**Describe, Explain, Plan and Select: Interactive Planning with Large Language Models Enables Open-World Multi-Task Agents**

描述、解释、规划和选择：支持开放世界多任务代理的基于大语言模型的交互式规划

**太长不看版：**

论文主要研究了在开放世界环境中多任务代理的任务规划挑战。任务规划主要有两个困难：(1). 由于任务是长线的，因此在开放世界中实现任务规划需要准确和多步骤推理。(2). 由于普通计划器在复杂计划中排序并行子目标时不会考虑当前代理实现给定子任务的难易程度，因此生成的计划可能效率低下甚至不可行。 为此，作者提出了基于大型语言模型的交互式规划框架 (Describe，Explain，Plan and Select, DEPS)。该框架由基于LLM的规划器和解释器、可训练的选择器和由事件触发的描述器构成。该框架的执行流程为规划器将给定任务拆分成一系列的子任务，并通过一个可训练的选择器，根据当前状态选择当前要执行的任务。当执行任务失败后，触发错误事件并激活描述器，描述器根据当前状态和任务执行结果生成总结文本发送给解释器。LLM扮演解释器根据当前的计划和总结文本定位到错误原因，并转换角色到规划器根据原因生成新的规划并重新分解为一系列子任务供选择器选择，通过不断迭代这样的过程实现任务目标。该框架可以稳健地完成70多个Minecraft任务，并将整体性能提高近一倍。更进一步的测试表明，交互式规划框架在非开放式领域中同样有效。

**摘要：**

论文主要研究了在开放世界环境中多任务代理的任务规划挑战。任务规划主要有两个困难：(1). 由于任务是长线的，因此在开放世界中实现任务规划需要准确和多步骤推理。(2). 由于普通计划器在复杂计划中排序并行子目标时不会考虑当前代理实现给定子任务的难易程度，因此生成的计划可能效率低下甚至不可行。 为此，作者提出了基于大型语言模型的交互式规划框架 (Describe，Explain，Plan and Select, DEPS)。DEPS通过描述计划执行过程并在扩展规划阶段失败时根据反馈给出自我解释，从而更好的纠正LLM生成的初始计划的错误。此外，它还包括一个目标选择器，这是一个可训练的模块，根据估计的完成步骤对并行候选子目标进行排名，从而优化初始计划。该框架标志着第一个多任务代理的里程碑，它可以稳健地完成70多个Minecraft任务，并将整体性能提高近一倍。更进一步的测试表明，交互式规划框架在非开放式领域中同样有效。

1. **引言：**

开发能够在复杂领域完成大量、多样化任务的多任务代理被认为是具有普遍能力人工智能 (AGI) 的关键里程碑之一。早期的研究使用分层目标执行结构，即计划器生成行动计划、低级目标条件控制器执行，这种方法在非开放领域取得了很好的效果。然而经过作者的对比实验表明，当该方法应用于开放世界的环境下 (Minecraft为例) 效果远远低于非开放领域。作者认为原因主要有以下两个方面，(1). 由于开放世界丰富的对象类型和复杂的依赖关系，任务通常需要被拆分成一长串具有严格依赖关系的子目标。而在桌面任务中，通常仅被拆分成几个子目标。(2). 制定的计划需要在开放世界中具有可行性，即当可以根据执行多个可能的路线完成任务时，代理需要根据当前的状态选择最佳的执行路线。

为了解决上述的两个问题，作者提出了“描述、解释、规划、选择” (DEPS)方案，即基于大语言模型，通过交互式规划的方法制定计划。首先，该方法通过在失败时有效的调整生成的计划解决第一个问题。具体来说，当控制器不能完成子目标时，描述器需要将当前的情况总结成文字，发送给基于LLM的规划器，并通过prompt使基于LLM的解释器找到计划中的错误。最后，规划器使用描述器和解释器的信息重新调整计划，提高计划的可行性。其次，该方法使用学习过的目标选择器，根据每个候选子目标与子任务的接近程度做出计划选择。DEPS方法的架构如图1所示。

