BTGenBot-2：基于小型语言模型的高效行为树生成

Riccardo Andrea Izzo、Gianluca Bardaro 和 Matteo Matteucci，
电子、信息与生物工程系，米兰理工大学，米兰（riccardo.izzo、
gianluca.bardaro、matteo.matteucci）@polimi.it

arXiv:2602.01870v1 [cs.RO] 2 Feb 2026

摘要—**机器人学习的最新进展日益依赖基于大语言模型的任务规划，利用其连接自然语言与可执行动作的能力。尽管先前的研究展现出优异性能，但这些模型在机器人学中的广泛应用仍面临挑战：1）现有方法往往是闭源的或计算开销巨大，忽视了在真实物理系统中的实际部署；2）目前尚无公认的、即插即用的机器人任务生成表示方法。为应对这些挑战，我们提出 BTGenBot-2，这是一种10亿参数的开源小型语言模型，可将自然语言任务描述和机器人动作原语列表直接转换为 XML 格式的可执行行为树。与以往方法不同，BTGenBot-2 支持零样本行为树生成，并能在推理和运行时进行错误恢复，同时足够轻量，可用于资源受限机器人。我们还推出首个面向基于大语言模型的行为树生成的标准化基准测试，涵盖 NVIDIA Isaac Sim 中的 52 项导航和操作任务。广泛的评估结果表明，在功能性和非功能性指标方面，BTGenBot-2 始终优于 GPT-5、Claude Opus 4.1 及规模更大的开源模型；其零样本和单样本平均成功率分别达到 90.38% 和 98.07%，推理速度最高可比此前的 BTGenBot 快达 16×倍。**

索引词—**机器人学习、大语言模型、行为树**

I. 引言

随着 GPT-4 [1], 、Gemini [2], 和 Llama [3]等基础语言模型的出现，能够自主生成可由机器人直接执行的任务序列这一过程引起了机器人学界的关注。这些模型展现出令人瞩目的推理能力 [4], ，能够表征常识知识 [5],，将语言与物理对象关联起来 [6], ，并根据上下文信息调整计划 [7]。因此，人们利用大语言模型（LLM）通过多种表示形式生成机器人任务，例如规划领域定义语言（PDDL） [8], 、线性时序逻辑（LTL） [9], 、Python代码 [10], [11] 和行为树（BT） [12], [13]。其中，行为树凭借其可扩展的结构化方法，在管理复杂机器人任务方面日益普及；BehaviorTree.CPP 库也已成为 Robot Operating System 2（ROS2）生态系统中的标准组件，这主要得益于其被纳入 Navigation2导航栈 [14]。尽管取得了这些进展，先前的大多数工作仍侧重于将 OpenAI 的 GPT 等闭源模型 [1]，集成到其流程中，却忽视了在资源受限机器人上的实际部署，并依赖无法在离线场景中访问的应用程序接口。尽管一些研究取得了令人瞩目的成果，但仍有两项主要局限阻碍其实际应用：1）当前模型是闭源的，或在模型架构、训练流程和数据混合方面透明度有限；2）依赖计算密集型模型，无法在实际机器人硬件上切实运行。我们认为，推动机器人研究进步需要基于开源大语言模型 [3]–[15]，开发开放且高效的模型。这样一来，发布的模型便可在真实机器人上本地部署，无需依赖外部API或额外组件。此外，该领域仍缺少用于评估机器人学中大语言模型驱动的行为树生成的标准化基准测试。为此，我们推出 BTGenBot-2：一款拥有 10 亿参数的开源小型语言模型，在生成可执行且兼容 ROS2 的行为树方面达到了新的最先进水平。如图 1 所示，BTGenBot-2 基于预训练小语言模型构建，并在定制的合成指令跟随数据集上采用计算高效方法 [16], [17] 进一步微调。与此前的方法不同，BTGenBot-2 支持零样本行为树生成，在推理和运行时均能抵御错误，并且足够轻量，可在消费级 GPU 上运行，同时保持最先进的性能。我们还推出了一个全面的基准测试，涵盖导航和操作领域的 52 项任务，按难度分为三个级别，并在 NVIDIA Isaac Sim 中实现[1]。实验结果表明，BTGenBot-2 的表现优于 BTGenBot [18]，（一款用于行为树生成、拥有 70 亿参数的最先进模型），也优于 OpenAI 的 GPT-5 和 Anthropic 的 Claude Opus 4.1 等专有模型，在功能性和非功能性指标上均是如此。我们的主要贡献如下：

1) 我们发布了一个新的合成指令跟随数据集，包含5,204对可执行行为树与自然语言任务描述，通过为可泛化的行为树生成奠定基础，填补了规划数据集中的空白。2) 我们推出BTGenBot-2，这是一个完全开源的小型语言模型，用于在ROS2技术栈内进行端侧行为树生成。3) 我们设计了两种新颖的故障检测机制，以处理推理与运行时的错误，显著增强了鲁棒性。4) 我们提出了首个用于评估基于大语言模型的行为树生成的标准化基准，从而实现可复现

[1]https://developer.nvidia.com/isaac/sim

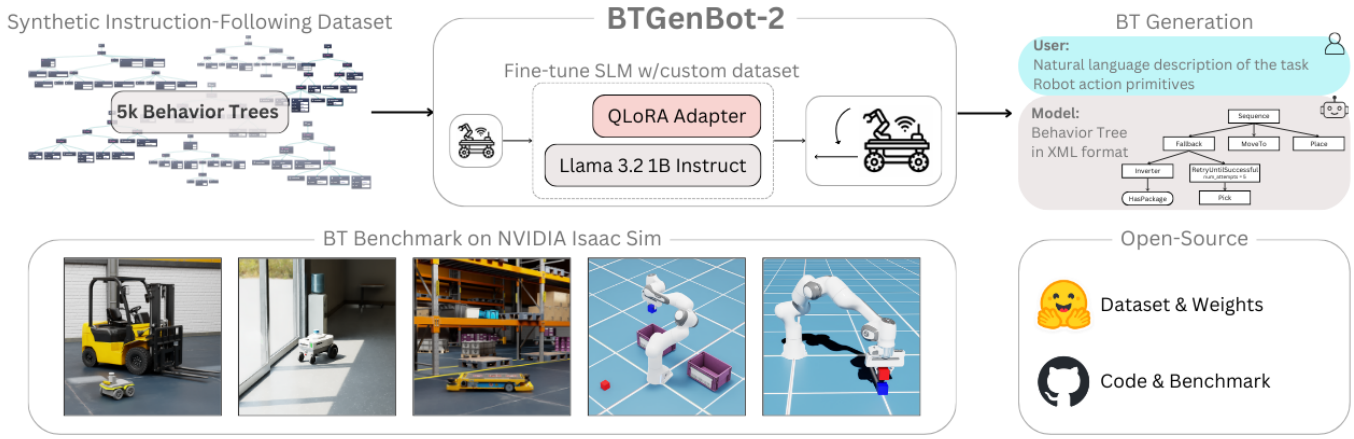


图 1. 我们介绍了**BTGenBot-2**，这是一款10亿参数开源小型语言模型（SLM），使用来自合成指令跟随数据集的 5,000 组自然语言指令与行为树配对数据进行训练。BTGenBot-2 在生成可直接执行的行为树方面取得了最先进的性能。我们将合成数据集、模型权重、代码库和行为树基准测试开源，以支持推广和结果复现。

