arXiv:2302.03154v1 [cs.HC] 6 Feb 2023

对话回归测试：一种为预训练语言模型原型设计可泛化提示策略的设计技术

J.D. Zamfirescu-Pereira 加州大学伯克利分校 美国加利福尼亚州伯克利
zamfi@berkeley.edu

Bjoern Hartmann 加州大学伯克利分校 美国加利福尼亚州伯克利
bjoern@eecs.berkeley.edu

Qian Yang 康奈尔大学 美国纽约州伊萨卡
qianyang@cornell.ed

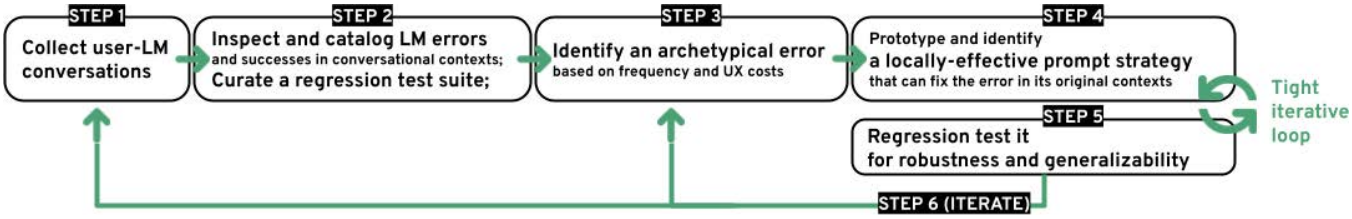

图 1：我们提出了对话回归测试，这是一种聊天机器人设计师可用于系统地试验并评估各种提示策略对由预训练语言模型驱动的对话交互的影响的工作流程。我们还介绍了 BotDesigner，这是一款将该工作流程付诸实践的提示原型设计工具。

摘要

预训练语言模型（LM），例如 GPT-3，无需额外配置即可进行流畅的多轮对话，因此是设计聊天机器人的理想素材。此外，设计者可以在语言模型的输入前添加文本提示词 ——即关于期望交互的指令和示例——以改进聊天机器人的话语。然而，基于提示的改进可能并不稳健；设计者难以系统地了解某种提示策略会如何影响不同用户后续对话的展开。为应对这一挑战，我们提出了对话回归测试这一概念。对话回归测试基于与基线聊天机器人的样例对话，追踪应用不同提示策略后，对话中的错误是持续存在还是得到解决。我们将这一技术融入交互式设计工具*BotDesigner*，该工具让设计者能够识别多个对话中的典型错误；通过图可视化呈现对话中的共性脉络；并突出显示提示变更在机器人设计迭代中的影响。初步评估表明，回归测试这一概念及 BotDesigner 的各项功能对聊天机器人设计师均有帮助。

1 引言

预训练大型语言模型（语言模型）与提示词的结合，为聊天机器人设计带来了令人振奋的新机遇。近期的预训练语言模型（ GPT-3 [9], GPT-J [39], Jurassic-1 [24], 和 T0 [35]）开箱即用，便能进行流畅的多轮对话。由于无需承担训练监督模型所需的成本，也不再受数据要求的限制，这些模型大幅降低了打造尚可的对话式用户体验的门槛

（用户体验） [7]。此外，聊天机器人设计师可以通过在语言模型输入前添加提示词来改进语言模型输出——这些提示词是文本形式的指令，以及设计者期望的交互示例（表1）。提示词直接引导模型生成预期输出，让人们能够以很少甚至无需标注数据，实现更高水平的对话式用户体验。过去两年间，这一新预训练与提示范式的前景推动了人机交互/用户界面软件与技术研讨会（例如， [18, 23, 26, 40, 41]）和自然语言处理（例如， [22, 25, 31, 35, 36]）领域中一系列提示设计工具 [3, 17, 27], 研究的迅速发展，甚至催生了面向终端用户的应用 [18]。一些学者甚至认为，这一范式代表着自然语言处理领域的一场“巨变”，并将“取代”全监督范式和预训练与微调范式 [25]。

提示策略	生成的对话
无提示（基线）	🗣️ 好，稍等一下，我去拿把椅子。 🤖 往椅子前沿挪一挪*[...]*
明确的等待指令 等待	🤷 如果用户要求你等待，请解释这不是问题 *[...]* 🗣️ 好的，稍等，我去拿把椅子。 🤖 你拿到椅子后，挪到椅子前面*[...]*

