arXiv:2303.17580v2 [cs.CL] 2023 年 4 月 2 日

HuggingGPT：在拥抱的脸上与 ChatGPT 及其朋友一起解决人工智能任务

沈永良 [1*]，Kaitao Song[2*]，徐谭 [2]，李东生 [2]，Weiming Lu[1]，岳亭庄 [1]
浙江大学 [1]，微软亚洲研究 [2]
{Syl, luwm, yzhuang}@zju.edu.cn， {Kaitaosong, xuta, dongsli}@microsoft.com

摘要

解决不同领域和模式的复杂人工智能任务是迈向高级人工智能的关键一步。虽然有丰富的人工智能模型可供 d 使用如果是不同的领域和模式，它们无法处理复杂的人工智能任务。 考虑到大型语言模型（LLM）在语言理解、生成、交互和推理方面表现出非凡的能力，我们主张 LLM 可以充当 Ntroller 用于管理现有的人工智能模型，以解决复杂的问题
人工智能任务和语言可以成为增强其能力的通用界面。 基于这一理念，我们提出了 HuggingGPT，这是一个利用 LLM（例如 ChatGPT）连接机器学习社区（例如 Hugging Face）中的各种 AI 模型的框架，以解决 AI 任务。 具体来说，我们在收到用户请求时使用 ChatGPT 进行任务规划，根据 Hugging Face 中可用的功能描述选择模型，使用选定的 AI 模型执行每个子任务，并总结响应命令执行结果。 通过利用 ChatGPT 强大的语言能力和 Hugging Face 中丰富的 AI 模型，HuggingGPT 能够在不同模式和领域涵盖许多复杂的人工智能任务，并取得令人印象深刻的结果 在语言、视觉、语言和其他具有挑战性的任务方面，这为先进的人工智能铺平了一条新道路 [2].

1 简单介绍

大型语言模型（LLM）[1、2、3、4、5、6]，如 ChatGPT，引起了两个学术界的极大关注和行业，由于他们在各种自然语言处理（NLP）任务上的出色表现。 基于大规模文本语料库的大规模预训练和来自人类反馈（RLHF）的强化学习，LLM 可以在 语言理解、生成、互动和推理。 LLM 的强大能力也推动了许多紧急研究课题（*例说*，上下文学习[1，7，8]，教学学习[9，10，11，12]和思想链提示[13，14，15，16]）进一步调查 LLM 的巨大潜力，并为我们构建先进的人工智能系统带来无限的可能性。

尽管取得了这些巨大成功，但当前的 LLM 技术仍然不完美，并面临一些紧迫的问题正在构建先进的人工智能系统。 我们从以下方面讨论它们：1）仅限于文本生成的输入和输出形式，当前的 LLM 缺乏处理视觉和语音等复杂信息的能力，无论它们如何 r 在 NLP 任务方面取得重大成就；2）在现实世界中，一些复杂的任务通常由多个子任务组成，因此需要多个模型的调度与合作，这些模型是 也超出了语言模型的能力；3）对于一些具有挑战性的任务，LLM 在零射中表现出优异的结果

---

[*] 前两位作者的贡献相等。 这项工作是在第一作者在微软亚洲研究公司实习时完成的。

[2] https://github.com/microsoft/JARVIS.

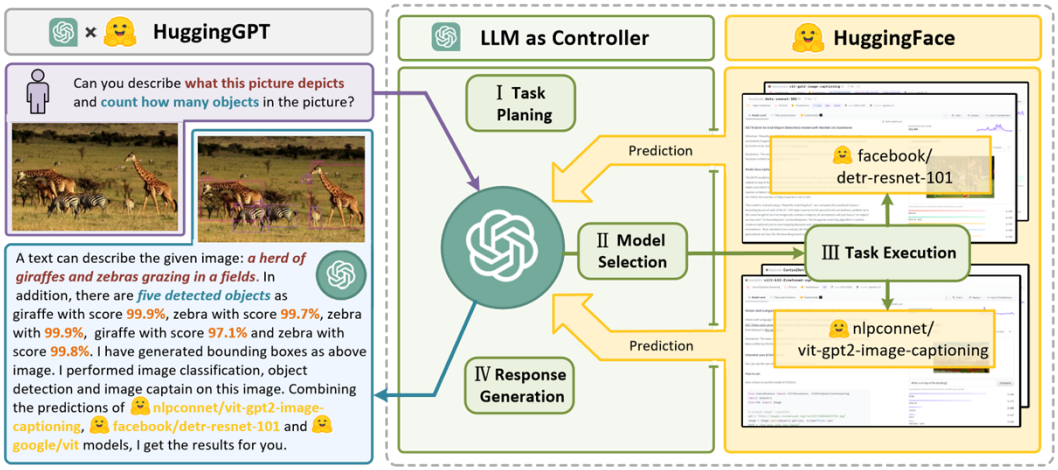


图 1： *语言是 LLM（例如 ChatGPT）连接众多人工智能模型的界面（例如，那些拥抱脸的人）用于解决复杂的人工智能任务。* 在这个概念中，LLM 充当控制器，管理和组织专家模型的合作。 法长首先根据用户请求计划任务列表，然后分配专家德尔斯完成每项任务。 专家执行任务后，LLM 会收集结果并回复用户。

或几枪设置，但它们仍然比一些专家弱（*例说*，微调模型）。 如何解决这些问题可能是第一个也是 LLM 朝着更先进的人工智能系统迈出了关键一步。

在本文中，我们指出，为了处理复杂的人工智能任务，LLM 应该能够与外部模型协调，以利用其能力。 所以，关键是如何选择 suitable 中间件来连接 LLM 和 AI 模型之间的联系。 为了解决这个问题，我们注意到，通过总结其模型函数，每个 AI 模型都可以将其表示为一种语言形式。 因此，我们引入了一个概念：*"语言是一个通用的 interLLM 连接人工智能模型的面孔"*。 换句话说，通过将这些模型描述纳入提示中，LLM 可以被视为管理规划、调度和合作等人工智能模型的大脑。 因此，这种策略使 LLM 能够调用外部模型来解决人工智能任务。 但是，如果我们想将多个人工智能模型集成到 LLM 中，将出现另一个挑战：解决许多人工智能任务需要收集大量高质量的模型描述，这需要大量提示摇摇大气。 巧合的是，我们注意到，一些公共 ML 社区通常提供各种适用的模型，并带有明确定义的模型描述，以解决语言、视觉和语音等特定人工智能任务。 这些观察给我们带来了一些灵感：我们可以链接 LLM 吗（ *例说*，ChatGPT）与公共 ML 社区（*例说*，GitHub，拥抱脸 [1]，Azure 等）通过基于语言的界面解决复杂的人工智能任务？

因此，在本文中，我们提出了一个名为 拥抱 GPT 连接 LL 女士（*也就是说*，、ChatGPT）和 ML 社区（*也就是说*，，拥抱脸），它可以处理来自不同模式的输入，并解决许多复杂的人工智能任务。 更具体地说，对于 Hugging Face 中的每个 AI 模型，我们使用 libr 中的相应模型描述 Ary，并将其融合到提示中，以建立 与 ChatGPT 的连接。 之后，在我们的系统中，LLM（*也就是说*，，ChatGPT）将充当大脑来确定用户问题的答案。 如图 1 所示，HuggingGPT 的整个过程 N 分为四个阶段：

- 任务规划： 使用 ChatGPT 分析用户理解其意图的请求，并通过提示将其分解为可能可解决的任务。
- 型号选择： 为了解决计划中的任务，ChatGPT 选择专家 m 根据模型描述托管在 Hugging Face 上的 odel。
- 任务执行： 调用并执行每个选定的模型，并将结果返回到 ChatGPT。

[1] https://huggingface.co/models

• 响应生成： 最后，使用 ChatGPT 集成所有模型和 genrat 的预测 E 为用户提供答案。

HuggingGPT 受益于这种设计，能够使用外部模型，因此可以集成多模式感知能力并处理多个复杂的人工智能任务。 此外，这条管道还允许我们的 HuggingGPT 继续 吸收特定任务专家的力量，实现可增长和可扩展的人工智能能力。