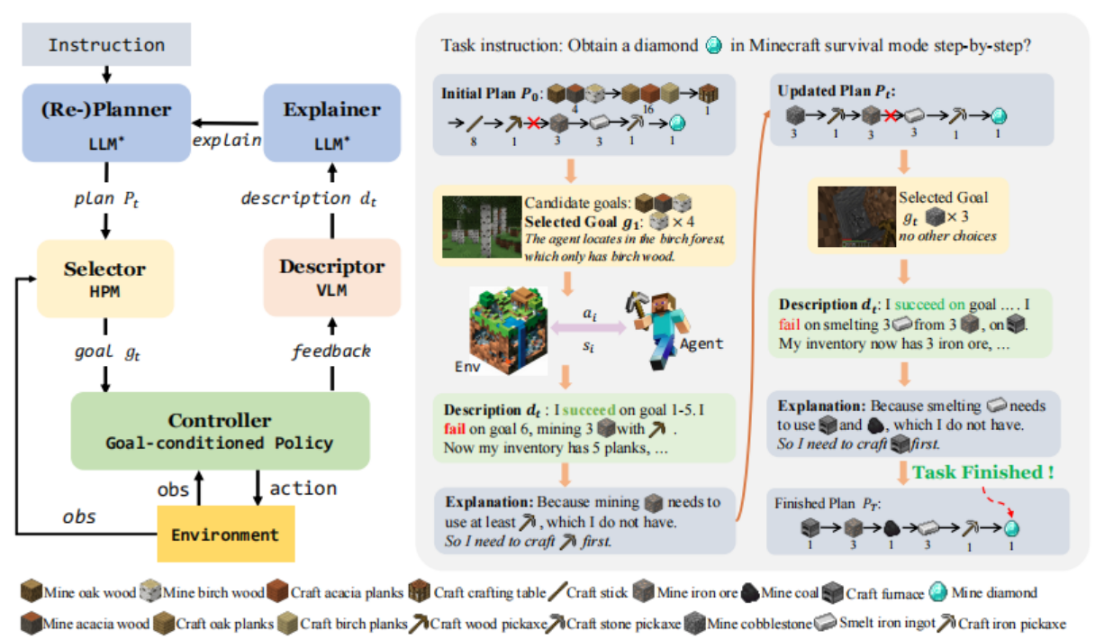


图1. 交互式规划方法的示例。

该方法在“Minecraft”中的71个任务上进行无演示测试，通过给控制器原子性的子任务，基于LLM的计划器可以在有限的步骤内完成所有任务。在相同的初始状态和目标条件控制器下，DEPS比所有语言规划器基线的总体成功率提高了近两倍。此外，DEPS在ALFWorld和桌面环境上，与现有基于LLM的规划方法相比，实现了50%以上的提升。

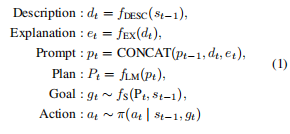
1. **背景**

作者设计了一种控制器和规划器的组合方法，规划器将计划将长序列任务分解为一系列子任务，然后由控制器执行。在第t个时间步数中，基于当前状态和指定子目标，即目标条件策略，生成当前动作。

此前的工作在开放世界任务中的挑战是 (1). 直接从任务描述中生成一个完美计划的概率，随着任务长度的增长而降低。(2). 即便生成了正确的计划，该计划执行起来也是十分困难甚至不能执行的。

1. **在典型开放世界中实现可靠规划**
   1. **DEPS概述**

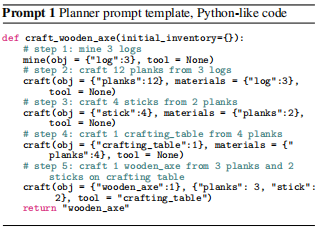
DEPS框架由事件触发描述器、基于LLM的解释器和规划器、时域预测目标选择器和目标条控制器构成。给定一个目标命令作为任务T，规划器将这个长序列任务分解为一系列子目标作为初始计划。然后通过控制器，根据目标条件策略按序执行子目标。当执行子目标发生错误时，描述器将会被错误事件激活，并把最近一个目标的当前状态和执行结果汇总为一个文本发送给LLM。当LLM接收到描述器发送的错误描述时，它将扮演解释器首先尝试定位中的错误，然后扮演规划器重新规划当前的任务T，并生成一个可执行的计划。上述过程的数学表述如公式(1)所示，DEPS将根据上面的描述迭代的更新计划直到任务完成。为了能够过滤掉低效的计划，作者通过训练选择器预测当前状态下的一组并行目标中实现每个目标所需要的时间步数，并根据时间步数选择合适的目标作为当前目标。