以及跨功能性与非功能性指标的可比评估。

BTGenBot-2借助该基准在仿真和真实机器人上得到了广泛验证，确立了其作为强大行为树生成器的地位。

我们公开发布模型、数据集、基准测试和部署代码[2]。

II. 相关工作

A. 行为树（*BT*）

行为树在形式上定义为一种有向根树，其中内部节点称为控制流节点，叶节点称为执行节点 [12]。根节点没有父节点，而其他节点之间则构成父子关系。在执行期间，周期性 tick 信号从根节点发出，并从左向右沿树传播。节点只有在收到 tick 信号时才会执行，并返回以下状态信号之一：成功、失败或运行中。经典的行为树表示法包含四类控制流节点（顺序节点、回退、并行节点和装饰节点）以及两类执行节点（动作节点和条件节点）。以往将行为树与有限状态机（FSM）、PDDL 和决策树等其他表示形式进行比较的研究，凸显了行为树在模块化、反应性和可解释性方面的优势 [19]。这些优势，加上 BehaviorTree.CPP[3] 和 pytrees[4]，等库的可用性，进一步推动了行为树的应用。

B. 机器人学中的大语言模型

大语言模型（LLM）是一类通常基于 Transformer 架构构建的深度学习模型。尽管它们最初是为文本生成而设计的 [20]，，大语言模型后来在指令遵循 [21]，、决策 [22]，、推理 [4] 和上下文学习 [20]方面展现出卓越的能力。如今，在最广为人知的大语言模型中，有 GPT [1]，、Gemini [2] 和 Llama [3]。它们迅速普及，也扩展到了机器人学领域；在该领域，大语言模型已被用于基于语言的任务生成 [10]，[18]，[23]，、结合视觉表示的端到端动作生成 [24]，[25]，以及人机交互 [26]。

C. 使用大语言模型生成行为树

近年来，机器人学领域的工作日益借助大语言模型或基础模型来实现高层任务规划。早期方法将用户命令转换为 Python 脚本，把基于语言的指令映射为代码原语 [10]，[11]，[23]，，或转换为 PDDL [27] 和 LTL [9] 等经典规划语言。一种新兴方法是根据任务的自然语言描述直接生成行为树 [18]，[28]–[32]。LLM-BrAIn [28] 使用完全合成的数据，且仅依赖人工评估，忽视了实际机器人部署。BTGenBot [18]利用经过真实世界测试的行为树对轻量级模型进行微调，提供全面的评估框架，并输出兼容 BehaviorTree.CPP 库的 XML 格式行为树。LLM-as-BT-Planner [29]提出了四种上下文学习策略，包括人在回路的方法，用于根据用户指令构建行为树。OBTEA [30]将目标编码为格式良好的一阶逻辑公式，再通过大语言模型处理这些公式，随后将规划扩展为行为树。类似地，LLM-BT [31] 融入语义地图，展现出对环境变化的鲁棒性和适应能力；BETR-XP-LLM [32] 则采用动态扩展来解决规划和执行阶段中的错误。先前的工作 [29]–[32]使用闭源 GPT 模型或依赖外部API，因此不适合部署在真实机器人上。我们的方法直接采用可部署于机器人上的开源小语言模型，并可在消费级硬件上运行。此外，尽管 [29]，[30] 高度依赖上下文学习和少样本示例等提示技术，我们的模型无需用户提供任何示例即可进行零样本生成

[2]https://airlab-polimi.github.io/BTGenBot-2/ [3]https://www.behaviortree.dev/ [4]https://py-trees.readthedocs.io/en/devel/

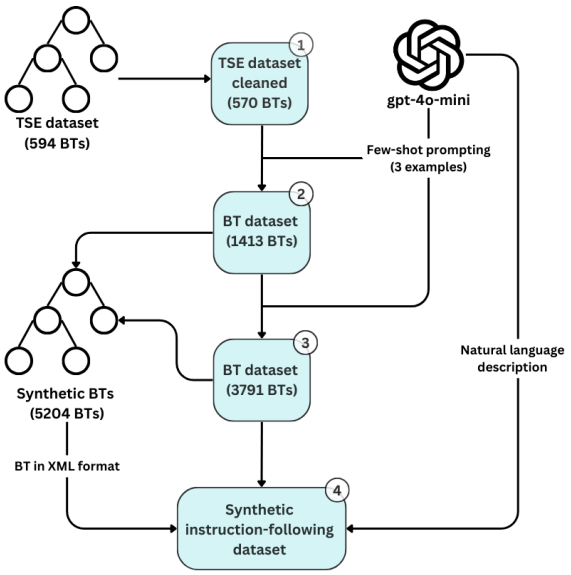


图 2. **数据集生成**。从 TSE 数据集出发，通过四个关键步骤创建新的指令跟随数据集：(1) 清理原始 XML 数据；(2) 针对每个原始行为树，使用 *gpt-4o-mini* 生成三个变体；(3) 使用新数据集重复步骤 2；(4) 合并所有生成的数据集，并为每个行为树生成自然语言描述。

Instruction:
You are a helpful assistant that can assist with creating behavior trees. Your task:
- Convert the provided summary of a behavior into an XML-formatted behavior tree.
- Ensure the behavior tree matches the description in the summary.
- The behavior tree must be compatible with the BehaviorTree.CPP library.
- Only use the actions and parameters provided in the action list below the summary.

Output Requirements:
- Output only the XML representation of the behavior tree. Do not include explanations, comments, or any additional text.
- Ensure all actions and parameters strictly match the provided list.
- If possible limit the use of SubTrees.