到目前为止，我们的 HuggingGPT 已经在 ChatGPT 周围的 Hugging Face 上集成了数百个模型，涵盖了文本分类、对象检测等 24 项任务躁狂分割、图像生成、问题回答、文本到语音和文本到视频。 实验结果证明了 HuggingGPT 处理多模态信息和复杂人工智能任务的能力。

总之，我们的贡献是一个如下：

1. 为了补充大型语言模型和专家模型的优势，我们提出了模型间合作协议。 大语言模型充当规划和决策的大脑，小模型充当每个样本的执行器 C 任务，为设计通用人工智能模型提供新方法。
2. 我们构建了 HuggingGPT，通过将 Hugging Face 集线器与围绕 ChatGPT 的 400 多个特定任务模型集成来处理通用人工智能任务。 通过模型的开放协作，HuggingGPT 为用户提供多模式和可靠的对话服务。
3. 前夫对跨语言、视觉、语音和跨模式的多个具有挑战性的人工智能任务的紧张实验证明了 HuggingGPT 从多种模式和领域理解和解决复杂任务的能力。

2 相关作品

2.1 大朗古年龄模型

近年来，自然语言处理（NLP）领域因大型语言模型（LLM）的出现而发生了革命性的变化[1、2、3、4、5、6]，例如 GPT3[1]、PaLM[3]和 LLaMa[6]等模型。 LLM 已经证明了我由于其庞大的语料库和密集的训练计算，零射和少射任务以及数学问题解决和常识推理等更复杂的任务具有紧迫的能力。

关于 LLM 的一些活跃研究领域包括 思想链提示（CoT）[13、14、15、16]和指令调优[9、10、11、12]。 [13]提出的思想链提示通过设置几个案例示例来提示大型模型生成解决问题的过程，从而显著地改进模型的推理能力。[14]将这种方法扩展到零镜头思维链，并发现大型模型仍然可以通过使用"*让我们一步一步地思考*"。 相比之下，[15]将程序语言 f 分开罗马自然语言，使大型语言模型能够生成代码逻辑来解决推理问题。 指令调优是大型语言模型应用程序的另一种方法。 [10]、[11]和[9]等研究工作已经收集并改变了 d 将传统的 NLP 任务数据集转换为指令，并对指令数据集上的大模型进行微调，以提高未知任务的泛化能力。 通过指令调优，Flan-T5[12]和 Flan-UL2 的表现优于 650B PaLM [3]型号 u 只唱 100B 参数。 另一方面，InstrutGPT 和 ChatGPT 使用从人类反馈技术中强化学习，使语言模型与指令保持一致，从而获得卓越的语言理解和生成能力。

2.2 LLM 功能的进步

为了将大型语言模型（LLM）的范围扩展到文本生成之外，当代研究研究了两种主要方法。首先，一些作品设计了统一的多式联运语言模型，如 BLIP-2 [17]，它使用 Q-former 来协调语言和视觉语义，Kosmos-1[18]，它将视觉输入集成到文本序列中，以合并语言和视觉输入。其次，其他研究侧重于 exte 的整合 Rnal 工具或模型。 开创性的 Toolformer[19]在文本序列中引入了外部 API 标签，方便 LLM 访问外部工具。 因此，许多作品扩展了 LLM，以涵盖视觉模式。 Visual ChatGPT [20]保险丝带有 LLM 的视觉基础模型，如 BLIP[21]和 ControlNet[22]。 可视化编程[23]和 ViperGPT[20]通过使用编程语言将 LLM 应用于视觉对象，将可视化查询解析为以 Python cod 表示的可解释步骤 E. 此外，研究人员还努力使这些 LLM 适应专门的视觉任务。 例如，先知[24]和 ChatCaptioner[25]分别将 LLM 纳入视觉问题回答和图像字幕任务中。

与这些约有区别疼痛，我们提议的 HuggingGPT 通过以下方式推进了更通用的人工智能功能：1）HuggingGPT 使用大型语言模型作为接口，将用户请求路由到专家模型，有效地结合了语言理解能力具有其他专家模型专业知识的大型语言模型。 2）HuggingGPT不限于视觉感知任务，还可以通过大型语言模型组织模型之间的合作来解决任何模式或领域的任务。 钨在大型语言模型的规划中，可以有效地指定任务过程并解决更复杂的问题。 3）HuggingGPT 采用更开放的方法，根据模型描述分配和组织任务。 通过只提供 model 描述，HuggingGPT 可以持续方便地集成各种专家模型，而无需更改任何结构或提示设置。 这种开放和持续的方式使我们离实现更先进的人工智能又近了一步。

3 拥抱 GPT

HuggingGPT 是一个协作系统，由作为控制器的大型语言模型（LLM）和作为协作执行器的众多专家模型组成。 HuggingGPT 的工作流程包括四个阶段：任务规划、模型选择、任务执行和响应生成，如图 2 所示。 1）LLM（例如 ChatGPT）首先解析用户请求，将其分解为多个任务，并根据其知识规划任务顺序和依赖性；2）LLM 分发解析的任务 o 根据 Hugging Face 中的模型描述进行专家模型；3）专家模型在推理端点上执行分配的任务，并将执行信息和推断结果记录给 LLM；4）最后，LLM 总结了执行过程 Ess 日志和推断结果，并将摘要返回给用户。

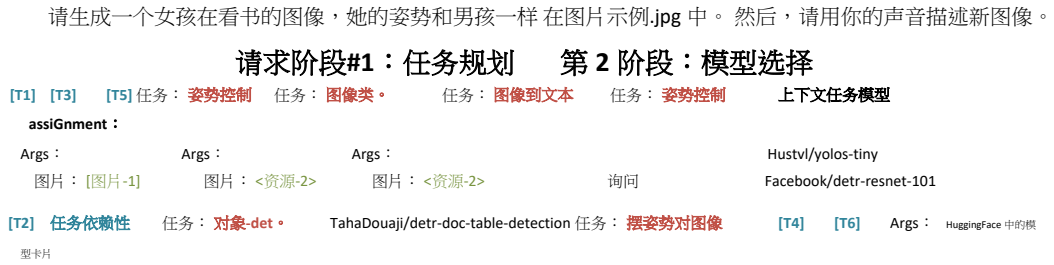
3.1 任务规划

在 HuggingGPT 的第一阶段，大型语言模型接受用户的请求并将其分解为一系列结构化任务。复杂的请求通常涉及多个任务，大型语言模型需要确定依赖项和 这些任务的执行顺序。 为了提示大型语言模型进行有效的任务规划，HuggingGPT 在其提示设计中采用了基于规范的指令和基于演示的解析。 我们在下面的 paragrap 中介绍细节 Hs。

基于规范的指令　任务规范为任务提供了一个统一的模板，并允许大型语言模型通过插槽归档进行任务解析。 HuggingGPT 设计了四个用于任务解析的插槽，即任务类型，任务 ID、任务依赖项和任务参数：

- 任务 ID 为任务规划提供了一个唯一的标识符，用于引用依赖任务及其生成的资源。
- 任务类型涵盖语言、视觉、视频、音频等方面的不同任务。 C 当前支持的 HuggingGPT 任务列表见表 1、2、3 和表 4。
- 任务依赖项定义了执行所需的先决条件任务。 只有当所有先决条件相关任务完成时，任务才会启动。
- 任务参数包含任务执行所需的参数列表。 它包含三个子字段，根据任务类型填充文本、图像和音频资源。 它们由用户的请求或依赖 tas 生成的资源解决 Ks。 表 1、2、3 和表 4 显示了不同任务类型的相应参数类型。

受益于指令调优[9]和从人类反馈中强化学习[2]，大型语言模型能够遵循指令。 拥抱 gingGPT 向大型语言模型提供这些任务规范，作为分析用户请求并相应解析任务的高级说明。