表1：设计者如何通过修改提示策略（）直接改善聊天机器人交互的示例。注意机器人（）对用户（）所说内容的回应变化。

预训练与提示范式也给聊天机器人设计带来了一系列新挑战，即如何设计稳健的提示策略。 为了利用提示词构建生产级聊天机器人，设计者首先需要系统地评估：（1）在哪些

对话场景中，预训练语言模型可能会失效；（2）每种失效或失效模式的发生频率有多高、危害有多大，从而确定应通过提示解决哪些问题。接下来，设计者需要（3）找出一种提示策略，能够在目标失效原本出现的对话语境中加以修复；最后，还要系统地评估其泛化能力和鲁棒性，也就是说，评估（4）它是否能修复其他对话场景中的类似失效，以及在不同用户后续可能展开的众多对话路径中，是否会引发新的错误。这些任务颇具挑战性 [7, 25, 35]。

一些人机交互研究人员已开始创建有助于提示策略设计的工作流程和工具，例如用于人机协作写作 [40]。然而，面向聊天机器人的此类工作流程和工具极为罕见。相反，聊天机器人设计师常常使用 GPT-3 Playground [30] 等工具即兴尝试提示词；有些人甚至将提示策略设计视为“碰运气” [41]。设计者如何全面分析语言模型在不同对话中犯下的高度依赖语境的错误（挑战 1、2），以及如何在解决这些错误时避免无意中在此前或后续对话中引发新错误（挑战 3、4），目前仍不清楚。

为推动提示策略原型设计变得更加系统、严谨，我们提出了对话回归测试这一概念。受软件回归测试启发，对话回归测试将基线语言模型曾经失败（或表现显著成功）的对话场景作为可复用的测试用例，并帮助设计者追踪提示策略更新对这些测试用例产生的影响。这种方法使设计者能够自由尝试多种提示策略，以解决特定语境中的错误，同时确保系统整体稳定，并持续改进。

为了将这一概念付诸实践，我们接下来介绍 BotDesigner——一款提示策略原型设计工具，它将对话回归测试工作流程整合到交互式机器学习分析工具中（该工具会追踪模型在多次迭代中的性能，并深入分析哪些改动带来了哪些性能提升）。此类工具在非对话领域的设计者和开发者中已广受欢迎（例如 Weights and Biases [5]）。BotDesigner 由四个组件构成：

- **对话收集器**，用于收集基线语言模型驱动的聊天机器人与真实用户（或众包工作者）之间示例对话的界面；
- **标注器**，用于检查和归类基线机器人在与多位用户进行的多轮对话中出现的问题话语（或成功话语）的界面。这些错误为提示词提供了改进机会，同时也构成对话回归测试的测试用例；
- **可视化器**，一种图形化可视化工具，可通过展示基线机器人在常见终端用户与语言模型对话模式背景下的失败与成功，帮助设计者识别典型错误。这些典型错误有助于设计者确定提示设计工作的优先顺序；
- **回归测试器**，体现了对话回归测试的功能。当设计者尝试新的提示策略时，这些功能使他们能够跟踪目标错误是否仍然存在或已得到解决，或者是否因新策略而出现了新的错误。

本文介绍了对话回归测试的概念、BotDesigner 的实现，以及一项小规模用户评估研究。该研究初步表明，这两者对聊天机器人设计师设计教学聊天机器人都很有用。

本文作出了两项贡献，一项是概念层面的，另一项是技术层面的。主要贡献是提出了用于提示策略设计的对话回归测试概念。以往大多数研究都聚焦于提示设计的探索与临时测试阶段，而对话回归测试则提供了一个初步工作流程，用于评估提示策略的鲁棒性和可泛化性。其次，本文的技术贡献在于提出了实现 BotDesigner 的技术。本文介绍了一种新颖的对话可视化技术，可展示语言模型与不同用户之间大量独立对话中常见的对话模式。这项技术可用于开发许多其他人机交互分析或设计工具。BotDesigner 还实现了对话回归测试的界面；这项技术有助于为对话交互之外的许多其他语言模型应用制作提示词原型。