Please generate the behavior tree based on the summary and action list provided.

Input:
The behavior tree facilitates a robot's navigation by implementing a fallback mechanism to achieve its tasks. It first attempts to compute and follow a path using the A* algorithm to reach a designated goal (goal1). If this sequence is unsuccessful, it tries to compute a path to a second goal (goal2) using the Dijkstra algorithm before following it. If both attempts to reach the goals fail, the robot will move to a predefined location called "Base." This structure ensures that the robot has multiple options to achieve successful navigation, providing robustness and flexibility in its operational environment.

Actions: [ComputePathToPose (parameters: goal, path, planner_id), FollowPath (parameters: path, controller_id), MoveTo (parameters: name, location)]

Output:

```
<root BTCPP_format="4">
 <BehaviorTree ID="MainTree">
  <Fallback>
   <Sequence>
    <ComputePathToPose goal="{goal1}" path="{path1}" planner_id="AStar"/>
    <FollowPath path="{path1}" controller_id="FollowPath1"/>
   </Sequence>
   <Sequence>
    <ComputePathToPose goal="{goal2}" path="{path2}" planner_id="Dijkstra"/>
    <FollowPath path="{path2}" controller_id="FollowPath2"/>
   </Sequence>
   <MoveTo name="return_to_base" location="Base"/>
  </Fallback>
 </BehaviorTree>
</root>
```

图 3. **数据集样例**。生成的指令跟随数据集中的一个代表性示例，包含三个组成部分：指令提供系统上下文信息；输入由自然语言任务描述及其对应的机器人动作组成；输出展示生成的基于 XML 的行为树。

额外示例。此外， [30], [32]侧重于扩展现有行为树，而我们的方法可直接生成可执行的行为树。最后，除 BTGenBot [18]和我们的框架外，现有的行为树生成方法均为闭源，且与 ROS2 不兼容。因此，我们将 BTGenBot 作为主要基线。

III. 方法

本节详细介绍我们使用小型语言模型生成可执行行为树的方法，如图 4 所示。在 BTGenBot 的基础上，我们综述了 Hugging Face 的 *Open LLM* 排行榜上表现最佳的开源小语言模型，并选择了 *Llama-3.2-1B-Instruct* [33], ，这是一种上下文长度为 8k 的指令微调模型。第 III-A 节对问题进行形式化。第 III-B 节详细介绍我们的数据整理流程，第 III-C 节则描述 QLoRA 微调过程。最后，第 III-D 节概述推理和运行时错误恢复机制，第 III-E 节介绍用于评估基于 LLM 的行为树生成的标准化框架。

A. 问题表述

我们将行为树的生成表述为一个监督学习任务。设 $\mathcal{D} = \{(\mathcal{I}_i, \mathcal{A}_i, \mathcal{T}_i)\}_{i=1}^{N}$ 为一个数据集，其中 $\mathcal{I}$ 表示一条自然语言指令， $\mathcal{A}$ 是机器人动作原语集合， $\mathcal{T}$ 是对应的 XML 格式行为树。我们的目标是微调一个由 θ 参数化的小型语言模型，以最大化在给定输入条件下生成正确行为树 $\mathcal{T}$ 的概率：

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{(\mathcal{I},\mathcal{A},\mathcal{T})\in\mathcal{D}} \log P_\theta(\mathcal{T} \mid \mathcal{I}, \mathcal{A}) \qquad (1)$$

生成的输出被隐式约束为与 BehaviorTree.CPP 兼容的有效 XML 架构，且仅包含动作空间 $\mathcal{A}$ 中存在的叶节点。

B. 数据集

我们按照 Alpaca 架构创建了一个新的指令跟随数据集 [34], ，每个样本包含三个组成部分：指令、输入和输出。指令部分为模型提供系统上下文信息，包括其角色、所需的输出格式以及任何其他约束。输入部分是对机器人任务的自然语言描述，并附有可用机器人动作列表。最后，输出部分是一个基于 XML 的行为树，与所提供的任务描述相匹配。我们的数据集以 TSE 数据集 [13], 为基础，其中包含约 600 棵源自开源机器人学项目的行为树。这些行为树已在实体机器人上得到验证，确保了真实场景中实现方案的高质量与可靠性。如图 2 所示，我们的数据集生成过程分为四个步骤：

（1）数据准备。我们从 TSE 数据集 [13] 中选取了 594 棵行为树。首先，我们使用 Python XML 解析器验证了它们的语法正确性及其与 *BehaviorTree.CPP* 的兼容性。完成这些验证步骤后，剩下 570 棵行为树。

(2) 首次行为树生成。基于这一精选数据集，我们使用 *gpt-4o-mini* 为每棵行为树生成三个新变体，遵循 [34], 的方法，该方法从精选人工编写的指令-响应对种子集中生成指令-响应对。我们选择*gpt-4o-mini* 是因为它推理速度快且成本低。随后，受 [35], 启发，我们采用核采样（top-p = 0.99），在使用相同少样本示例的同时激发创造力，

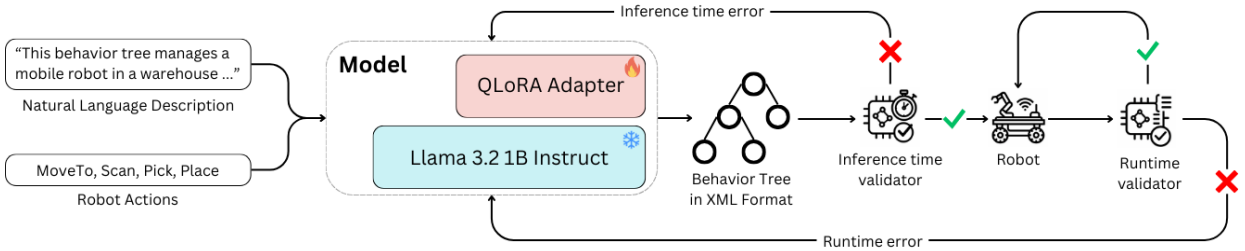


图4. **模型架构概览**。 模型以机器人任务的自然语言任务描述和可用机器人动作原语集作为输入，生成兼容 ROS2 的 XML 格式行为树。模型通过 QLoRA 适配器进行微调，同时冻结预训练参数。生成的行为树会在推理时首先进行验证，在执行前检查语法和动作空间一致性。此外，在运行时，内联日志记录器会跟踪堆栈跟踪和黑板状态，并在发生错误时触发子树重新生成。