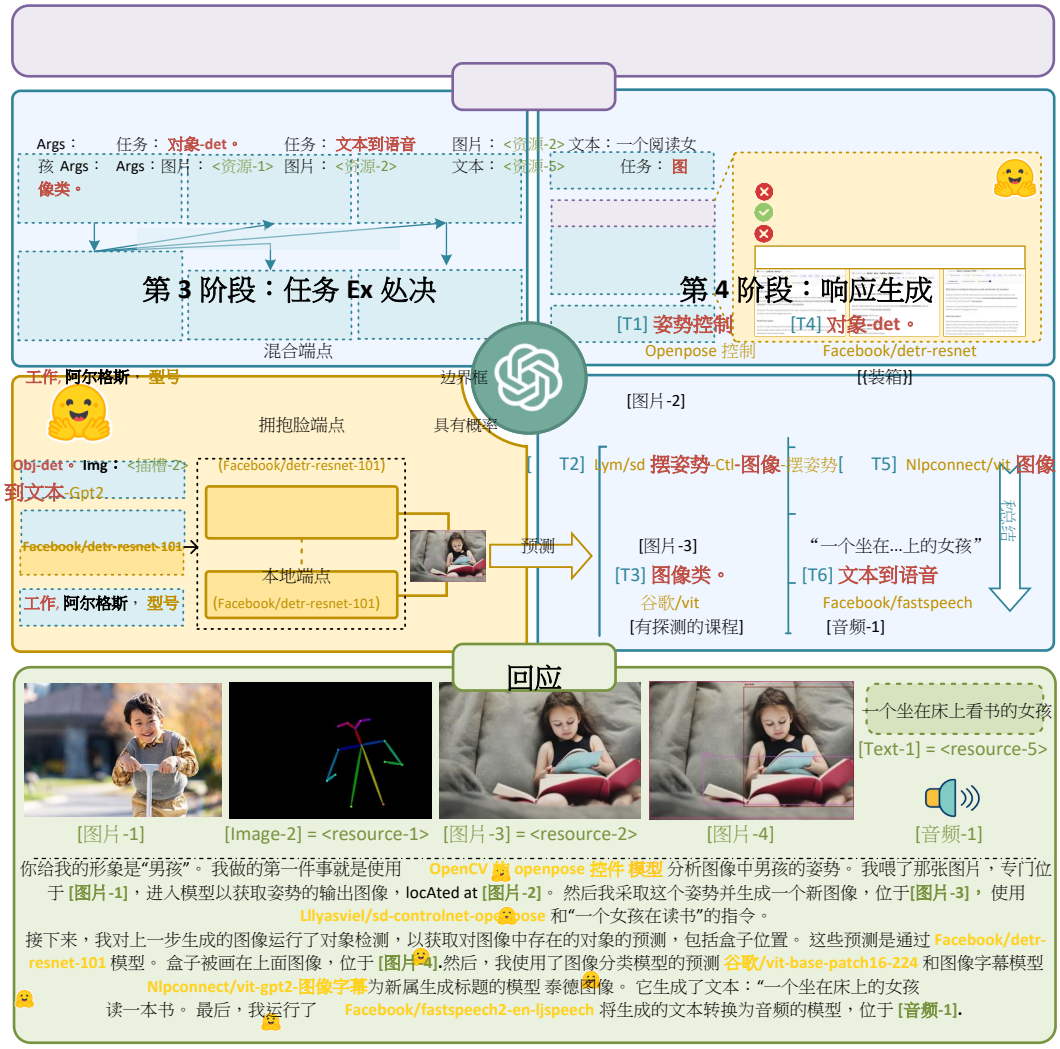


图 2：HuggingGPT 概述。 以 LLM（例如 ChatGPT）为核心控制器，以专家模型为执行者，HuggingGPT 的工作流程由四个阶段组成：1） 任务规划：LLM 将用户请求解析为任务列表，并确定 执行顺序和任务之间的资源依赖性；2） 型号选择：LLM

工作	阿尔格
文本-cls	文
Token-cls	本
文本 2 文本生成	文
总结	本
翻译	文
回答问题	本
交谈的	文
文本生成	本
表格-cls	文
	本
	文
	本
	文

工作	阿尔格
图像到文本	图像图像
文本到图像	文本+图像
VQA	图像
细胞分裂	文本+图像
DQA	图像图像
Image-cls	图像图像
图像到图像	
对象检测控制网-sd	

工作	阿尔格
文本到语音	文本
音频-cls	声 音 回路
ASR	声 音 回路
音频到音频	声 音 回路

根据拥抱脸专家模型的描述为任务分配适当的模型；3） 任务执行：混合端点的专家模型执行分配的任务 baSed 关于任务顺序和依赖项；4） 响应生成：LLM 集成了专家的推断结果，并生成工作流程日志摘要以响应用户。

本文本	

表 3：音频任务。

工作	阿尔格
文本到视频	文本
视频-cls	视频

表 1：NLP 任务。 表 2：简历任务。表 4：视频任务。

基于演示的假释 HuggingGPT 引入了上下文学习，以进行更有效的任务解析和规划。 通过在提示中注入几个演示，HuggingGPT 允许大型语言模型更好地理解任务规划的意图和标准。 每个的演示是一组关于任务规划的输入和输出-用户的请求和要解析的预期任务序列。 此外，这些演示包括从用户请求解析的任务之间的依赖性，有效地帮助 HuggingGPT 理解任务之间的逻辑关系，并确定执行顺序和资源依赖性。

此外，对话的上下文管理对聊天机器人至关重要，因为它提供聊天日志来促进理解用户请求的 ension。 为了将聊天上下文纳入任务规划阶段，我们在说明中附上了以下段落： *聊天日志被记录为{{聊天日志}}。 从聊天日志中，您可以找到任务计划的历史记录资源 Ing 。*，在哪里 *{{聊天日志}}* 是 HuggingGPT 和用户之间的聊天日志。

3.2 模型选择

解析任务列表后，HuggingGPT 接下来需要匹配任务和模型，即为任务列表中的每个任务选择适当的模型。 为此，我们首先从 Hugging Face Hub 获取专家模型的描述，然后 通过上下文任务模型分配机制动态选择任务模型模型。 这种做法允许增量模型访问（只需提供专家模型的描述），并且更加开放和灵活。

模型描述 专家 mHugging Face Hub 上托管的 odels 附有全面的模型描述，这些描述通常由开发人员提供。 这些描述包含有关模型功能、架构、支持的语言和域、许可证的信息 Ing，还有更多。 这些信息有效地支持了 HuggingGPT 根据用户请求和模型描述的相关性为任务选择正确模型的决定。

上下文中任务模型分配 我们处理任务和模式的分配 Ls 作为单一选择问题，其中潜在模型在给定上下文中作为选项呈现。 通过在提示符中包含用户查询和解析任务，HuggingGPT 可以为手头的任务选择最合适的模型。 然而，由于 const 关于最大上下文长度，并不总是能够在提示中包含所有相关的模型信息。 为了解决这个问题，我们根据任务类型筛选模型，只保留与当前任务类型匹配的模型。 残余然后根据他们在 Hugging Face 上收到的下载次数对 ng 型号进行排名，因为我们认为这个数字在某种程度上反映了模型的质量。 然后，我们选择顶部-K 基于此排名的模型作为候选人 HuggingGPT 可供选择。

3.3 任务执行

一旦任务被分配给特定模型，下一步就是执行任务，即执行模型推断。 为了加快速度和计算稳定性，HuggingGPT 在混合推理上运行这些模型端点。 通过将任务参数作为输入，模型计算推断结果，然后将其发送回大型语言模型。 为了进一步提高推理效率，没有资源依赖性的模型可以并行化。 钍 意味着满足先决条件依赖项的多个任务可以同时启动。

混合端点 一个理想的场景是，我们只在拥抱脸上使用推断端点。 然而，在某些情况下，我们必须部署本地推断 E 端点，例如当某些模型的推断端点不存在时，推断很耗时，或者网络访问受到限制。 为了保持系统的稳定性和效率，HuggingGPT 在本地拉动并运行一些常见或耗时的模型。 公吨本地推断端点很快，但覆盖的模型更少，而 Hugging Face 的推断端点则

