LLM-BRAIn: AI-driven Fast Generation of Robot Behaviour Tree based on Large Language Model

Artem Lykov
Department of Engineering Systems
Skolkovo Institute of Science and Technology
Moscow Oblast, 121205, Russia
artem.lykov@skoltech.ru

Dzmitry Tsetserukou
Department of Engineering Systems
Skolkovo Institute of Science and Technology
Moscow Oblast, 121205, Russia
d.tsetserukou@skoltech.ru

Abstract

本文提出了一种名为 LLM-BRAIn 的自主机器人控制新方法，该方法可以根据操作员的指令生成机器人行为。LLM-BRAIn 是一个基于 Transformer 的大型语言模型（LLM），由斯坦福 Alpaca 7B 模型微调而来，用于从文本描述生成机器人行为树（BT）。我们使用通过 text-davinchi-003 以 self-instruct 风格生成的 8500 个指令遵循演示来训练 LLM-BRAIn。所开发的模型能够准确构建复杂的机器人行为，同时其规模小到可以在机器人的机载微型计算机上运行。该模型能生成结构和逻辑正确的行为树，并能成功处理训练集中未出现的指令。实验未揭示由 LLM-BRAIn 生成的行为树与人类创建的行为树之间存在任何显著的主观差异（平均而言，参与者在 10 次中有 4.53 次能够正确区分 LLM-BRAIn 生成的行为树和人类创建的行为树，这表明他们的表现接近于随机猜测）。该方法有望应用于移动机器人技术、无人机操作、机器人操纵器系统和工业 4.0。

1 引言

近年来，高性能的大型语言模型（LLM）的发布，如开源的斯坦福 Alpaca 模型，为相关技术领域的研究人员带来了巨大的可能性。这主要得益于基于 transformer 的大型语言模型能够执行遵循指令的任务。目前已明显可以预见，类似 GPT 的模型将很快在“人-机器人交互”（human-robot interaction）领域得到广泛应用。在这一背景下，值得强调的是该技术的融合对机器人领域尤为重要，因为它对 AI 驱动的机器人系统的开发与实现具有深远影响。此外，机器人与人之间的交互问题也日益受到关注。

对机器人指令与逻辑的形式化探索由来已久。此类工作的优先目标是使非技术人员能够以便捷且逻辑清晰的方式在工作中控制机器人并与其交互。行为树（behavior tree, BT）方法是一种先进的机器人逻辑描述方法。它允许将复杂的机器人行为由简单的模块（节点）组合而成。此外，BT 具备模块互换性，并保持一种清晰明了的结构，可被存储为 XML 文件。现代基于 transformer 的 LLM 非常适合根据指定逻辑生成严格结构化的文本文件，甚至能够完成程序代码的生成，这是更具挑战性的任务。

Preprint. Under review.

本文提出了一种新的自主机器人控制方法，使用基于 transformer 的 LLM，通过人类操作员的语言描述，生成符合 BT 格式的复杂机器人行为。行为的生成依赖于对 LLM 的微调，并使用预编写节点的库。

2 相关工作

近年来，自 transformer 模型在开创性论文《Attention is all you need》中被 Vaswani et al. [1]首次提出以来，已在多个领域中获得广泛应用。这些 transformer 模型在语言建模、翻译和语音识别等任务中表现出色。

特别是基于 transformer 架构的 LLM，在近年来引起了极大的关注。这一趋势受到 OpenAI 的 GPT2 和 GPT3 等模型推动，其特性分别在 Radford et al. [2]和 Brown et al. [3]中进行了讨论。OpenAI [4]提出的 ChatGPT 极大地推动了 GPT 模型的普及，使得更广泛的受众得以接触和使用该技术，从而使其受欢迎程度达到了新高度。这些模型展示了生成连贯、语境恰当文本的能力，因而非常适合用于广泛的自然语言处理（NLP）任务。

在一段时间内，只有与 OpenAI 合作的人员才能参与 GPT 模型的开发并见证其非凡的成果。但随着 Google 的 T5 模型（见 Raffel et al. [5]）和 Meta 公司最近提出的 LLaMa（见 Touvron et al. [6]）等 GPT 类 LLM 模型的出现，情况发生了变化。这些模型为不受单一机构控制的替代方案提供了可能，并已被广泛用于多种 NLP 应用，如语言生成、问答、情感分析等。