但我们对生成过程施加了约束，确保执行节点保持不变。这确保了生成结果与底层机器人API及行为树*.CPP*兼容。经过最后的清理阶段，该流程共生成了 1413 棵合成行为树。

(3) 第二轮行为树生成。 我们使用新生成的 1413 棵行为树作为输入集，重复步骤 (2) 中的过程，额外获得了 3791 棵合成行为树。这一步进一步提升了数据集的整体多样性。

(4) 指令生成。 我们合并了步骤 (2-3) 中生成的两个合成行为树数据集，得到总计 5204 棵行为树。对于其中每棵行为树，*gpt-4o-mini* 随后生成：(i) 一段简短的任务自然语言描述（少于 200 个词），以及 (ii) 与之对应的动作节点和参数列表。此步骤通过为每棵行为树配上任务描述和所需的机器人动作，完成了数据集的构建。

为了评估*gpt-4o-mini*生成结果的质量，我们对部分结果进行了视觉评估。具体而言，我们检查了自然语言描述与对应行为树之间的语义连贯性，并使用 XML解析器验证行为树的语法正确性。图3展示了生成数据集中的一个典型示例。尽管生成更多样本或许能进一步丰富数据，但我们的初步实验表明，边际收益会逐渐降低，行为树质量也会下降。生成的数据集通过保留原始训练分布中的任务类型，反映了真实世界的规模。行为树平均包含 11.23 个节点和 10.23 个状态转换，最多则可扩展至 157 个节点和 156 个状态转换，涵盖常见行为树以及复杂工业场景中使用的大型行为树。与现有行为树集合相比，我们的数据集具有显著优势。例如， [18]提供了高质量的真实世界数据集，但其训练令牌数量远低于微调一个70亿参数模型所需的数量。相反， [28] 引入的数据集完全由合成数据构成。尽管其约 8500 个样本的规模更接近推荐的词元数量，但它不具备源自 TSE 数据集的行为树所具有的同等真实世界验证水平。我们的数据集融合了两种方法的优势：以精心整理的真实世界行为树为核心，并通过受控合成生成扩充其规模和多样性。

C. 微调

微调大语言模型是一项计算密集型任务，因此我们采用了参数高效微调（PEFT）方法，具体而言是 QLoRA [17], , 这是 LoRA [16]的一种量化扩展。QLoRA 保留了 LoRA 的优势，例如无需重新训练整个模型，通常只需训练原始模型约千分之一的参数。例如，对于 *Llama-3.2-1B-Instruct*这类10亿参数模型，LoRA 可能只需训练约一百万个参数。每参数 15 个词元的启发式估算表明，10亿参数模型约需 1500 万个词元 [36],, 而我们的数据集约含 350 万个词元，比目标少了一个数量级。尽管未达到先前启发式估算的要求，我们仍决定继续，因为新近证据表明，即使规模相对较小但质量较高的数据集，也能有效微调大语言模型并取得有竞争力的性能 [37]。训练时，我们使用了第 III-B 节所述的数据集，并按常规的 95%-5% 比例划分为训练集和测试集。大多数 QLoRA 超参数均采用默认值（例如*lorar*、*lora alpha*和*lora dropout*）。按照 [17], , 我们将原有的 LoRA 目标模块从注意力层 [中的 'k proj'、'q proj'、'v proj'、'o proj']扩展至多层感知机层 [中的 "gate proj"、"down proj"、"up proj"]。大语言模型通常最多训练一到两个轮次，但我们发现，对于规模相对较小的数据集，训练最多五个轮次效果更好，训练准确率超过 95%。学习率方面，我们采用了标准固定值 1e-4。我们使用比例为 0.1 的学习率预热，以提升训练稳定性并减少早期过拟合。最终模型在两块 NVIDIA RTX Quadro 6000 显卡上训练约 30 小时，总显存为 48GB，批次大小为 16。

D. 错误处理与恢复

选择行为树作为我们的表示方法，一个关键原因是它本身能够处理失败并支持恢复策略。在 BTGenBot-2 中，我们明确处理自动生成行为树时必须考虑的两类错误：从自然语言生成行为树时出现的推理时间错误，以及机器人执行过程中出现的运行时错误。

1）推理时验证器：大语言模型生成的输出常常存在语法错误，或在语义上与机器人的动作空间不一致。为确保生成结果可部署，BTGenBot-2 添加了一个验证层，将解码限制为 XML 格式，并强制使用封闭词汇表中的行为树节点和动作原语。任何格式错误的输出都会被自动拒绝，并触发重新生成。该验证层利用专为行为树设计的 Python 解析器，并通过 YAML 文件描述所有允许使用的原语，从而避免生成行为树时出现幻觉。通过这一分层验证，可确保 BTGenBot-2 生成的每棵行为树在执行前都符合语法规范，且与目标机器人系统的机器人原语一致，无需人工干预。

2）运行时验证器： 与扁平动作序列不同，行为树支持通过回退和重试等控制流节点实现响应式执行。此外，BTGenBot-2 专注于局部恢复机制，这些机制可以直接编码到生成的行为树中。具体而言，每个树节点都通过 C ++ 进程内日志记录器实现该机制。当任何叶节点或条件节点返回“失败”时，该节点会记录堆栈跟踪和黑板快照。如果局部恢复（例如回退、重试）成功，则不会触发重新生成。否则，如果预期失败会向上传播，运行时解析器便会请求 LLM 重新生成子树。通过推理时验证器验证后，原树会用新子树进行修改。这一机制进一步增强了响应性和鲁棒性，使系统甚至能够在执行过程中纠正意外错误。

E. 行为树基准测试

在机器人学中评估行为树生成本身就颇具挑战，因为既要评估结构有效性，也要评估执行情况。然而，现有方法往往依赖含义模糊的表示方式，这些表示不受 ROS2 BehaviorTree.CPP 和 py trees 等广泛采用的框架支持；同时，它们采用的任务套件定义范围狭窄，限制了可复现比较的可能性。此外，要有效测试组合式行为树，需要复杂且经过刻意多样化设计的机器人配置，以确保全面涵盖各种行为树变体和任务领域。如今，大语言模型已能够根据自然语言合成行为树，因此，缺少通用且可复现的基准已成为关键瓶颈。我们提出了一种用于评估行为树生成的标准化基准，并采用*BehaviorTree.CPP* 作为规范语法，因为它与 ROS2 兼容。该基准涵盖导航和操作领域的 52 项任务，并分为三个难度等级。导航任务共 32 项，包括 12 项简单任务、10 项中等难度任务和 10 项困难任务。操作任务共 20 项，包括 6 项简单任务、8 项中等难度任务和 6 项困难任务。总体而言，该基准包含 18 项简单任务、18 项中等难度任务和 16 项困难任务。导航任务包括简单难度下的点到点和航点序列问题、中等难度下的优先级导航和条件导航，以及困难难度下这些任务的混合形式，并特别强调新颖动作原语的使用、故障恢复和更复杂的推理。操作任务聚焦于