#1 任务规划阶段-人工智能助理可以解析用户对几个任务的输入：[{"task": task, "id", task_id, "dep": dependency_task_ids, "args": {"text": text, "image": URL, "audio": URL, "video": URL}}]. “dep”字段表示上一个任务的 ID，该 ID 生成当前任务所依赖的新资源。 一个特殊的标签"<resource>-task_id”指生成的文本图像、音频和视频在依赖项任务中，id 为 task_id。 必须从以下选项中选择任务：{{可用任务列表}}。 任务之间存在逻辑关系，请注意它们的顺序。 如果无法解析用户输入，您需要空回复 JSON。 以下是几个案例供您参考：{{ Demonstrations }}。 聊天记录记录为{{聊天日志}}。
从聊天日志中，您可以找到用于任务规划的用户提到的资源的路径。

演示

相反。 因此，本地端点比 Hugging Face 的推理端点具有更高的优先级。 仅当匹配的模型没有部署 loca 时 lly，HuggingGPT 将在 Hugging Face 端点上运行模型。

资源依赖性 尽管 HuggingGPT 有能力通过任务规划开发任务顺序，但有效管理任务之间的资源依赖性仍然具有挑战性 任务执行阶段。 原因是 HuggingGPT 无法指定未来生成

提示

任务规划

看看/exp1.jpg，你能[{"任务": "image-to-text", "id": 0, "dep": [-1], "Args": {"image": tell me how many objects in "/exp1.jpg" }}, {"task": "object-detection", "id": 0, "dep": [-1] ，图片？ "Args": {"image": "/exp1.jpg" }}]

[{"Task": "image-to-text", "id": 0, "dep":[-1], "args": {"image": "/Exp2.jpg" }}, {"task":"image-Classification", "id": 1, "dep": [-1], In /exp2.jpg, what’s the ani- "args": {"image": "/exp2.jpg" }}, {"task":"object-detection", "id": Mal，它在做什么？ 2, "dep": [-1], "args": {"image": "/exp2.jpg" }}, {"task": "visual-问题-Answering", "id": 3, "dep":[-1], "args": {"text": "What’s the animal doing?", "image": "/exp2.jpg" }}]

给定图像/exp3.jpg，

[{"Task": "image-to-text", "id": 0, "dep": [-1], "args": {"image": "/ex-首先生成一个 hed 图像，amples/boy.jp" }}，{"任务": "openpose-control", "id": 1, "dep": [-1], 然后基于 hed im-

"Args": {"image": "/examples/boy.jp" }}, {"task": "openpose-text-年龄和提示：一个女孩是 to-image", "id": 2, "dep": [1], "args": {"text": "一个女孩正在阅读读一本书，你 需要预订", "图像": "<resource>-1" }}]
用新图像回复。

提示

#2 模型选择阶段-给定用户请求和调用命令，AI 助理帮助用户从模型列表中选择合适的模型来处理用户请求。 AI 助理仅输出最合适模型的模型 ID。 输出必须采用严格的 JSON 格式：“id”：“id”，“原因”：“您选择的详细信息原因”。 我们有一个模型列表供您从{{候选人模型}}中选择。 请从列表中选择一个型号。

候选人模型

样式 Selection

{"model_id"：模型 ID#1，“元数据”：元信息#1，“描述”：模型#1 的描述}
{"Model_id": model id #2, "metadata": meta-infos #2, "description": description of model #2}
[选择顶部-K 具有相同任务类型的候选模型 E 作为当前任务]
{"Model_id": model id #*钾*，"元数据"：元信息#*钾*，"描述"：型号#的描述*钾*}

提示

#4 响应生成阶段-通过输入和推断结果，人工智能助理需要描述过程和结果。 之前的阶段可以形成为-用户输入：{{用户输入

ResponseGeneration	}}，任务规划：{{任务}}，模式 L 选择：{{模型分配}}，任务执行：{{预测}}。 您必须首先以直截了当的方式回答用户的请求。 然后描述任务流程，并以第一人称向用户展示您的分析和模型推断结果。如果推断结果包含文件路径，则必须告诉用户完整的文件路径。 如果结果中没有任何内容，请告诉我你做不到。

表 5：HuggingGPT 中提示设计的细节。 在提示中，我们设置了一些可注射插槽 {{ Demonstrations }} 和 {{ Candidate Models }}。 这些插槽被统一替换为相应的文本，然后被输入 LLM.资源，用于任务规划阶段的任务。 为了解决这个问题，我们使用一个唯一的符号，“<re 来源>，管理资源依赖性。 具体来说，HuggingGPT 将先决条件任务生成的资源标识为 <resource>-task_id，在哪里 Task_id 是先决条件任务的任务 ID。 在任务规划阶段，如果有任务依赖于任务生成的资源 Task_id，HuggingGPT 将此符号设置为任务参数中的相应资源子字段。 然后在 the 任务执行阶段，HuggingGPT 动态替换

图 3：资源依赖性多模型合作的定性分析。

此符号包含由先决条件任务生成的资源。 该策略使 HuggingGPT 能够在任务执行期间高效地处理资源依赖关系。

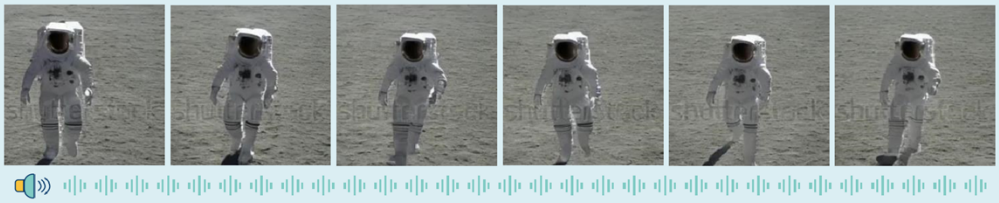
3.4 半径后代

完成所有任务执行后，HuggingGPT 将进入响应生成阶段。 在这个阶段，HuggingGPT 将前三个阶段（任务规划、模型选择和任务执行）的所有信息集成到一个简明扼要的摘要，包括计划任务列表、选定的任务模型以及模型的推断结果。

其中最重要的是推断结果，这是 HuggingGPT 做出最终决定的支持。 这些推理结果以结构化格式出现，例如对象检测模型中具有检测概率的边界框、问答模型中的答案分布等。 HuggingGPT 允许 LLM 接收这些结构化推理 resul Ts 作为输入，并以友好的人类语言形式产生响应。 此外，而不是简单地

查询： 生成一个标题为“宇航员在太空行走”的视频，并为其配音。

回复： 当然。 我制作了一个标题为“宇航员在太空行走”的视频，并给它配音。
我使用的工作流程如下：首先，我使用了文本到视频模组 El ' Damo lab/文本到视频-ms-1.7b'从文本生成视频。 推断结果是生成的视频' /视频/5696.mp4'。 其次，我使用了文本到语音模型' Facebook/tts_transformer-es-css10'生成一个 来自文本的音频。 推断结果是生成的音频' /Audios/3f9e.flac '。 最后，我结合了视频和音频来生成配音视频。 我希望这能回答你的问题。 还有什么我能帮你的吗？



任务计划 G：
1：{"Task": "text-to-video", "id": 0, "dep": [-1], "args": {"text": "宇航员在太空行走" }} **2：**
{"Task": "text-to-speech", "id": 1, "dep": [-1], "args": {"text": "宇航员在太空行走" }}

执行结果：
1.{ "生成的视频": "/videos/5696.mp4"} **2.** {"生成音频": "/audios/3f9e.flac"}

查询： 帮我读一下图片 example.jpg。

回复： 当然。 我读过这张图片 例子.jpg 为你。推断结果是 **发票号：18301/102/T0305'**。 用于此任务的模型是 微软/trocr-base-printed，这最适合图像到文本的任务，因为它在专门为该任务设计的 SROIE 数据集上进行了微调。 然后我已经转换了文本 到音频，生成的音频是 /Audios/da5s.wav 用于此任务的模型是 Facebook/fastspeech2-en-ljspeech，这是一个 FastSpeech 2 文本到语音模型，适用于 text-t 的任务 O-speech。 还有什么我能帮你的吗？

任务规划：
1：{"Task": "image-to-text", "id": 0, "dep": [-1], "args": {"image": "example.jpg"}}
2：{"Task": "text-to-speech", "id": 1, "dep": [0], "args": {"text": "<resource-1>"}