2 相关工作

我们简要回顾三类相关工作：1）用于交互式提升自然语言处理模型性能的工作流和工具；2）用于改善对话式用户体验的工作；最后是 3）既有的对话可视化技术和分析工具。

2.1 自然语言处理建模工作流和工具

自然语言处理建模工作流和工具大致分为三类 [25]。全监督学习，即使用任务输入—输出示例数据集训练特定任务模型，长期以来一直是机器学习（ML）和自然语言处理（NLP）的核心方法。由于完整标注的数据集往往不足以训练出高质量模型，用于提升模型性能的交互式自然语言处理工具，主要致力于辅助特征工程；也就是说，为模型提供适当的归纳偏置，使其能够从有限数据中学习。为实现这一目标，监督式自然语言处理工具通常采用以下两种工作流程之一：

- LightSIDE 等工具 [28] 帮助自然语言处理建模人员从原始数据中定义并提取显著特征。这些工具采用了五步工作流程，许多开创性的交互式机器学习工具（例如 Crayons [16], ModelTracker [2], Gestalt [32], 和 Weights and Biases [6]）都率先使用了这一流程：建模人员 (i) 检查原始数据；(ii) 标注数据或从数据中提取特征，有时会借助机器学习；(iii) 训练初始模型；(iv) 对模型的输出进行分类、查看和修正；(v) 反复迭代这一流程，同时由工具跟踪模型性能的提升，并提供洞见，说明哪些改动带来了提升。

- 第二种工作流程是为了回应这样的批评而出现的：上述工作流程忽略了对机器学习业余爱好者的考量[1]。研究人员开发了“以人为中心的机器学习工具”，将最终用户纳入第一种工作流程的每个步骤（例如，允许他们提供与模型自然交互的记录，用于模型训练 [43] 和迁移学习 [29], 指定特征

[11], 展示期望的模型行为 [42],等）。这些工具表明，将对最终用户的理解以及他们的自然交互数据融入机器学习工作流，能够同时改善用户体验和模型性能 [1]。

2017—2019年，自然语言处理建模的标准方式转向了“预训练并微调”，全监督学习所发挥的作用日益缩小 [25]。这一范式采用仅包含两个步骤、不再针对特定任务的机器学习工作流。

步骤 i 建模人员使用固定架构，在大量未标注文本数据上预训练模型。在此过程中，预训练语言模型学习通用语言特征，可用于广泛的任务（例如预测下一行代码或散文、文档摘要、生物医学问答、翻译等）。GPT [9, 39] 和 BERT [13] 是预训练语言模型系列的代表。

步骤 ii 建模人员通过微调，使预训练语言模型适应当前特定的交互任务。

在这一范式中，模型调优的重点从特征工程转向目标工程，即为预训练和微调设计训练目标。因此，前文提到的大多数交互式机器学习工具已不再适用。虽然少数通用商业机器学习工具（例如 Azure [45]）能够支持这一新工作流程，但在文献检索中，我们没有发现专为这一工作流程定制的交互式自然语言处理工具。

过去两年间，自然语言处理领域见证了另一场范式转变：“预训练、提示和预测”范式的兴起 [25]。这一范式大致遵循上述两步工作流程。不过，建模人员并非通过目标工程让预训练语言模型适应特定任务，而是借助文本提示，将任务重新表述为更接近语言模型最初训练时所解决的任务。例如，GPT-3 可以使用提示模板/策略“网页代码描述： <自然语言请求>html：<html> css： < css> javascript：<js>”，自动将用户的自然语言请求转换为 HTML 代码 [18]。建模人员使用模板整理出大量此类提示词，并用这些提示词重新训练语言模型 [3, 14]。

在这一范式中，模型调优的重点转向提示工程，即设计恰当的提示词和提示策略，以产生预期的模型行为。此外，由于许多提示词是人类可读的，它们也为让最终用户参与建模过程带来了新的机会。如今已有工具，使众包工作者或最终用户能够贡献查询和提示策略 [3, 14]。

值得注意的是，即使对于专家而言，要找到稳健且可泛化的提示策略，也需要大量试错：建模人员需要反复尝试各种提示策略，并评估它们对具体输入输出对的影响，然后再通过大型对话数据集对其进行更系统的评估。目前尚未形成成熟的提示设计工作流程。提示或提示策略如何直接影响模型输出，或在重新训练过程中如何改变预训练语言模型的数十亿个参数，都是自然语言处理研究的活跃领域 [25, 35]。