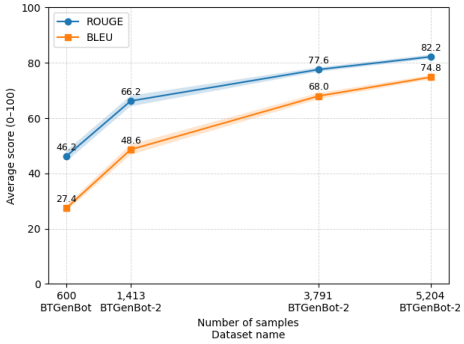


图 5. **数据扩展评估**。 随着数据集规模增大，ROUGE 和 BLEU 分数的平均值（均值 ± 标准差）：ROUGE：.....；BLEU：.....，分别对应 600、1,413、3,791 和 5,204 个样本。标准差基于温度=0.9 下的 5 次运行进行评估

桌面场景，包括抓取与放置、堆叠、排序和物体重新排序。为检验泛化能力，困难任务相对于我们数据集中的原始行为树被刻意设计为分布外任务，从而大幅提高了测试难度，更严格地检验模型能否胜任具备广泛泛化能力的行为树生成器。所有任务均在*NVIDIA Isaac Sim*中实现；该平台是广泛用于机器人学习的仿真环境。

IV. 实验与结果

我们实验评估的目标是评估我们的模型作为一种强大工具的性能与有效性，该工具通过行为树和自然语言指令生成机器人任务。具体而言，我们将我们的模型与先前采用大语言模型进行行为树生成的工作，以及当前最先进的通用大语言模型进行基准测试。在我们的评估中，BTGenBot [18] 作为基于大语言模型的行为树生成的主要基线，而 GPT-5 和 Claude Opus 4.1 则代表领先的通用大语言模型。

A. 数据整理流程有效性

在本节中，我们评估数据集规模对行为树生成质量的影响，从而验证第 III-B 节中介绍的数据整理流程。我们按照第 III-C 节所述的流程，对*Llama-3.2-1B-Instruct*进行微调，采用四种数据集配置，并根据数据集规模调整训练轮数，同时固定其他超参数。为了评估模型向目标数据分布收敛的情况，我们采用了 ROUGE [38]和 BLEU [39]等基于文本的指标。ROUGE 分数通过比较生成文本与参考文本中重叠的 n 元语法和词序列，衡量两者的相似度。BLEU 分数则衡量文本相对于原文的整体质量。所有模型均在每种数据集配置中占 5% 的测试集划分上进行评估。如图 5 所示，随着训练样本增加，两项指标均单调上升。从原有的 BTGenBot 语料库扩展到包含 1,413 棵行为树的整理后子集时，提升幅度最大。进一步扩大规模虽带来较小但持续的改进，最终趋于收敛。这些结果表明

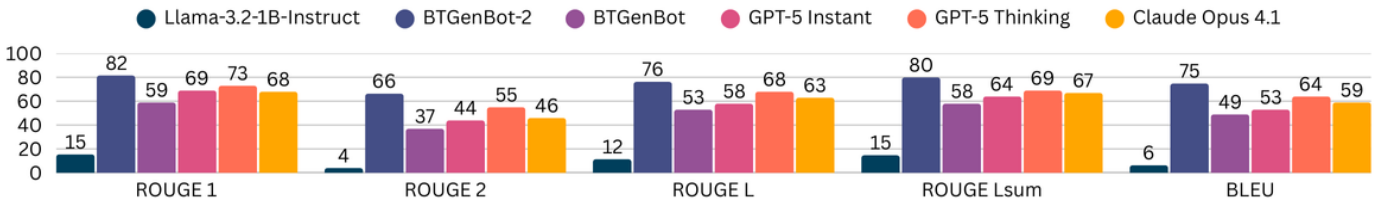


图 6. **ROUGE 和 BLEU 分数**。 在定制指令遵循数据集中的 250 个行为树测试集上进行评估，结果表明，与预训练基线相比，微调显著提升了性能，其中 BTGenBot-2 的得分最高。GPT-5 Thinking 表现出色，GPT-5 Instant 和 Claude Opus 4.1 略逊一筹。原版 BTGenBot 的得分低于其后续版本 BTGenBot-2。

我们的数据整理流程通过构建经过整理的行为树语料库，切实提升了行为树生成效果。

B. 初步评估阶段

在初步评估阶段，我们在涵盖导航和操作任务的 250 个行为树测试集上，对六个模型的行为树生成质量进行了基准测试。我们将 *Llama-3.2-1B-Instruct* 模型作为下界参考，而其微调版本 *BTGenBot-2*则代表所提方法。作为基线，我们考察了 *BTGenBot*，以及领先的专有大语言模型，例如 OpenAI*GPT-5* （Instant 和 Thinking 模式）和 Anthropic*Claude Opus 4.1*。首先使用 ROUGE 1/2/L/Lsum 和 BLEU 评估模型质量。尽管这些分数无法严格评估行为树的语义有效性，但它们是初步验证模型能否掌握正确 XML 语法和动作使用的重要依据。ROUGE 1/2 衡量一元词和二元词重叠（词汇召回率和词序敏感性），而 ROUGE L/Lsum 则通过最长公共子序列评估序列级结构相似性。如图 6 所示，*BTGenBot-2* 在所有指标上均取得最高分，甚至优于最强的专有基线（*GPT-5Thinking*）。相比之下，该模型未经微调的预训练版本表现最差，凸显了领域特定微调的重要性。*BTGenBot* 和*GPT-5 Instant*表现居中，其中后者的表现略优于原版*BTGenBot*。