众多实验室和行业机构在获得 LLM 后，积极探索使用 GPT 或其变种来构建机器人行为的方法。例如，Driess et al. [7]提出的 PaLM-E 是一种具身多模态语言模型，使机器人能够遵循指令执行任务。另一个工作 Brohan et al. [8]则提出了一种面向大规模现实控制的机器人 transformer。这两者都是规模庞大的模型，训练和使用都需要大量计算资源和数据。除此之外，波士顿动力公司（Boston Dynamics）也在探索将 GPT 用于构建机器人行为的可能性。他们成功地训练了机器狗 Spot 使用 GPT 汇报工作结果，详见 Petkauskas [9]。

此外，Taori et al. [10]提出的斯坦福 Alpaca 模型的公开发布，成为了全球 LLM 研究的一个转折点。Wang [11]的仓库提供了一套完整的资源，供研究人员以斯坦福大学相同的方法微调 Alpaca 7B 模型，包含所有必要组件与操作说明。与其他 LLM 不同的是，斯坦福 Alpaca 模型不仅达到了接近 GPT3 的性能，且具备能够在普通个人计算机上运行的特性。该模型的架构支持指令跟随类任务的微调，这为相关技术领域的研究人员带来了巨大可能性。

另一个工作 Cao and Lee [12]提出使用 GPT 生成机器人行为。实际上，该研究并未直接使用 LLM 生成完整 BT 结构，而是借助 OpenAI 产品将行为节点填充到一个固定结构的 BT 中。由于 OpenAI 的产品并非专为机器人行为生成而设计，研究人员只能采用固定 BT 结构，仅使用 sequence 和 action 节点。这种方法虽限制了 BT 模块化结构的优势，但实验结果依然具有前景。尽管所提出的 BT 构建方法是有效的，但若针对该任务对 LLM 进行专门微调，通常可以获得更优效果，从而生成不受上述限制的各种 BT。

然而，要想对 LLM 进行微调以生成复杂的 BT，仍需构建一个涵盖多种结构和应用场景的丰富多样的 BT 数据集。虽然如 Stiennon et al. [13]所述的“基于人类反馈的强化学习”（Reinforcement Learning from Human Feedback, RLHF）方法广受认可，但生成如此大规模数据集通常需要具备 BT 构建能力的专家，任务极具挑战性。因此，本研究采用了 Wang et al. [14]在 Taori et al. [10]研究中提出的“self-instruct”方法，使用 text-davinci-003 模型构建数据集。最终的微调模型可在特定任务上超越用于生成该数据集的原始模型，因为其经过专门训练以解决这些任务。

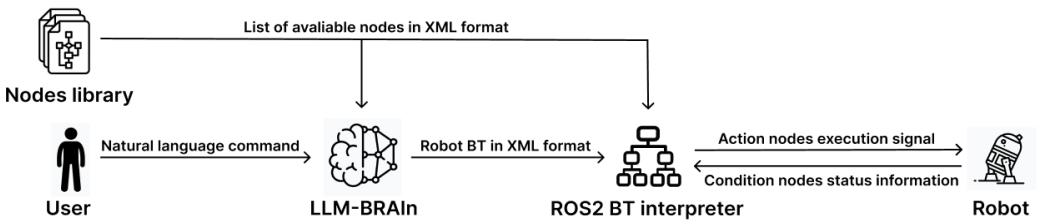

Figure 1: 系统架构。

3 系统概述

系统架构如图 1所示。在预期使用场景中，所需的硬件必须包括一台能够运行经过再训练的 LLM-BRAIn 模型（拥有 70 亿参数）的板载微型计算机。硬件还包括机器人执行其节点库中描述任务所需的所有传感器、执行器和机械组件。软件系统包括一个运行 LLM-BRAIn 模型的应用程序、一个将其转换为可执行文件的 BT 解释器（基于已成为机器人标准的 ROS2，参见 Macenski et al. [15]），以及一个列出机器人可执行动作的节点库。

操作员可以用自然语言向机器人下达指令。随后，该指令会被构造成模型请求，并附加来自节点库的可用节点列表。模型处理该查询后，其输出在转换特殊字符后即为一个以 XML 格式表示的 BT。过去这类 BT 是由人手工编写的，但如今该方法因其通用性和灵活性而在机器人领域广泛应用。基于 BehaviorTree.CPP 库的命令执行作为一个 ROS2 节点实现，库可见于 Faconti and Colledanchise [16]的仓库中。