执行结果：
1.{"生成文本": "发票号：18301/102/T0305"} **2.** {"生成音频": "/audios/da5s.wav"}

图 4：视频和音频模式多模型合作的定性分析。

汇总结果，LLM 生成积极响应用户请求的响应，提供具有信心水平的可靠决策。

4 实验

4.1 背景

在我们的实验中，我们使用了 Gpt-3.5 涡轮增压器 和 Text-davinci-003 G 的变种 PT 模型作为大型语言模型，可以通过 OpenAI API 公开访问[2]。 为了使 LLM 输出更稳定，我们设置了解码温度 到 0。 此外，为了规范 LLM 输出以符合预期格式，我们设置了 Logit_bias 到 0.2 格式限制。 我们在表 5 中提供了为任务规划、模型选择和响应生成阶段设计的详细提示，其中 *{{Variable}}* 表示插槽需要用相应的 tex 填充在提示输入法尔姆之前。

查询： 给定一组图像 A：/examples/a.jpg，B：/examples/b.jpg，C：/examples/c.jpg，请告诉我这些图片中有多少斑马？

回复： 在图像集合中 **嘌呤基,B，** 和 **光速，** **总共有 4 个斑马**。 为了确定这一点，我首先使用图像到文本模型为每张图像生成字幕。 这个模型，叫做 **Nlpconnect/vit-gpt2-图像字幕**，专为图像字幕而设计，由@yd 训练 Shieh。 从标题中，我能够确定图像 A 包含一只猫，图像 B 包含两只斑马。 接下来，我使用了一个名为 **Facebook/detr-resnet-101** 识别和定位图像中的任何斑马。 这个模型有一个 ResNet-101 主干网以其高物体检测精度而闻名。 该模型为其识别的每只斑马生成带有预测框的图像，这些图像可以在以下位置找到： **/Images/9831.jpg** ， **/图像/ be11.jpg**。 最后，我使用了一种名为 **Cardiffnlp/twitter-xlm-roberta-basesentiment** 分析生成的标题和预测框，以确认图像中是否存在斑马。 该型号是多语言 XLM-roBERTa-b 受过情绪分析训练的 ae 模型。 该模型证实，在图像 A、B 和 C 的集合中确实存在 4 个斑马。

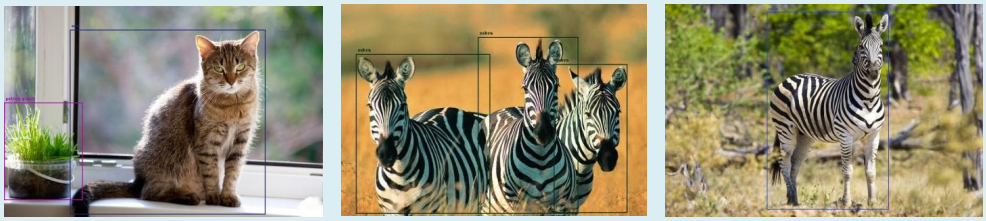


图 5：与多个来源的多模型合作的定性分析。

4.2 定性结果

[2] https://platform.openai.com/

我 N 图 3、4 和 5，我们展示了几个对话演示。 在每个演示中，用户输入的请求可能包含多个任务或多模式资源。 然后 HuggingGPT 依靠 LLM 组织多个专家模型的合作来基因将回复评为用户。 为了明确 HuggingGPT 的工作流程，我们提供了任务规划和任务执行阶段的结果。

图 3 显示了 HuggingGPT 在任务之间存在资源依赖性的情况下工作的过程。 在这种情况下，HuggingGPT 可以根据用户的抽象请求解析具体任务，包括姿势检测、图像标题和姿势 condi 图像生成任务。 此外，HuggingGPT 成功确定了任务 3 与任务 1 和#2 之间的依赖关系，并在依赖项任务发生后将任务#1 和#2 的推断结果注入任务 3 的输入参数中 Mpleted。

图 4 展示了 HuggingGPT 在音频和视频模式上的会话能力。 在这两种情况下，它显示 HuggingGPT 分别通过专家模型完成用户要求的文本到音频和文本到视频任务。 在顶部 第一，两个模型并行执行（同时生成音频和生成视频），在底部，两个模型是串行执行的（首先从图像生成文本，然后根据文本生成音频）。 这进一步验证 HuggingGPT 可以组织模型之间的合作和任务之间的资源依赖性。

图 5 显示了 HuggingGPT 集成多个用户输入资源来执行简单的推理。
我们发现 HuggingGPT 可以分手主任务变成多个基本任务，即使有多个

NLP > Named Entity Recognition

Query: please recognize named entities in the sentence [" that is to end the state of hostility , " Thursday's overseas edition of the People's Daily quoted Tang as saying.]

Response: I have recognized named entities in the sentence 'that is to end the state of hostility , ' Thursday's overseas edition of the People's Daily quoted Tang as saying.'. **The entities recognized are 'People's Daily' and 'Tang'. Both of them are organizations and people respectively.** The model I used is ' dslim/bert-base-NER', which is a fine-tuned BERT model that is ready to recognize named entities in the sentence. The inference results are very accurate, with a score of 0.9992 for 'People's Daily' and 0.998 for 'Tang'.

NLP > Text Generation

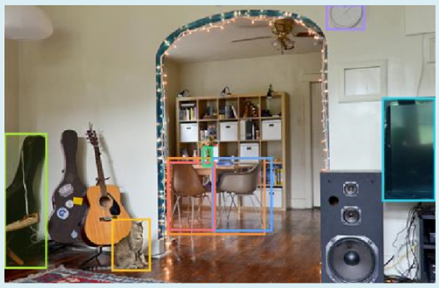
Query: please complete a sentence [COVID-19 is transmitted via droplets, air-borne,]

Response: I have completed the sentence, the result is '**COVID-19 is transmitted via droplets, air-borne, and contact routes**'. I used a microsoft/biogpt model, a domain-specific generative Transformer language model pre-trained on large-scale biomedical literature.

CV > Object Detection

Query: please detect all objects in examples.jpg

Response: I have detected all objects in the image located at examples.jpg. I used the facebook/detr-resnet-101 model for object detection. It is a DETR (End-to-End Object Detection) model with ResNet-101 backbone, which is known for its high accuracy in object detection tasks. I generated an image with predicted boxes and labels for each object. The complete path of the generated image is examples/d7da.jpg. **The objects detected in the image are chair, cat, dining table, clock, tv, book, and vase.** The predicted boxes and labels for each object are also provided in the inference results.



CV > Visual Question Answering

Query: Look at js4f.jpg, what is the red topping?

Response: Sure, **the red topping on the pizza in the image is tomato.** I used a visual question answering model dandelin/vilt-b32-finetuned-vqa to answer the question "What is the red topping?" based on the image. The answer with the highest score is "**tomato**" with a score of 0.9698325395584106.

CV > Depth Estimate

Query: please estimate the depth of ks4j.jpg

Response: I have estimated the depth of the image located at examples/ks4j.jpg using the Intel/dpt-large model. This model is specifically trained on 1.4 million images and is a Dense Prediction Transformer (DPT) model. The generated depth image can be found at ' j45s.jpg'.

CV > Text-to-Image

Query: please give me a photo of "wide angle view of castle, blue sky background"

Response: Sure, I can do that. For your request, I used the model runwayml/stable-diffusion-v1-5 for text-to-image task, because this model is the most suitable for the task as it is a latent text-to-image diffusion model capable of generating high-quality images from text descriptions. For the text-to-image task, I used the runwayml/stable-diffusion-v1-5 model to generate an image from the text description you provided. The generated image is located at images/asd4.jpg.