C. 验证阶段

在此验证阶段，我们评估与第 IV-B 节相同的模型，但不包括基础版本*Llama-3.2-1B-Instruct* ，因为它在第 IV-B 节中的表现始终较差。我们的模型 BTGenBot-2 专为零样本运行而设计，旨在打造直观且便于最终用户使用的即插即用系统。这样无需提示工程或加入示例，同时仍能生成高质量的行为树。不过，为了公平比较，我们还在单样本设置下评估了该模型，以便直接与未在我们的数据集上微调的专有模型进行比较。对于 BTGenBot-2，我们还考虑了一个额外版本，即 BTGenBot-2-ER（错误恢复），该版本采用了第 III-D 节所述的两个验证器。为评估错误恢复模块的泛化能力，我们还将其应用于 Claude Opus 4.1，开展了消融研究。本阶段旨在评估端到端流程作为通用型行为树生成器的表现。评估采用第 III-E 节提出的基准测试，并分别报告各层级的结果。评估时，我们结合使用功能性指标和非功能性指标 [40]。功能性指标用于评估行为树是否能够执行预期功能：

- **成功率（SR）**：若行为树可执行且能达到目标状态，则视为成功。
- **Pass@k（P@k）**：衡量前 k 个生成候选中至少有一个正确的可能性 [41], ，正确性依据成功率（SR）的定义判定。
- **推理时间**：4 位量化模型生成完整行为树所需的时间，在 NVIDIA GTX 1080 GPU（8 GB 显存）上测量。

此外，非功能性指标用于衡量行为树中与其功能无直接关系的方面，且不受其性能影响：

- **动作一致性**：验证行为树是否仅使用任务提示中预先定义的机器人动作，并使用第 III-D1 节中的 Python 解析器进行计算。
- **XML 语法**：使用第 III-D1 节中的系统，评估 XML 架构的正确性及其与 BehaviorTree.CPP 库的兼容性。
- **语义正确性**：由三位人类专家采用二元量表进行评估，以衡量行为树的结构和逻辑是否符合自然语言任务描述。最终决定由多数票作出。

表 I 使用功能性和非功能性指标，在零样本和单样本设置下对模型在简单、中等和困难任务上的表现进行了比较。所有模型均采用统一的提示方式，只接收任务描述和可用机器人原语列表。在零样本设置下，BTGenBot-2 的平均成功率达到 84.61%，比表现最强的基线模型 GPT-5 Thinking 高近 14%，也大幅超过 GPT-5 Instant 和 Claude Opus 4.1；与先前的 BTGenBot 相比，成功率更是高出 30% 以上。不出所料，Pass@3 提高了绝对准确率，但没有改变模型的相对排名。在效率方面，BTGenBot-2 平均只需 11 秒即可生成一棵行为树，速度仅与 GPT-5 Instant 相当；与其他专有推理模型相比，最多快 3× ，与原版 BTGenBot 相比则快 16× 。在非功能性指标方面，BTGenBot-2 展现出很强的可靠性：它是唯一始终只使用所提供机器人动作的模型，而原版 BTGenBot 则由于原始微调数据集未在输入提示中包含动作列表，始终无法成功。

表 I 模型在功能性指标和非功能性指标上的评估

模型	功能性									非功能性		
	成功率 (E)	P@3 (E)	时间 (E)	成功率 (M)	P@3 (M)	时间 (M)	成功率 (H)	P@3 (H)	时间（小时）	平均成功率	动作一致性	XML 语法	语义正确性
零样本													
BTGenBot-2-ER	**18 / 18**	**18 / 18**	5.8s	**17 / 18**	**17 / 18**	8.2s	**12 / 16**	**13 / 16**	11.7秒	**90.38%**	**100%**	**100%**	**94.23%**
BTGenBot-2	17 / 18	**18 / 18**	**5.1s**	15 / 18	16 / 18	**7.5s**	**12 / 16**	12 / 16	**11.0秒**	84.61%	**100%**	96.15%	**94.23%**
BTGenBot	13 / 18	15 / 18	85s	10 / 18	12 / 18	98s	7 / 16	8 / 16	178s	57.69%	63.46%	78.84%	46.15%
GPT-5 Instant	15 / 18	16 / 18	12.8秒	14 / 18	15 / 18	15.0秒	5 / 16	7 / 16	15.2秒	65.38%	88.46%	82.69%	80.76%
GPT-5 Thinking	17 / 18	**18 / 18**	34.2s	15 / 18	15 / 18	36.8s	5 / 16	6 / 16	39.7秒	71.15%	92.30%	86.53%	86.53%
Claude Opus 4.1-ER	**18 / 18**	**18 / 18**	23.9秒	15 / 18	15 / 18	28.1秒	7 / 16	9 / 16	30.4秒	76.92%	92.30%	90.38%	88.46%
Claude Opus 4.1	17 / 18	**18 / 18**	23.1秒	13 / 18	14 / 18	27.4s	4 / 16	6 / 16	29.8s	65.38%	90.38%	82.69%	80.76%
单样本													
BTGenBot-2-ER	**18 / 18**	**18 / 18**	**8.4s**	**18 / 18**	**18 / 18**	9.6s	**15 / 16**	**15 / 16**	13.1秒	**98.07%**	**100%**	**100%**	**98.07%**
BTGenBot-2	**18 / 18**	**18 / 18**	**8.4s**	16 / 18	17 / 18	**8.9s**	14 / 16	**15 / 16**	**12.6秒**	92.38%	**100%**	98.07%	94.23%
BTGenBot	16 / 18	17 / 18	101s	14 / 18	14 / 18	114s	7 / 16	8 / 16	197s	71.15%	73.07%	84.61%	51.92%
GPT-5 Instant	**18 / 18**	**18 / 18**	15.3秒	16 / 18	16 / 18	16.2秒	7 / 16	8 / 16	18.8秒	78.84%	96.15%	86.53%	84.61%
GPT-5 Thinking	**18 / 18**	**18 / 18**	35.8秒	17 / 18	17 / 18	36.8秒	9 / 16	10 / 16	42.2秒	84.61%	**100%**	92.30%	88.46%
Claude Opus 4.1-ER	**18 / 18**	**18 / 18**	25.5秒	**18 / 18**	**18 / 18**	29.8秒	10 / 16	12 / 16	34.3秒	88.46%	**100%**	96.15%	94.23%
Claude Opus 4.1	**18 / 18**	**18 / 18**	25.0秒	**18 / 18**	**18 / 18**	29.4秒	8 / 16	10 / 16	33.6秒	84.61%	96.15%	90.38%	88.46%