4 使用 LLM 生成的 BT 定义机器人行为

4.1 在机器人中使用行为树的优势

行为树（BT）是一种用于在抽象层次表示机器人任务的分层结构，作为状态机范式的替代方案。构建机器人行为的 BT 方法在 Colledanchise and Ögren [17]中有所讨论。形式上，BT 是一个有向根树，其叶节点负责任务执行，分支节点定义控制流程逻辑。叶节点包括 Action（动作）和 Condition（条件）节点：前者指定一个基本任务并在完成后返回 Success 信号，后者用于评估布尔条件，例如某一传感器读数是否满足。常见的分支节点包括 Sequence（顺序）节点和 Fallback（回退）节点。Sequence 节点依次执行其子节点，直到接收到第一个 Failure 信号；Fallback 节点则依次执行子节点，直到接收到第一个 Success 信号。仅使用这四种节点类型，BT 即可完成与状态机等价的任务执行，但同时具有诸多优势。例如其模块化结构允许添加、移除、替换节点而无需重构整个结构。

由于其直观性与高效性，BT 已成为机器人系统控制中的广泛方法。它提供了一种结构化的方式来表示与控制自主体的行为，适用于多种机器人应用场景。

BT 已在许多机器人竞赛和挑战中成功应用，包括 DARPA 机器人挑战赛、RoboCup 和 Eurobot。例如，在 DARPA 挑战中，团队使用 BT 控制机器人完成如驾驶、开门、使用电动工具等任务，详见 Colledanchise and Ögren [18]；在 RoboCup 中，BT 被用于控制多机器人系统完成如足球比赛、搜索与救援等任务，参见 Safronov et al. [19]；在 Eurobot 中，BT 被用于控制机器人完成如迷宫导航、物体操作等任务，见 Granosik et al. [20]。

4.2 将行为树作为 LLM 输出的优势

行为树作为一种分层、模块化结构，为基于 transformer 的 LLM 输出提供了一种有前景的解决方案。其结构允许替换同类型节点，令其非常适合作为 transformer 的输出结构。此外，BT 的模块性与可扩展性也支持轻松添加新节点或修改已有节点。

BT 结构的另一显著优势在于可将子树用作主树的一部分。这意味着已生成的 BT 可被加入节点库，作为更复杂行为的一部分。在递归模式下，模型可以首先在高层抽象层级构建 BT，然后向低层级递归生成缺失节点，从简单组件构建完整节点。这种方法支持在模型输出 token 长度限制内构建大型复杂行为结构，从而提升模型性能，解除最终 BT 规模的限制。

5 数据集收集

5.1 数据集表示格式

我们来考虑如何构建用于训练 LLM 生成 BT 的数据集。虽然斯坦福 Alpaca 模型逐词生成文本，其训练数据集由完整文本组成，但在执行指令跟随任务时，文本被划分为三部分：“instruction:”、“input:”和“output:”。其中“input:”部分是可选的，仅在斯坦福原始工作中的部分样本中使用。在生成 BT 任务中，我们未使用此元素。因此，每个数据样本由构建机器人行为的指令和一个逻辑与结构正确的 BT（作为输出）组成，使机器人能执行该任务。该指令包含一个通用部分：“Write a behavior tree for the robot to execute the command using only available nodes.”。该部分在后续拓展任务（如纠错、子树构建、解释某一行为等）中是必需的。在这段不可更改的指令之后，是对所需机器人行为的描述。在使用 text-davinci-003 模型生成样本时，对指令的自然性尤为重视，应为一条人类可能发出的简单、逻辑清晰的命令，因为微调后的模型将用于响应人类指令。指令的最后一部分是机器人可执行的 Action 与 Condition 节点列表。这是指令中的关键元素，因其限制与指定了可用节点，从而保证生成的 BT 可在目标机器人上执行。除 Action 与 Condition 节点外，节点库有时也包含 SubTree 节点。SubTree 节点与 Action 节点类似，但自身是已编译的 BT。引入 SubTree 节点支持分阶段生成机器人行为的整体逻辑，从可用节点中构建缺失部分。此方法避免 LLM 一次性生成大型结构，避免了由于序列长度增加而导致内存需求呈二次增长的问题。

