为 1.0	假设抓住了就能放下
Terminate（终止）	固定为 0.1	小概率，防止过早结束

🧠 LLM 如何打分？（Prompt Engineering）

SayCan 使用 **Scoring LM**（打分语言模型），而不是让 LLM 直接生成文本：

1. **构造 Prompt**：包含任务示例 + 对话格式 + 当前指令
2. **对每个候选技能**，计算 LLM 输出该技能描述的概率
3. **概率高** = LLM 认为这个技能"应该做"

关键：这不是让 LLM 自由生成，而是 **受限解码**——只能从预定义技能中选择

🔗 与我课题的关系

课题方向：面向工业机械臂的模糊指令语义消歧

SayCan 的做法	对我课题的启发	待解决的问题
LLM 理解高层指令	用 LLM/VLM 理解"那个有点脏的"	如何处理更模糊的工业术语？
Affordance 过滤不可行动作	用机械臂运动学约束过滤	工业场景的 Affordance 如何定义？
技能集合 Π 是预定义的	需要设计工业装配的技能库	技能粒度如何确定？
迭代选择下一步	复杂装配任务的分步规划	失败后如何回溯？

⚠️ SayCan 没解决的问题（也是我可以做的方向）

1. **没有视觉定位**：SayCan 假设物体位置已知，但我的场景需要先定位"那个脏零件"

2. **没有消歧机制**：如果有两个脏零件怎么办？SayCan 无法处理
3. **没有闭环反馈**：执行失败后无法动态调整（**Inner Monologue 解决了这个问题**）

❓ 我的疑问

概念层面

1. **Affordance vs Feasibility**：Affordance 强调"物体允许什么操作"，Feasibility 强调"机器人能不能做"，SayCan 似乎混用了这两个概念？
2. **技能粒度问题**：SayCan 的技能是 "pick up the apple" 这种级别，如果是更细粒度的 "rotate gripper 30度"，LLM 还能有效打分吗？

技术层面

3. **Value Function 的泛化性**：RL 训练的 Value Function 在新物体上效果如何？论文没有详细讨论。
4. **LLM 打分的可靠性**：不同 LLM（GPT-3 vs PaLM）打分差异大吗？论文 Table 3 显示 PaLM 比 FLAN 好很多。

实验层面

5. **技能数量的扩展性**：论文只有约 550 个技能，如果是上千个技能，遍历打分的效率如何？
6. **Real-world 部署**：论文在厨房场景做实验，工业场景（更高精度、更多约束）是否适用？

📊 关键图表索引

图表	页码	内容	重要性
Figure 1	p.1	SayCan vs 纯 LLM 的对比，一图看懂核心思想	⭐⭐⭐
Figure 2	p.3	Value Function 可视化，展示 Affordance 如何随状态变化	⭐⭐⭐
Figure 3	p.4	SayCan 完整流程图，包含 LLM 打分 + Affordance 融合	⭐⭐⭐
Figure 8	p.11	开源 Colab 环境截图，可以跑 demo	⭐⭐
Figure 12	p.22	Prompt Engineering 详解，如何构造 LLM 输入	⭐⭐
Algorithm 1	p.4	SayCan 伪代码，可以直接对照实现	⭐⭐⭐
Table 3	p.9	不同 LLM 的性能对比，PaLM 84% vs FLAN 70%	⭐⭐

📖 术语表

术语	中文	解释
Affordance	可供性	环境/物体允许什么操作（来自心理学）
Grounding	落地/接地	把抽象语言对应到具体物体、位置或动作
Value Function	价值函数	RL 中评估从当前状态能获得多少回报的函数
Scoring LM	打分语言模型	计算 LLM 输出某段文本的概率，而非自由生成
Skill / Primitive	技能/原语	机器人能执行的原子级动作（如 pick, place）
Task Planner	任务规划器	把高层指令分解为技能序列的模块
BC (Behavioral Cloning)	行为克隆	直接模仿专家动作的监督学习方法
MT-Opt	多任务优化	Google 的多任务 RL 训练框架
Long-horizon Task	长程任务	需要多步骤完成的复杂任务

📝 阅读建议

部分	建议	原因
Abstract + Figure 1	必看（10分钟）	快速抓住核心思想
Section 3 (Method)	精读（30分钟）	理解 SayCan 的数学原理
Section 4 (Implementation)	精读（20分钟）	理解技能和 Affordance 如何实现
Section 5 (Experiments)	快速浏览	看 Table 3 的数字就够了
Section 7 (Related Work)	跳过	以后写论文再回来看
Appendix D	选读	想跑代码时再看 Prompt 细节

🔗 与下一篇论文的关系

Inner Monologue 是 SayCan 的直接后续工作，解决了 SayCan 的一个关键问题：

对比	SayCan	Inner Monologue
执行方式	**开环执行**——选完动作就执行，不管成功与否	**闭环反馈**——执行后获取环境反馈，动态调整计划
反馈机制	无	成功检测器、场景描述器等
失败处理	无法处理	可以重新规划

Inner Monologue 在 SayCan 基础上加入了"成功检测器"、"场景描述器"等反馈模块，让机器人能够"思考"执行是否成功，并据此调整下一步行动。

笔记完成时间: 2025年1月

开源代码: say-can.github.io/#open-source