XML 语法正确性方面，BTGenBot-2 几乎达到完美，优于偶尔会生成无效输出的其他模型。BTGenBot-2 展现出出色的能力，能够生成语法正确且完全兼容 BehaviorTree.CPP 的行为树。在语义正确性方面，我们的模型对任务意图的把握最为准确，其次是 GPT-5 Thinking；GPT-5 Instant、Claude Opus 4.1 和 BTGenBot 则表现较差。采用单样本提示时，每个模型都会收到一个行为树生成示例。所有模型都从这个额外示例中获益，专有模型的平均成功率约为 80%。BTGenBot-2 表现最佳，成功率达到 92.38%；尽管单样本提示更复杂、平均推理时间更长，它仍然是速度最快的模型。在非功能性指标方面，只有 BTGenBot-2 和 GPT-5 Thinking 达到了完美的动作一致性，不过 GPT-5 Thinking 在其他指标上的表现较差。考察具有错误恢复功能的变体 BTGenBot-2-ER 可见，集成两个验证器进一步提升了零样本和单样本设置下的性能，平均成功率分别达到 90.38% 和 98.07%。值得注意的是，在几乎所有情况下，只需一次迭代就足以成功修正行为树。这一改进的代价微乎其微：由于需要重新生成有误的行为树，平均推理时间仅增加不到一秒。XML 语法正确率达到 100%，确保生成的行为树完全有效且可执行。零样本和单样本场景下的结果均表明，即使将错误恢复模块应用于 Claude Opus 4.1 这类通用大语言模型，也能带来益处。值得注意的是，错误恢复模块提高了模型的成功率，尤其是在困难任务上，同时显著改善了所有非功能性指标，实现了完美的动作一致性和很高的语法正确率。总体而言，结果表明，BTGenBot-2-ER 在各类任务和提示设置下，兼具最强的可靠性、结构保真度与效率。它大幅优于先前的 BTGenBot，并始终胜过专有模型，充分证明自己是一个稳健有效的工具，能够根据自然语言指令生成复杂行为树。

D. 真实机器人验证

我们通过在实体机器人上开展真实世界试验，评估了本流程的实际适用性。在导航方面，我们使用了 AgileX Scout，这是一款采用滑移转向运动学的轮式移动机器人；在操作方面，我们使用了 SO-ARM 101，这是一款 6 自由度机械臂。在该设置下，我们测试了基准测试中的简单任务，在 Pass@3 下取得了 17/18 的成功率，并在引入错误恢复后提升至 18/18 。这些实验表明，BTGenBot-2 能够有效部署于真实世界场景。由于该基准测试包含超出我们实验室条件限制的复杂任务，表 I 中的结果对模型能力提供了更全面的评估。

V. 结论

我们提出了 BTGenBot-2，这是一个拥有 10 亿参数的开源小语言模型，可将自然语言任务描述转换为可执行行为树。借助精心构建的合成数据集、计算高效微调以及两个新颖的验证器，BTGenBot-2 在保持足够轻量、可部署于资源受限机器人的同时，实现了出色的性能。我们还发布了首个面向基于大语言模型的行为树生成的标准化基准，涵盖 52 项不同难度等级的导航与操作任务，为评估提供了可复现的基础。在 NVIDIA Isaac Sim 中的实验以及对真实机器人的定性测试表明，BTGenBot-2 持续优于先前系统，包括其前代模型以及 GPT-5 和 Claude Opus 4.1 等大型基础模型。这些结果凸显了小型语言模型在机器人任务规划中的优势，弥合了任务的自然语言指令与可在真实机器人上执行的实用表示之间的差距。

参考文献

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat 等，“Gpt-4 technical report”，*arXiv*预印本 *arXiv:2303.08774*，2023年。[2] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang 等，“Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context”，*arXiv*预印本 *arXiv:2403.05530*，2024年。

[3]H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozi`ere, N. Goyal, E. Hambro, F. Azhar 等，“Llama: Open and efficient foundation language models”，arXiv预印本 arXiv:2302.13971，2023年。[4] J. Huang 和 K. C.-C. Chang，“Towards reasoning in large language models: A survey”，arXiv预印本 arXiv:2212.10403，2022年。[5] Z. Zhao、W. S. Lee 和 D. Hsu，“将大语言模型作为大规模任务规划的常识知识”，《神经信息处理系统进展》，第36卷，第31 967–31 987页，2023年。[6] C. H. Song、J. Wu、C. Washington、B. M. Sadler、W.-L. Chao 和 Y. Su，“Llm-planner：借助大语言模型为具身智能体进行少样本落地规划”，载于《IEEE/CVF国际计算机视觉会议论文集》，2023年，第2998–3009页。[7] S. Gupta、K. Yao、L. Niederhauser 和 A. Billard，“动作情境化：利用大语言模型进行自适应任务规划和动作调整”，《IEEE机器人与自动化快报》，2024年。[8] L. Guan、K. Valmeekam、S. Sreedharan 和 S. Kambhampati，“利用预训练大语言模型构建和使用世界模型，以进行基于模型的任务规划”，《神经信息处理系统进展》，第36卷，第79 081–79 094页，2023年。[9] J. Pan、G. Chou 和 D. Berenson，“面向机器人任务规范的自然语言到线性时序逻辑转换器的数据高效学习”，载于《2023年IEEE国际机器人与自动化会议》（ICRA）。IEEE，2023年，第11 554–11 561页。[10] J. Liang、W. Huang、F. Xia、P. Xu、K. Hausman、B. Ichter、P. Florence 和 A. Zeng，“代码即策略：用于具身控制的语言模型程序”，载于《2023年IEEE国际机器人与自动化会议》（ICRA）。IEEE，2023年，第9493–9500页。[11] I. Singh、V. Blukis、A. Mousavian、A. Goyal、D. Xu、J. Tremblay、D. Fox、J. Thomason 和 A. Garg，“Progprompt：利用大语言模型生成情境化机器人任务计划”，载于《2023年IEEE国际机器人与自动化会议》（ICRA）。IEEE，2023年，第11 523–11 530页。[12] M. Colledanchise 和 P. ¨Ogren，《机器人学与人工智能中的行为树：导论》。CRC出版社，2018年。[13] R. Ghzouli、T. Berger、E. B. Johnsen、A. Wasowski 和 S. Dragule，“机器人应用中的行为树与状态机”，《IEEE软件工程汇刊》，第49卷，4267，2023 9，4243 no. pp. – 。[14] S. Macenski、F. Mart´ın、R. White 和 J. G. Clavero，“The marathon 2：一种导航系统 ”，载于《 2020 年 IEEE/RSJ国际智能机器人与系统会议》（IROS）。IEEE，2020年，第2718–2725页。[15] G. Team、T. Mesnard、C. Hardin、R. Dadashi、S. Bhupatiraju、S. Pathak、L. Sifre、M. Rivi`ere、M. S. Kale、J. Love 等，“Gemma：基于Gemini研究与技术的开放模型”，arXiv预印本 arXiv:2403.08295，2024年。[16] E. J. Hu、Y. Shen、P. Wallis、Z. Allen-Zhu、Y. Li、S. Wang、L. Wang、W. Chen 等，“LoRA：大语言模型的低秩适配”。ICLR，第1卷，第2期，第3页，2022年。[17] T. Dettmers、A. Pagnoni、A. Holtzman 和 L. Zettlemoyer，“QLoRA：量化大语言模型的高效微调”，《神经信息处理系统进展》，第36卷，第10 088–10 115页，2023年。[18] R. A. Izzo、G. Bardaro 和 M. Matteucci，“BTGenBot：使用轻量级大语言模型生成机器人任务的行为树”，载于《2024年IEEE/RSJ国际智能机器人与系统会议》（IROS）。IEEE，2024年，第9684–9690页。[19] O. Biggar、M. Zamani 和 I. Shames，“行为树及相关架构的表达能力层级”，《IEEE机器人与自动化快报》，第6卷，第3期，第5397–5404页，2021年。[20] T. Brown、B. Mann、N. Ryder、M. Subbiah、J. D. Kaplan、P. Dhariwal、A. Neelakantan、P. Shyam、G. Sastry、A. Askell 等，“语言模型是少样本学习者”，《神经信息处理系统进展》，第33卷，第1877–1901页，2020年。[21] S. Zhang、L. Dong、X. Li、S. Zhang、X. Sun、S. Wang、J. Li、R. Hu、T. Zhang、F. Wu 等，“大语言模型的指令微调：综述”，arXiv预印本 arXiv:2308.10792，2023年。[22] S. Li、X. Puig、C. Paxton、Y. Du、C. Wang、L. Fan、T. Chen、D.-A. Huang、E. Aky¨urek、A. Anandkumar 等，“用于交互式决策的大语言模型”，《神经信息处理系统进展》，第35卷，第31 199–31 212页，2022年。[23] A. Brohan、Y. Chebotar、C. Finn、K. Hausman、A. Herzog、D. Ho、J. Ibarz、A. Irpan、E. Jang、R. Julian 等，“照我能做的做，而不是照我说的做：将语言落地于机器人可供性”，载于《机器人学习会议》。PMLR，2023年，第287–318页。