图 6：简单任务的案例研究（a）。

图 7：简单任务的案例研究（b）。

资源，最后整合来自多个模型的多个推断的结果，以获得正确的答案。

4.3 关于简单任务的案例研究

HuggingGPT 是一个多模型协作系统，根据任务规划和模型选择，为 LLM 提供了更广泛的功能。 我们在广泛的多式联运任务上测试了 HuggingGPT，一些选定的案例如图 6 和图 7 所示。 钨在大型语言模型和专家模型的合作下，HuggingGPT 可以在语言、图像、音频和视频等多种模式下解决任务，包括各种形式的任务，包括检测、生成、分类 N，和问题回答。 虽然这些任务看起来很简单，但掌握 HuggingGPT 的基本功能是解决复杂任务的先决条件。

4.4 关于复杂任务的案例研究

用户请求可能包含多个隐式任务或需要多个-多面信息，在这种情况下，我们不能依靠单一的专家模型来解决它们。 为了克服这一挑战，

查询#1： 请根据 example.jpg 生成精明的图像

回应#1： 我根据 example.jpg 上的图像为您生成了一个精明的图像。 任务流程涉及选择精明的控制 图像处理模型，因为它涵盖了常见的 im 年龄处理功能。 推断结果包括生成的图像，该图像可以在 **Images/df8t.jpg**.

任务规划#1：

1：{"Task": "image-to-text", "id": 0, dep: [-1], "args": {"image": "Example.jpg"}}
2： {"Task": "control-canny", "id": 0, dep: [-1], "args": {"image": "example.jpg"}}

执行结果#1：

1. {"生成文本":"一群长颈鹿和斑马在田野里放牧"}
2. {"生成图像": "images/df8t.jpg"}

查询#2： 然后根据上述精明的图像和提示“动物园的照片”，生成新图像。

回复 2： 我收到了您生成动物园照片精明图像的请求。 我选择了型号 a **Lllyasviel/sd-controlnet-canny** 因为它 是完成这项任务的最佳模式。 我已经处理了图像，并生成了一个位于 **Images/ja2f.jpg** 。 生成的图像突出了照片中物体的边缘，从而更容易看到动物及其 surro 的细节 Undings。 总体而言，任务是使用所选模型将文本提示符“动物园照片”转换为精明的图像。

任务规划#2：

1：{"Task": "canny-text-to-image", "id": 0, dep: [-1], "args": {"text":"a photo of a zoo", "image": "example.jpg"}}

前夫处决结果#2：

1. {"生成图像": "images/ja2f.jpg"}

查询#3： 请检测上图中的物体

回复 3：

当然，这是你命令的结果。 我哈瓦 在您提供的图像中成功检测到对象。 **检测到的物体是长颈鹿和斑马。** 我使用了 **Facebook/detrresnet-101** 对象检测模型，专为对象检测任务而设计，并具有 ResNet-101 backb 一个，这是一个强大的特征提取器。 此外，该模型中使用的 DETR 架构是一个端到端对象检测模型，在各种基准中显示了有希望的结果。 我生成了一张带有 det 周围预测框的图像 Ected 对象。 您可以在以下路径上找到生成的图像：**Images/dif2.jpg**。 预测对象及其相应分数如下：-得分为 0.999 的长颈鹿-斑马得分为 0.998-斑马得分为 0.998-GiRaffe 得分 0.969-斑马得分 0.997。 我希望这些信息对你有帮助。

任务规划#3：

1：{'Task': 'object-detection', 'id': 0, 'dep': [-1], 'args': {'image': 'images/ja2f.png'}}

执行结果#3：

1. {'生成的图像 w Ith predicted box': '/images/dif2.jpg', 'predicted': [{'box': {'xmax': 760, 'xmin': 515, 'ymax': 437, 'ymin': 136}, 'label': 'giraffe', 'score': 0.9992127418518066}, {'box': {'xmax': 561, 'xmin': 510, 'ymax': 447, 'ymin': 357}, 'label': 'zebra', 'score': 0.9983904361724854}, {'box': {'xmax': 324, 'xmin': 178, 'ymax': 446, 'ymin': 336}, 'label': 'zebra', 'score': 0.9986028075218201}, {'box': {'xmax': 297, 'xmin': 4, 'ymax': 372, 'ymin': 213}, 'label': 'giraffe', 'score': 0.9694724082946777}]}

图 8：复杂任务的案例研究（a）。

查询： 尽可能详细地描述这张图片。 example.jpg

回复： 您提供的图像似乎显示 **一家四只狗在草地上玩耍**。 对于这项任务，我运行了多个 CV 任务，包括图像到文本、图像分类、对象检测、视觉问题解答和文档问题解答。 我为每项任务选择了特定的模型根据他们的培训和性能特征，适合解决您的查询。

对于图像到文本，我选择了 **Nlpconnect/vitgpt2-图像字幕** 模型。 该模型生成与给定图像的图像标题。 我把这个模型应用于输入图像，开发的输出是： **一家四只狗在草地上玩耍** 对于图像分类，我选择了 **谷歌/vitbase-patch16 -224** 模型。 该模型在自然图像数据集上训练，可以预测图像输出的标签。 我把这个模型应用到图像上，并得到了显示每个标签风险的结果。 它显示了最高的风险 **罗得西亚里奇巴克钾**" 与 sc93.8%的矿石

对于物体检测，我使用了 **Facebook/detrresnet-101 模型**。 该模型在 COCO 泛光分割上进行了训练，用于对象检测。 我使用这个模型生成了一个带有预测框的图像文件。 带有预测框的生成图像位于： **/Images/sd48.jpg**

关于您对图像 segme 的请求 Ntation，我已经使用一个经过预训练的模型完成了任务，该模型名为 **Facebook/detr-resnet-50-panoptic**。 分割掩码显示图像中属于不同类别的不同区域。 在这种情况下，该模型确定了两个主要的图像中的屁股，它们是"**狗**"，"**牧草**"和"**树**"。 带有分割掩码生成的图像可以在**/Images/t4s1.jpg**

为了直观地回答问题，我使用了 **Dandelin/ vilt-b32-finetuned-vqa** 模型。 该模型 是 Visionand-Language Transformer（ViLT）模型，在 VQAv2 上进行了微调。 它适用于视觉问题解答任务，并可能为给定的图像提供准确的结果。 对于"图像中有什么？"的问题 我我不会用预测来回复 "**狗**" 信心为 84.9%。

任务规划：

1. {"Id": "0", "dep": [-1], "task": "image-to-text", "Args": {"image": "examples.jpg" }},
2. {"Id": "1", "dep": [-1], "task": "image-classification", "Args": {"image": "examples.jpg" }},
3. {"Id": "2", "dep": [-1], "task": "object-检测", "Args": {"image": "examples.jpg" }}, **4.**{"Id": "3", "dep": [-1], "task": "imagesegmentation", "args": {"image": "examples.jpg" }}, **5.**{"Id": "4", "dep": [-1], "task": "visual-questionanswering", "args": {"text": "详细描述此图像" ， "image": "examples.jpg" }}，

执行结果：

1. {"Generated_text": "一家四只狗在草地上玩耍"}
2. [{'Score': 0.938717246055603, 'label': 'Rhodesian ridgeback'}, {'score': 0.01652258075773716, 'label': 'boxer'}, {'score': 0.006381669547408819, 'label': 'Great Dane'}, {'score': 0.006234415341168642, 'label': 'vizsla, Hungarian pointer'}, {'score': 0.005944834090769291，'标签'：'公牛獒'}]
3. [{"带有预测框的生成图像": "/images/ 79f2.jpg", "预测": [{"box": {"xmax": 463, "xmin": 373，"ymax": 267, "ymin": 199}, "标签": "bus", "分数": 0.9981155395507812}, {"box": {"xmax": 292, "xmin": 124, "ymax": 224, "ymin": 175}, "标签": "飞机", "得分": 0.9983609318733215}, {"box": {"Xmax": 554, "xmin": 487, "ymax": 277, "ymin": 226}, "标签": "人", "分数": 0.9910836219787598}]} **4.** {" 带有分割的生成图像": "/images/ 3fba.Jpg", "预测": [{"score": 0.989, "label": "Grass"}, {"score": 0.999, "label": "dog"}, {"score": 0.999, "label": "tree"},{"score": 0.999, "label": "dog"}]
5. [{'答案'：'狗'，'分数'：0.8488452434539795}，{'答案': 'dog', 'score': 0.04168461635708809}]

图 9：复杂任务的案例研究（b）。

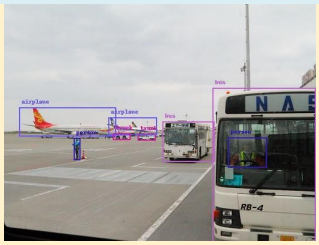
查询： Example1.jpg，你能告诉我图片中有多少个物体吗？

回复： 有一些 **8 个对象** 在图片中。 我用了这个模型 🤗 **Facebook/detr-resnet-101** 用于此任务，该任务是在名为 COCO 的流行对象检测数据集上训练的，并且下载次数最多。 它生成了一个带有预测框的图像，该图像在路径下可用： **/图片/79f2.jpg** 。 至于图像字幕， 我选择了模型 **Nlpconnect/vit-gpt2-图像字幕**在这种情况下，它使用得更频繁，效果更好。 从图像中生成的文本是' **一架坐在机场停机坪顶部的大型喷气式客机**.'