以我的世界为例，给定的任务为“获取钻石”，规划器将这一任务分解为一系列任务。当任务无法使用木制的鹤嘴锄完成时，将会触发错误事件并激活描述符。描述符将当前状态和执行结果汇总为文本发送给LLM。当LLM接收到错误文本描述时，将扮演解释器定位先前计划中的错误为需要使用石制的鹤嘴锄才能完成任务，然后扮演规划器重新生成新的任务。在选择器根据当前状态和计划实现的目标进行选择时，如果处于热带草原环境中，砍下相思树比砍下橡树具有更少的时间步数，砍下相思树则会成为当前的目标。

* 1. **使用LLM进行描述、解释和规划，并生成可执行计划**

在3.1中，作者已经详细的描述了DESP框架的执行过程。在本小节中将为这些过程补充一些细节。作者将Prompt重写为交互式对话格式，并使用格式化的Prompt提高计划的可读性和可解释性，同时方便定位错误，如Prompt1所示。



Prompt1. 规划器Prompt模板。

描述器收到的执行任务期间的反馈可以来自于用户，也可以来自于与训练的视觉语言模型CLIP，前一种需要密集的人工参与，后一种需要针对特定领域微调，这降低了框架的自动化程序和泛化能力。为了能够将非结构化信息转化为结构化的信息，作者将游戏中可用的符号转化为反馈描述，并进一步提取与计划相关的消息，避免在Prompt中携带不相关的信息。如图1所示。

为了能够使解释器更好的定位错误，作者向LLM提供了少量的演示，思维链提示。

* 1. **时域预测择器选择可执行的计划**

为了提高计划的效率，作者使用了选择器选择执行成功率最高的最有效的路径作为最终的计划。首先，作者设计了一个状态感知选择器，它通过预测当前状态和计划下的目标分布 (其中，描述了中的所有当前可执行目标) 从候选目标集中选择状态下时间步长最少的目标作为当前目标。实现选择器的一种直接方法是使用视觉语言模型，通过利用当前状态和目标文本之间的语义相似性进行选择。然而，这可能并不能完全反映完成目标的困难，因为视觉语言模型缺乏实践经验。例如，在代理前面的一棵“橡树”可能会导致“砍树”目标的高度语义相似性，但如果在代理和橡树之间有一个峡谷，那么实现这一目标可能会低效。

为了缓解这种情况，作者实现了一个水平预测选择器，它嵌入了实际的任务经验，以根据目标的效率和可行性对目标进行准确的排序。在这里，我们定义一个目标时域范围作为完成给定目标的剩余时间步长，其中是完成目标g的时间。这个度量准确地反映了我们从当前状态实现给定目标的速度。为了预测时域，作者训练了一个神经网络µ，通过最小化熵损失来拟合执行轨迹，其中是实际完成目标g的执行轨迹。因此，目标分布可以表述为：

我们将目标敏感的Impala CNN作为选择器的backbone。时域预测选择器与控制器联合训练，并共享backbone。

1. **实验**

**4.1. 实验设置**

作者从我的世界中选取了71个任务，并将其分为8个元组MT1-MT8。作者将DEPS与GPT、ProgPrompt、Chain-of-Thought、Inner Monologue进行比较。对于所有的基线模型，作者在Prompt中使用相同的演示示例，基于OpenAI的相同的LM，并且为每一个基线模型设计提示和反馈模板保证所有规划器的方法都通过。

**4.2. 主要成果**

每个任务执行30次，平均结果如表2所示。DEPS框架在所有方法中取得了最好的性能。从MT6开始，几乎所有的现有的LLM规划框架均失败，DEP方法（没有选择器的DEPS）以显著的优势击败了所有的LLM规划框架。这验证了DEPS可以找出当前计划失败的原因，并纠正最初有缺陷的计划。但由于情节长度的限制，最终的成功率仍然有待提升。