[24] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid 等，“Rt-2：视觉-语言-动作模型将网络知识迁移至机器人控制”，载于《机器人学习会议》。PMLR，2023年，第2165–2183页。[25] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi 等，“Openvla：一种开源视觉-语言-动作模型”，arXiv预印本 arXiv:2406 09246，2024年。。[26] C. Y. Kim, C. P. Lee 和 B. Mutlu，“理解由大语言模型（LLM）驱动的人机交互”，载于《2024年ACM/IEEE人机交互国际会议论文集》，2024年，第371–380页。[27] Z. Zhou, J. Song, K. Yao, Z. Shu 和 L. Ma，“Isr-llm：用于长时程序列任务规划的迭代自我优化大语言模型”，《2024年IEEE机器人与自动化国际会议（ICRA）》，第2081–2088页，2023年。 [在线]。可访问：https://api.semanticscholar.org/CorpusID:261245497

[28] A. Lykov 和 D. Tsetserukou，“Llm-brain：基于大语言模型的机器人行为树快速生成”，载于《2024年第2届基础模型与大语言模型国际会议（FLLM）》。IEEE，2024年，第392–397页。[29] J. Ao, F. Wu, Y. Wu, A. Swiki 和 S. Haddadin，“Llm-as-bt-planner：利用大语言模型生成用于机器人任务规划的行为树”，载于《2025年IEEE机器人与自动化国际会议（ICRA）》，2025年，第1233–1239页。[30] X. Chen, Y. Cai, Y. Mao, M. Li, W. Yang, W. Xu 和 J. Wang，“结合意图理解与最优行为规划，根据人类指令生成行为树”，载于《国际人工智能联合会议》，2024年。 [在线]。可访问：https://api.semanticscholar.org/CorpusID:269758016

[31] H. Zhou, Y. Lin, L. Yan, J. Zhu 和 H. Min，“Llm-bt：基于大语言模型与行为树执行机器人自适应任务”，载于《2024年IEEE机器人与自动化国际会议（ICRA）》。IEEE，2024年，第16 655–16 661页。[32] J. Styrud, M. Iovino, M. Norrl¨of, M. Bj¨orkman 和 C. Smith，“利用大语言模型自动扩展用于机器人操作的行为树”，载于《2025年IEEE机器人与自动化国际会议（ICRA）》，2025年，第1225–1232页。[33] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan 等，“Llama 3 模型家族”，arXiv预印本 arXiv:2407.21783，2024年。[34] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang 和 T. B. Hashimoto，“Alpaca：一种强大且可复现的指令跟随模型”，斯坦福基础模型研究中心。https://crfm. stanford. edu/2023 7，2023 03 13 3，6，/ /alpaca. html，第。号。页。[35] O. Honovich, T. Scialom, O. Levy 和 T. Schick，“非自然指令：以（几乎）无需人工劳动的方式微调语言模型”，arXiv预印本 arXiv:2212.09689，2022年。[36] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark 等，“训练计算最优大语言模型”，arXiv预印本 arXiv:2203.15556，2022年。[37] H. Bansal, A. Hosseini, R. Agarwal, V. Q. Tran 和 M. Kazemi，“更小、更弱，却更优：通过计算最优采样训练大语言模型推理器”，载于《第十三届国际学习表征会议》，2024年。[38] C.-Y. Lin，“ROUGE：一种用于摘要自动评估的工具包”，载于《文本摘要分支拓展》。西班牙巴塞罗那：计算语言学协会，2004年7月，第74–81页。 [在线]。可访问：https://aclanthology.org/W04-1013/ [39] K. Papineni, S. Roukos, T. Ward 和 W.-J. Zhu，“BLEU：一种用于机器翻译自动评估的方法”，载于《第40届计算语言学协会年会论文集》，2002年，第311–318页。[40] S. Gugliermo, D. C. Dom´ınguez, M. Iannotta, T. Stoyanov 和 E. Schaf- fernicht，“行为树评估”，《机器人与自主系统》，第178卷，第104714页，2024年。[41] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman 等，“对基于代码训练的大语言模型进行评估”，arXiv预印本 arXiv:2107 03374，2021年。。