5.2 使用 Text-Davinci-003 生成数据集

如前所述，OpenAI 的产品并非专为根据描述生成 BT 而设计。然而，text-davinci-003 模型具备以 XML 格式生成具有不同结构的机器人随机行为 BT 的能力，这一能力成为数据集构建的起点。在第一个请求中，我们要求模型生成一个 BT。经验表明，该请求需要额外说明，如该 BT 应可由移动机器人执行、具和平应用场景、包含描述运动与物体操作的元素。这些描述可根据需求拓展到其他机器人类型。此外，我们发现 text-davinci-003 可能会生成结构或逻辑上的错误，经过多轮优化请求，我们得以规避这些问题。第二个请求中，我们要求 text-davinci-003 创建构成 BT 的节点库，并提供期望的响应结构样例。第三个请求中，我们要求其基于生成的 BT 输出口语化的机器人行为描述，该任务模型执行效果良好。最后，我们调试上述三个请求，并使用 OpenAI 的 API 将生成结果填入样本对应位置以构建数据集。按照此方法，我们生成了三个数据集：1000 条样本、5000 条样本和 8500 条样本。衡量此类数据集质量的标准包括：样本在目标机器人任务中的多样性、所有 BT 结构的正确性以及 BT 是否符合给定任务要求。

6 LLM-BRAIn 7B 模型微调

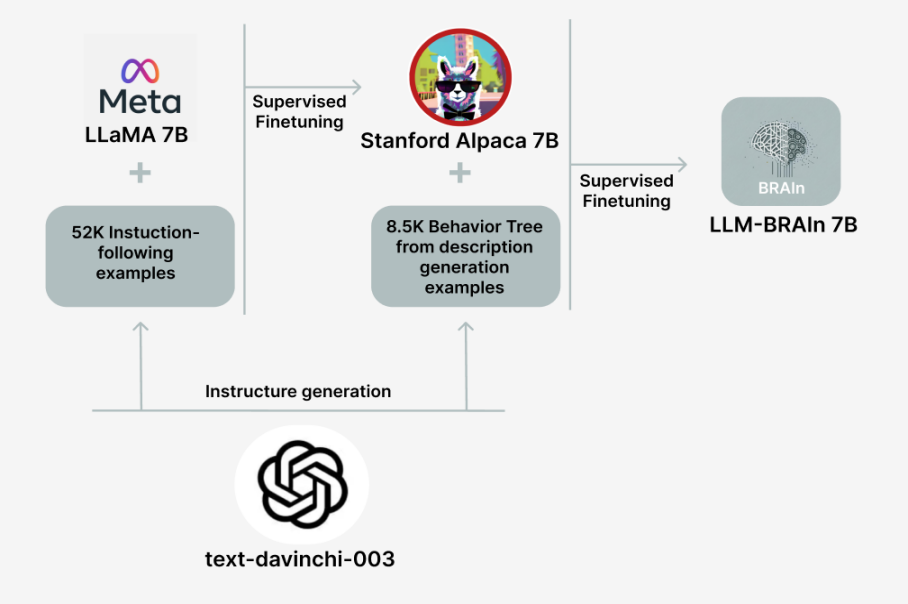

Figure 2: LLM-BRAIn 模型微调流程概述。

6.1 LLM-BRAIn 获取方法

斯坦福 Alpaca 7B 模型是基于 LLaMA 7B 模型，通过 52000 条指令跟随示例微调而得的语言模型。该微调方式使模型在指令跟随任务上表现卓越。然而，构建机器人 BT 对语言模型而言仍是极具挑战性的任务。为应对该挑战，我们对模型进行了 BT 生成样本的进一步微调。图 2展示了我们获取 LLM-BRAIn 模型的流程。

本次微调的目标有二：一是确保微调后模型能够生成满足所有指定条件的 BT；二是评估数据集规模对生成 BT 质量的影响。为实现这一目标，我们采用了参数高效微调（Parameter-Efficient Fine-Tuning, PEFT）方法。PEFT 方法支持在不需重新训练整个模型的前提下，对 LLM 进行高效微调，相关内容见 Pu et al. [21]。

PEFT 特别适用于大模型微调成本高昂的情况。其思路是调整少量附加参数，而非模型全部参数，从而显著减少计算与存储开销。本文中，我们采用的 PEFT 方法为 LoRA（Low-Rank Adaptation），该方法在 transformer 层中添加低秩矩阵，仅调整这些矩阵参数，而非整个模型，相关内容参见 Hu et al. [22]。