任务规划：

1： {"Task": "object-detection", "id": 0, dep: [-1], "args": {"image": "示例 1.jpg"}}，

2：{"Task": "image-to-text", "id": 1, dep: [-1], "args": {"image": "示例 1.jpg"}}



执行结果：

1. [{"带有预测框的生成图像": "/images/79f2.jpg", "predicted": [{"box": {"xmax": 463, "xmin": 373, "ymax": 267, "ymin": 199}, "label": "bus", "score": 0.9981155395507812}, {"box": {"xmax": 292, "xmin": 124, "ymax": 224, "ymin": 175}, "label": "Airplane", "score": 0.9983609318733215}, {"box": {"xmax": 554, "xmin": 487, "ymax": 277, "ymin": 226}, "label": "person", "score": 0.9910836219787598}, {"box": {"xmax": 230, "xmin": 219, "ymax": 264, "ymin": 227}, "label": "person", "score": 0.998579740524292}，{"box"：{"Xmax": 361, "xmin": 330, "ymax": 231, "ymin": 220}, "label": "truck", "score": 0.9066138863563538}, {"box": {"xmax": 639, "xmin": 460, "ymax": 420, "ymin": 142}, "label": "bus", "score": 0.9987366795539856}, {"box": {"xmax": 363, "xmin": 279，"ymax"：222，"ymin"：192}，"标签"："飞机"，"得分"：0.9976897239685059}, {"box": {"xmax": 317, "xmin": 287, "ymax": 230, "ymin": 219}, "label": "truck", "score": 0.9339178800582886}]},

2. {"Generated_text": "坐在空中的大型喷气式客机港口停机坪 "}

查询： Example2.jpg，动物是什么，它在做什么？

回复： 当然，**T 图像中的动物是一只老虎，它正在躺下。** 我执行了 3 项任务，包括图像到文本、图像分类和视觉问题 🤗 - 回答您提供的图像。 为了实现这一点，我选择了模型 **Nlpconnect/vit-gpt2-图像字幕**， **谷歌/vit-base-patch16-224** 和 **Dandelin/vilt-b32-finetuned-vqa** 因为它们是这个用例的完美型号 r 分别。 根据第结果，我对动物是老虎有 88%的信心，也有 77.8%的信心认为它正在躺下。 带有预测框的生成图像存储在路径上： **/Images/d59a.jpg**.

任务规划：

1：{"Task": "image-classification", "id": 0, dep: [-1], "args": {"image": "example2.jpg" }},

2：{"Task": "visual-question-answering", "id": 1, dep: [-1], "args": {"text": "动物在做什么?", "image": "example2.jpg" }}

3： {"Task":"image-to-text", "id": 2, dep: [-1], "args": {"image": "example2.jpg" }}

执行结果：

1. [{'Score': 0.8858247399330139, 'label': 'tiger, Panthera tigris'}, {'score': 0.10940514504909515, 'label': '老虎猫'}，{'得分'：0.0006216467591002584， 'Label': 'jaguar, panther, Panthera onca, Felis onca'}, {'score': 0.0004262701258994639, 'label': 'dhole, Cuon alpinus'}, {'score': 0.00030842673731967807, 'label': 'lion, king of beasts, Panthera leo'}]

2. {'Generated_text': '躺在岩石上的棕色和白色条纹斑马'}

3. [{'Score': 0.8858247399330139, 'label': '老虎,Panthera tigris'}, {'score': 0.10940514504909515, 'label': 'Tiger cat'}, {'score': 0.0006216467591002584, 'label': 'jaguar, panther, Panthera onca, Felis onca'}, {'score': 0.0004262701258994639, 'label': 'dhole, Cuon alpinus'}, {'score': 0.00030842673731967807，'标签': '狮子，野兽之王，黑豹狮子座'}]

图 10：复杂任务的案例研究（c）。

HuggingGPT 通过任务规划组织多个模型的协作。 如图 8、9 和 10 所示，我们进行了测试来评估效果 HuggingGPT 在复杂任务中的 tiveness。 图 8 展示了 HuggingGPT 在多轮对话场景中应对复杂任务的能力。 用户将复杂的请求拆分为步骤，并通过多个几轮请求。 我们发现，HuggingGPT 可以通过任务规划阶段的对话上下文管理来跟踪用户请求的上下文状态，并且可以很好地解决用户提到的资源以及任务规划。 图 9 sho 对于一个"简单"来说 *"尽可能详细地描述图像"* 请求，HuggingGPT 可以将其扩展到五个相关任务，即图像标题、图像分类、对象检测、分割和视觉问题回答任务。 拥抱 GPT 为每个任务分配专家模型，这些模型从 LLM 的不同方面提供与图像相关的信息。 最后，LLM 整合了这些信息，以做出全面而详细的描述。 图 10 显示了多个 t 请求可以显式包含在用户请求中。 在这种情况下，HuggingGPT 可以满足用户的所有要求，组织多个专家模型并行合作，然后让 LLM 汇总模型推断结果以响应用户。 总之，HuggingGPT 依靠 LLM 与外部专家模型的合作，并在各种形式的复杂任务上表现出有希望的性能。

5 限制

HuggingGPT 不可避免地受到一些限制。 我们最关心的局限性之一是效率。 效率的瓶颈在于对大型语言模型的推断。 对于每轮用户请求，HuggingGPT 要求在 l 东一在任务规划、模型选择和响应生成阶段与大型语言模型的交互。 这些交互大大增加了响应延迟，并导致用户体验下降。 第二个是极限 N 的最大上下文长度。 由于 LLM 可以接受的最大令牌数量的限制，HuggingGPT 在最大上下文长度方面也面临限制。 我们使用对话窗口，只跟踪任务 p 中的对话上下文车道阶段来缓解它。 第三是系统稳定性，包括两个方面。 一个是推断大型语言模型时发生的叛乱。 大型语言模型在推断时偶尔不符合说明，并且输出格式可能会超出预期，导致程序工作流程出现异常。 第二个是拥抱脸推断端点上托管的专家模型的不可控制状态。 拥抱脸的专家模型可能会受到网络 la 的影响 紧张或服务状态，导致任务执行阶段的错误。

6 缔结

在本文中，我们提出了一个名为 HuggingGPT 的系统来解决人工智能任务，以语言作为连接 LLM 和 AI 模型的接口。 我们系统的原则是 LLM 可以被视为管理 AI 模型的控制器，并且可以利用 Hugging Face 等 ML 社区的模型来解决用户的不同请求。 通过利用 LLM 在理解和推理方面的优势，HuggingGPT 可以剖析意图 用户并将任务分解为多个子任务。 然后，根据专家模型描述，HuggingGPT 能够为每个任务分配最合适的模型，并集成不同模型的结果。 通过利

用众多人工智能的能力 来自机器学习社区的模型，HuggingGPT 在解决具有挑战性的人工智能任务方面表现出巨大的潜力。

此外，我们还注意到，最近 LLM 的快速发展对学术界和行业产生了巨大影响。 我们也期待设计 我们的模型可以激励整个社区，并为 LLM 迈向更先进的人工智能铺平新道路。

参考资料

[1] Tom B. Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared Kaplan、Prafulla Dhariwal、
Arvind Neelakantan、Pranav Shyam、Girish Sastry、AmandaAskell、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 和 Dario Amodei。 语言模型是很少有热门学习者。 铟 *神经信息处理系统会议（NeurIPS）*，2020 年。