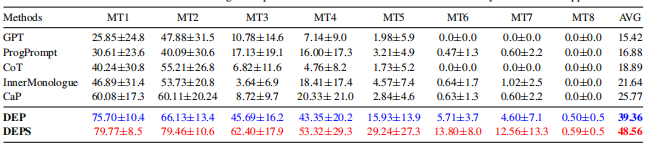


表2. DEPS和现有的LLM规划款就改在《我的世界》任务101上的成功率。

**对不同的控制器和不同的《我的世界》版本的鲁棒性。**作者还评估了MineRL 和MC-Textworld上的DEPS。值得注意的是，DEPS是一种规划方法，它需要装备目标条件控制器来与《我的世界》环境交互。作者选择mc-控制器和Steve-1控制器，分别与Minedojo和MineRL交互。这两种方法都是感知视觉部分观察并产生鼠标和键盘动作的控制策略。而MC-Textworld是一个文本世界，它只保留《我的世界》制作食谱和挖掘规则。所以mc-文本世界不需要控制器。任务集MT1-MT8在不同的《我的世界》环境下的DEPS结果如表3所示。研究结果表明，DEPS可以在各种《我的世界》环境中生成有效的计划。对MC-Textworld的结果也表明，从MT6到MT8，更困难的任务集的性能下降主要来自于控制器的限制。

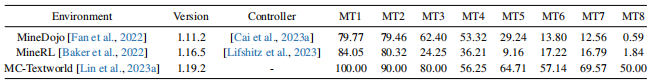


表3. DEPS在不同的《我的世界》环境下的成功率。

**4.3. 消融实验**

**4.3.1 选择器消融**

作者验证了选择器在不同平行子目标下的鲁棒性。代理被要求分别完成2、3和4个候选目标（所有目标的先决条件都是一致的）。该任务的目标对应于不同种类的怪物或材料。作者报告了使用不同选择器的DEP的最终成功率，包括使用固定的目标序列，随机的目标序列，以及基于MineCLIP，CLIP，以及我们的水平预测选择器（HPS）。如图4所示，在一轮平行的候选目标中，与没有任何选择器相比，使用我们的水平预测选择器分别提高了∆=+22.3%，+29.2%，+32.6%。在有限的情节长度下，如1000步，使用选择器显示出更大的优势，这证明了选择器可以提高计划在环境中的执行效率。此外，与使用CLIP和MineCLIP等视觉语言模型作为目标模型相比，HPS具有最好的性能。

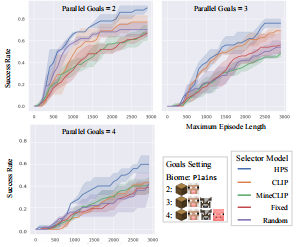


图4. 不同选择器在不同平行目标和最大事件时间步长下的成功率。

**4.3.2 重新规划轮数消融**

作者通过增加DEPS的最大轮数，在所有任务上评估。每一轮为一个交互式规划周期，包括描述、解释、规划和选择，即一个更新的计划。每轮最多任务的所有任务均执行30次，平均成功率见表4。作者将常规的LLM规划框架作为基线，即框架将最初生成的计划作为最终的执行计划，在任务执行过程中没有涉及任何描述、重新规划或自我解释过程。作者还报告了表4中∆列中每个元任务从普通规划框架到DEPS的成功率增量。这组实验表明，DEPS可以在开放世界环境中迭代地改进其计划。

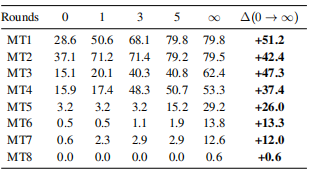


表4. 不同最大轮重新计划下DEPS的成功率。

**4.4. 获得钻石挑战**

作者将获得钻石任务作为一个补充实验，来展示DEPS在具体环境中的复杂任务的能力。

1. **局限性**

DEPS框架至少有两个主要的限制。(1). 该框架依赖于私有LLM，如GPT-3和ChatGPT，这使得使用成为了一个问题。(2) 规划器的瓶颈阻碍了模型被进一步扩展。此外，以前在规划方面的一些基本挑战可能在所采用的环境中并不普遍，因此可能会被忽视。

1. **结论**

作者研究了在开放世界中的规划问题，确定了这些环境所特有的两个主要挑战： (1) 长序列任务需要精确和多步骤的推理，(2)由于规划器没有考虑到任务执行的难易程度，导致了规划不可行或难以被完成。因此作者提出了“描述、解释、计划和选择”（DEPS）来解决这两个问题，这是一种基于大型语言模型的交互式规划框架。作者在具有挑战性的《我的世界》领域的实验验证了该框架的优越性，稳定完成70个+《我的世界》的任务。同时DEPS也是第一个可以在这个游戏中完成获取钻石任务的规划框架。