6.2 微调过程

图 3展示了 LLM-BRAIn 模型在不同数据集规模下的微调损失曲线。可观察到，误差曲线形状相似，从而便于判断训练何时停止以避免收益递减。我们还注意到，在训练样本中仅使用“instruction:”部分而不使用“input:”部分有助于加快训练速度。微调在一张配备 80GB 显存的 NVIDIA Tesla A100 显卡上进行，使用的 batch size 为 128，micro batch size 为 4。如将 batch size 改为 32，micro batch size 改为 1，则可在较少显存的条件下复现实验结果。所有超参数设置采用 Wang [11]仓库中的默认值。学习率设为 3e-4，验证集占总数据集的 10%。最终数据集训练共进行 3 轮（epoch），但若对更小数据集进行训练，为获得相似损失，需要增加训练轮次。

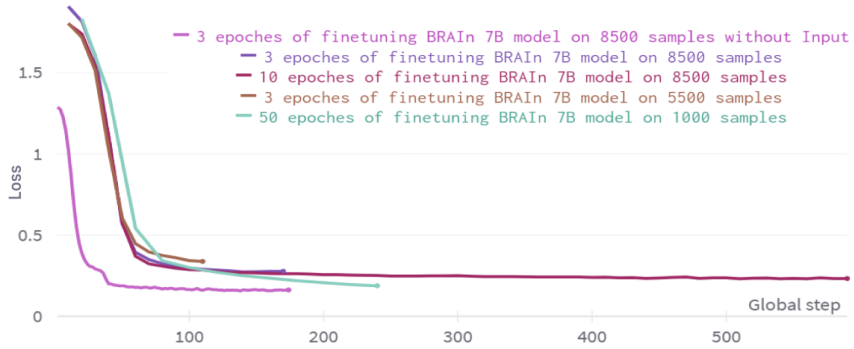

Figure 3: LLM-BRAIn 微调过程中的损失曲线。

在模型微调完成后，我们在验证集中的随机样本上进行了评估。使用 1000 条样本进行微调后，模型可在格式上正确生成各个节点，但无法构建复杂的 BT 逻辑。当数据集扩大到 5500 条并提高多样性后，模型开始生成与工业中常用 BT 相似的结构，但在逻辑层仍存在错误(如建议机器人在开门前进入房间)。此外，模型也会产生一些格式错误，例如在 BT 根节点下包含多个节点。

为解决这些问题，我们构建了一个包含 8500 条样本的数据集，聚焦于前一版本模型难以应对的任务。结果表明，逻辑与格式错误显著减少。在一张配备 80GB 显存的 NVIDIA Tesla A100 显卡上训练最终模型共耗时 8 小时。为获得验证模型质量的数值指标，我们还进行了实验评估。

7 人类识别模型生成行为树能力的实验

为评估模型性能，我们进行了一个实验，考察参与者区分 LLM-BRAIn 生成的行为树（BT）与人类专家手动编写 BT 的能力。类似的研究在 Brown et al. [3]中已有开展，但不同的是，我们的实验中所有参与者均了解行为树的基本原理与构造方法。

参与者： 我们招募了 15 名参与者，包括本科生与研究生，均专攻机器人方向，以评估 BT 生成质量。在这些参与者中，有 10 人为本地 Eurobot 团队成员，平时经常使用 BT；其余 5 人此前未接触过 BT，在实验前接受了关于 BT 原理与构建方法的明确培训。在实验开始前，所有参与者均签署了知情同意书，确认其已掌握机器人行为树的运行与结构知识。

实验流程： 实验中，使用 LLM-BRAIn 生成了 10 个行为树，模拟了移动机器人可执行的多种行为与交互。此外，还由人类专家手动编写了另外 10 个行为树，旨在实现与 LLM-BRAIn 生成 BT 相似的功能与行为。参与者被呈现机器人行为的描述及一对 BT（包含一个由 LLM-BRAIn 生成，另一个由人类编写)。为避免偏见，展示顺序被随机化。参与者需判断哪一个 BT 由 LLM-BRAIn 生成，哪一个由人类创建。他们被告知依据主观判断以及可识别的差异特征作答。图 4展示了一个示例题目。