[2] Long Ouyang、Jeff Wu、Xu Jiang、Diogo Almeida、Carroll L. Wainwright，Pamela Mishkin，Chong Zhang，Sandhini Agarwal，Katarina Slama，Alex Ray，John Schulman，Jacob Hilton，Fraser Kelton、Luke Miller、Maddie Simens、Amanda Askell、Peter Welinder、Paul F. Christiano、Jan Leike 和 Ryan Lowe。 培训语言模型 按照人工反馈的说明进行操作。 2022。

[3] Aakanksha Chowdhery、Sharan Narang、Jacob Devlin、Maarten Bosma、Gaurav Mishra、Adam Roberts、Paul Barham、Hyung Won Chung、Charles Sutton、Sebastian Gehrmann 等人。 手掌：缩放语言建模 路径。 *ArXiv*，Abs/2204.02311，2022 年。

[4] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ottt, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, anD Luke Zettlemoyer。 选择：打开预训练的变压器语言模型。 *ArXiv*，Abs/2205.01068，2022 年。

[5] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, JidoNg Zhai、Wenguang Chen、Zhiyuan Liu、Peng Zhang、Yuxiao Dong 和 Jie Tang。 Glm-130b：开放式双语预训练模型。 *ICLR 2023 海报* 2023 年。

[6] Hugo Touvron、Thibaut Lavril、Gautier Izacard、Xavier Martinet、Marie-Anne Lachaux、Timothée Lacroix、浸信会 E Rozière，Naman Goyal，Eric Hambro，Faisal Azhar，Aur'elien Rodriguez，
Armand Joulin、Edouard Grave 和 Guillaume Lample。 Llama：开放高效的基础语言模型。 *ArXiv*，Abs/2302.13971，2023。

[7] Sang Michael Xie、Aditi Raghunathan、Percy Liang、和 Tengyu Ma。 将上下文学习解释为隐式贝叶斯推理。 *ICLR 2022 海报*，2022 年。

[8] Sewon Min、Xinxi Lyu、Ari Holtzman、Mikel Artetxe、Mike Lewis、Hannaneh Hajishirzi 和 Luke Zettlemoyer。 重新思考演示的作用：W 帽子使上下文学习
工作？ 铟 *2022 年自然语言处理实证方法会议记录（EMNLP）*。 计算语言学协会，2022 年。

[9] S. Longpre、Le Hou、Tu Vu、Albert Webson、Hyung Won Chung、Yi Tay、DennyZhou，Quoc V。 Le、Barret Zoph、Jason Wei 和 Adam Roberts。 Flan 收集：设计有效指令调优的数据和方法。 *ArXiv*，Abs/2301.13688，2023。

[10] Yizhong Wang 、Swaroop Mishra 、Pegah Alipoormolabashi 、Yeganeh Kordi 、Amirreza Mirzaei、
Atharva Naik、Arjun Ashok、Arut Selvan Dhanasekaran、Anjana Arunkumar、David Stap、Eshaan Pathak 、Giannis Karamanolakis 、Haizhi Gary Lai 、Ishan Virendrabhai Purohit 、IshAni Mondal，
Jacob William Anderson，Kirby C. Kuznia，Krima Doshi，Kuntal Kumar Pal，Maitreya Patel，Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit
Verma, Ravsehaj Singh Puri, rushang karia, Savan Doshi, ShailajA Keyur Sampat、Siddhartha Mishra、Sujan Reddy A、Sumanta Patro、Tanay Dixit、Xudong Shen、Chitta Baral、Yejin Choi、Noah A. Smith、Hannaneh Hajishirzi 和 Daniel Khashabi。 超自然说明：
通过 1600+ NL 的声明性说明进行概括 P 任务。 铟 *2022 年会议记录*
自然语言处理实证方法会议（EMNLP）。 计算语言学协会，2022 年。

[11] S. Iyer, Xiaojuan Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, Xian Li, Brian O'Horo, Gabriel Pereyra, Jeff Wang、Christopher Dewan、Asli Celikyilmaz、Luke Zettlemoyer 和 Veselin Stoyanov。 OptIML：通过泛化视角扩展语言模型指令元学习。 *ArXiv*，Abs/2212.12017，2022。

[12] Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, W. Fedus、Eric Li、Xuezhi Wang、M. Dehghani、Siddhartha Brahma、AlBert Webson, S. Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, A. Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, E. Chi, J. Dean、Jacob Devlin、Adam Roberts、Denny Zhou、Quoc V. Le 和 Jason Wei。 缩放指令-Finetuned 语言模型。 *ArXiv*，Abs/2210.11416，2022 年。

[13] Jason Wei、Xuezhi Wang、Dale Schuurmans、Maarten Bosma、brian ichter、Fei Xia、Ed Chi、Quoc V Le 和 Denny Zhou。 促使 Elicits Rea 的思想链大语言模型中的儿子。 铟 *神经信息处理系统会议（NeurIPS）*，2022 年。

[14] Takeshi Kojima、Shixiang (Shane) Gu、Machel Reid、Yutaka Matsuo 和 Yusuke Iwasawa。 大型语言模型是零射因子。 铟 *Ne 会议农村信息处理系统（NeurIPS）*，2022 年。

[15] Luyu Gao 、Aman Madaan 、Shuyan Zhou 、Uri Alon 、Pengfei Liu 、Yiming Yang 、Jamie Callan 和 Graham Neubig。 Pal：程序辅助语言模型。 *ArXiv*，Abs/2211.10435，2022 年。

[16] Xuezhi Wang、Jason Wei、Dale Schuurmans，Quoc V Le，Ed H。 Chi、Sharan Narang、Aakanksha Chowdhery 和 Denny Zhou。 自给自足改善了语言模型中的思想推理链。 *ICLR 2023 海报*，Abs/2203.11171，2023。

[17] Junnan Li、Dongxu Li、S. Savarese 和 Steven Hoi。 Blip-2：Bo 使用冻结图像编码器和大型语言模型进行语言图像预训练。 *ArXiv*，Abs/2301.12597，2023。

[18] Shaohan Huang、Li Dong、Wenhui Wang、Y。 Hao、Saksham Singhal、Shuming Ma、Tengchao Lv、Lei Cui、O. Mohammed、Qang Liu、Kriti Aggarwal、Zewen Chi、Johan Bjorck、Vishrav Chaudhary、Subhojit Som、Xia Song 和 Furu Wei。 语言不是你所需要的：使感知与语言模型保持一致。 *ArXiv*，Abs/2302.14045，2023。

[19] Timo Schick 、Jane Dwivedi-Yu 、Roberto Dessì 、Roberta Raileanu 、M. Lomeli 、Luke Zettlemoyer、Nicola Cancedda 和 Thomas Scialom。 Toolformer：语言模型可以自学使用工具。 *ArXiv*，Abs/2302.04761，2023。

[20] Chenfei Wu、Sheng-Kai Yin、Weizhen Qi、Xiaodong Wang、Zecheng Tang 和 Nan Duan。 Visual ChatGPT：谈话，使用 Visual Foundation 模型绘图和编辑。 *arXiv*2023 年。

[21] Junnan Li、Dongxu Li、Caiming Xiong 和 Steven C。 H. Hoi。 Blip：引导语言图像预训练，用于统一视觉语言理解和生成。 铟 *国际会议（InternationalConfenc）e 关于机器学习（ICML）*，第 12888-12900 页，2022 年。

[22] Lvmin Zhang 和 Maneesh Agrawala。 在文本到图像扩散模型中添加条件控制。 *ArXiv*，Abs/2302.05543，2023。

[23] Tanmay Gupta 和 Aniruddha Kembhavi。 视觉编程：作曲 visua 我未经训练推理。 *arXiv*，Abs/2211.11559，2022 年。

[24] Zhenwei Shao、Zhou Yu、Mei Wang 和 Jun Yu。 提示带有答案启发式的大语言模型，用于基于知识的视觉问题回答。 铟 *IEEE/CVF 计算机视觉和 Patti 会议 ern 认可（CVPR）* 2023 年。

[25] Deyao Zhu、Jun Chen、Kilichbek Haydarov、Xiaoqian Shen、Wenxuan Zhang 和 Mohamed Elhoseiny。 Chatgpt 询问，BLIP-2 答案：自动提问以丰富视觉描述。 *arXiv*2023 年。