实验结果： 在所有行为树对完成评估后，收集了实验数据并进行了分析。结果如图 5所示。平均每位参与者在 10 道题中答对 4.53 道，表明参与者区分 LLM-BRAIn 生成 BT 与人类编写 BT 的能力与随机猜测相当。

为检验问卷回答与具体题目的相关性，我们进行了单因素方差分析（ANOVA），显著性水平设为 5%。分析结果表明，不同题目下用户判断之间无显著差异（$F = 0.75$，$p = 0.66 > 0.05$），

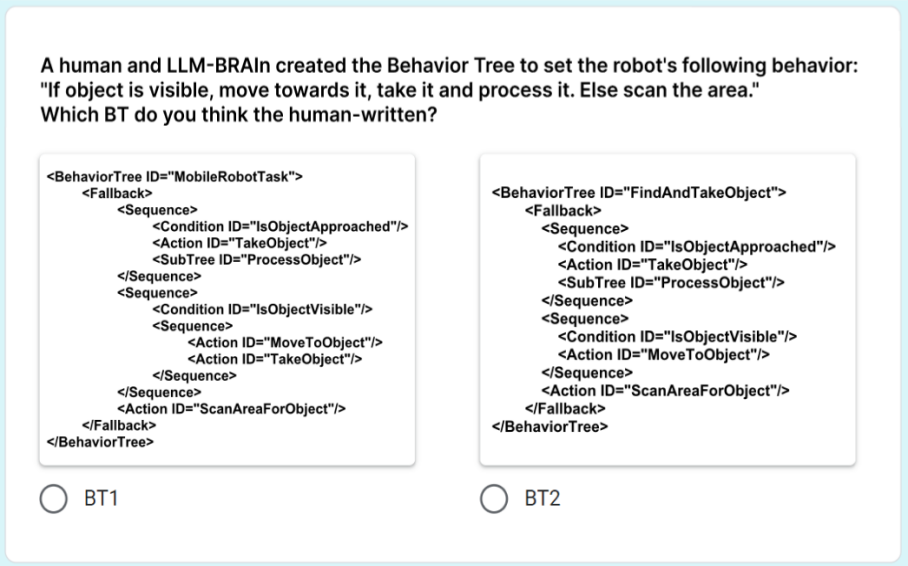

Figure 4: 任务描述及两个解决方案示例：人工编写（BT1）与 LLM 生成（BT2）的行为树。

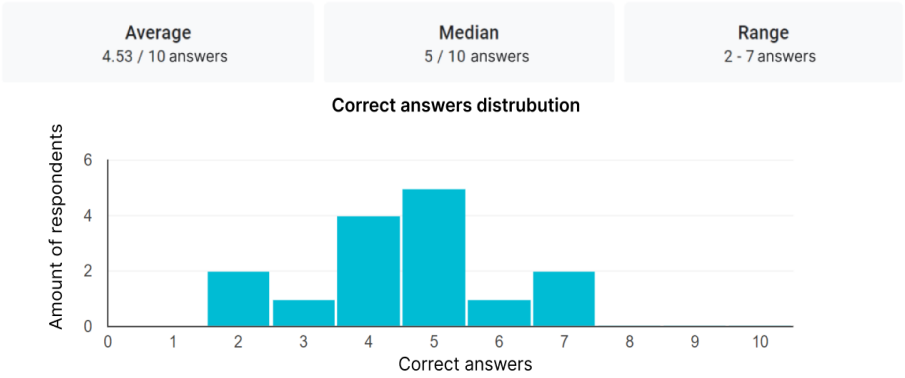

Figure 5: 受试者给出的正确答案数量。

说明正确回答的概率不受具体题目影响。为评估人类生成 BT 与 LLM-BRAIn 生成 BT 之间是否存在显著差异，我们进行了 t 检验，显著性水平仍设为 5%，以验证“无法区分二者”为零假设。若零假设成立，平均正确数应为 5，平均得分应为 0.5。根据结果（平均得分 = 0.453，t 统计量 = -1.200，临界值 = 2.145，$p > 0.05$），我们没有理由拒绝零假设，即平均得分与 0.5 之间无显著差异。

本次包含 15 名志愿者的用户研究未发现 LLM-BRAIn 生成 BT 与人类生成 BT 之间存在主观差异。LLM-BRAIn 模型展现出生成与人工相当的机器人行为能力，至少在本实验设置中，在主观感知层面上效果相当。

8 局限性

在使用 LLM-BRAIn 时，需要认识到模型运行方式带来的一些限制。首先，当前训练阶段尚未涵盖子树生成能力，因此生成的 BT 大小受限于执行模型的设备能力。运行该模型需在显存或内存中存储生成文本所有 token 之间的注意力权重，这限制了可生成 token 数量，进而限制了 BT 规模。该限制将在未来通过引入递归生成子树的机制得到缓解。

此外，尽管 LLM-BRAIn 定位为完整的机器人控制系统，但在具体设备上的应用可能需要对模型进行额外训练，使其适配该设备的 BT 样例及控制逻辑。例如，为有效控制无人机，训练集中应包含包含飞行控制逻辑的 BT。

系统的另一个限制来自节点库。该库涵盖机器人可执行的所有动作及可评估的所有条件。若要在某一机器人或系统中部署 LLM-BRAIn，需单独构建其节点库。尽管这一做法限制了机器人可执行动作范围，但也提升了行为的可预测性与安全性。如果节点库中所有节点均以安全性为首要设计原则，则机器人行为也将以安全为导向。这种方式有效降低了 LLM-BRAIn 机器人潜在的负面影响。

9 结论

本文提出了一种新颖的自主机器人控制方法，该方法能够根据操作员指令生成机器人的行为树。LLM-BRAIn 模型的突出之处在于：其不仅能够精确生成复杂机器人行为，同时具备足够紧凑的模型体积，适合部署于板载微型计算机上。用户研究结果未显示 LLM-BRAIn 生成的行为树与人类编写行为树在主观认知上存在显著差异。在所有测试中，参与者仅有 45.3% 的情况能成功区分人工与模型生成的行为树。

在未来的工作中，我们计划通过引入子树的递归生成机制来扩展节点库，从而增强模型能力。此外，我们还希望加入功能，使操作员能够根据自身需求更改机器人的行为并请求相关注释信息。进一步地，我们计划将 LLM-BRAIn 集成至移动机器人、机器人操作臂系统以及无人机集群的控制系统中。

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[2] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[4] OpenAI. Introducing chatgpt. *OpenAI Blog*, 2022. URL `https://openai.com/blog/introducing-chatgpt/`.

[5] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[6] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[7] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus

Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2022.

[9] Vilius Petkauskas. Chatgpt injected into boston dynamics' spot. *Cybernews*, 2023. URL `https://cybernews.com/tech/chatgpt-google-boston-dynamics-spot/`.

[10] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpaca: A strong, replicable instruction-following model. *CRFM Stanford University*, 2023. URL `https://crfm.stanford.edu/2023/03/13/alpaca.html`.

[11] Eric J. Wang. Alpaca-LoRA, 2023. URL `https://github.com/tloen/alpaca-lora`.

[12] Yue Cao and C. S. George Lee. Robot behavior-tree-based task generation with large language models, 2023.

[13] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022.

[14] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions, 2022.

[15] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66), may 2022. doi: 10.1126/scirobotics.abm6074. URL `https://doi.org/10.1126%2Fscirobotics.abm6074`.

[16] Davide Faconti and Michele Colledanchise. BehaviorTree.CPP, 2019. URL `https://github.com/BehaviorTree/BehaviorTree.CPP`.

[17] Michele Colledanchise and Petter Ögren. *Behavior Trees in Robotics and AI*. CRC Press, jul 2018. doi: 10.1201/9780429489105. URL `https://doi.org/10.1201%2F9780429489105`.

[18] Michele Colledanchise and Petter Ögren. How behavior trees modularize robustness and safety in hybrid systems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1482–1488, 2014. doi: 10.1109/IROS.2014.6942752.

[19] Evgenii Safronov, Michele Colledanchise, and Lorenzo Natale. Task planning with belief behavior trees, 2020.

[20] Grzegorz Granosik, Kacper Andrzejczak, Mateusz Kujawinski, Rafal Bonecki, Łukasz Chlebowicz, Blazej Krysztofiak, Konrad Mirecki, and Marek Gawryszewski. Using robot operating system for autonomous control of robots in eurobot, erc and robotour competitions. *Acta Polytechnica CTU Proceedings*, 6:11, 11 2016. doi: 10.14311/APP.2016.6.0011.

[21] George Pu, Anirudh Jain, Jihan Yin, and Russell Kaplan. Empirical analysis of the strengths and weaknesses of peft techniques for llms, 2023.

[22] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.