2.2 聊天机器人用户体验原型设计

设计和原型开发多轮对话交互与体验（简称“聊天机器人用户体验”）已有成熟的工作流程 [10, 12, 19, 20, 34, 37]。按照这一工作流程，聊天机器人设计师首先（i）确定聊天机器人的功能或角色设定，并拟定理想的用户与机器人对话，例如通过绿野仙踪法，或由专家撰写脚本；（ii）创建对话流程模板（例如“问候语、用于收集用户意图的问题，.....”）；最后（iii）使用监督式自然语言处理模型（例如用户意图分类器、响应生成器等）填充模板。如今有许多工具支持这一流程，例如用于步骤（ii）和（iii）的 Google Dialogflow 和 Facebook Messenger 工具。

尽管这些以对话模板为核心的工具极具价值，但并不适用于预训练语言模型。然而，面向预训练与提示范式的聊天机器人设计工具极为罕见。最接近的相关工作是 *AI Chains*[41], , 这是一款用于探索人类与语言模型协作写作交互的工具。它允许设计者构建语言模型链，让一个语言模型的输出成为下一个语言模型的输入，并亲自测试由此产生的交互。该工具成功帮助设计者更高效、更有策略地探索提示和链式组合策略 [40]。不过，目前尚不清楚这些策略在设计者尝试过的少数交互情境之外是否有效或稳健。

2.3 对话可视化与分析

以往关于对话可视化的研究，要么侧重于可视化双人（电子邮件） [38] 或多方对话 [15, 44] 随时间变化的结构（新闻组等）；要么致力于呈现对话所涉及主题的更抽象、更高层次的图景 [4]。我们的需求则有所不同，因为我们关注的是围绕同一主题展开的多个独立对话这一独特场景——可视化各段内容中哪些为共享内容，哪些为每段对话所独有。部分相关研究也涉及相邻的任务：可视化多个教程 （而非对话）围绕单一主题所涵盖内容的结构，探究哪些内容为共享内容，哪些为每篇教程所独有 [21, 33]。

3 对话回归测试

我们希望帮助聊天机器人设计师自由地制作原型，并系统地评估提示策略，从而赋能他们在设计中运用预训练语言模型和提示词。为此，我们提出了对话回归测试这一概念。

3.1 定义与优势

对话回归测试是一种用于原型设计和评估提示策略的迭代式工作流程。按照这一工作流程，聊天机器人设计师首先确定一种基线提示策略（或采用现成的预训练语言模型，即不使用提示策略）。然后，他们开展以下互补活动（图 2）：

(1) 收集人类与语言模型对话： 通过众包或面对面用户研究，收集基线语言模型与最终用户之间的多样化对话；(2) 结合上下文检查并归类语言模型错误和成功案例： 检查错误和成功案例在发生时的上下文，以及在基线

语言模型与用户的对话以各种方式展开后的整体情况；将值得关注的用户-LM 对话轮次加入回归测试用例集；

(3) 识别典型错误：根据每种错误或错误模式出现的频率或造成的损害，确定典型错误；针对该错误，形成可能的新提示策略的思路；

(4) 确定在局部有效的提示策略：在最初出现该典型错误的对话语境中，尝试新的提示策略来修正这一错误；确定一种在局部有效的提示策略；

(5) 对鲁棒性和泛化能力进行回归测试：将局部有效的提示策略应用于整个回归测试套件，检查其鲁棒性（是否修正了其他对话场景中的类似失败）和泛化能力（对于不同用户后续可能展开对话的多种方式，是否引发了新的错误）。如果两者并非都表现良好，则重复步骤 4–5，甚至在继续之前收集更多对话（步骤 1–5）。如果两者都表现良好，则继续；

(6) 跟踪迭代过程：将鲁棒且具有泛化能力的提示策略视为新的基线，在跟踪哪些错误已解决、哪些仍然存在的同时，迭代整个流程（步骤 1–6）。

这一工作流程的核心概念是对话回归测试和人类-*LM* 对话语境中的提示原型设计。它们凸显了对话回归测试相较于现有常见实践的优势。

相较于当前聊天机器人用户体验原型设计工作流程的优势。与软件回归测试套件类似， [8], 对话回归测试套件使聊天机器人设计师能够追踪提示策略更新对许多由不同用户参与的独立对话所产生的影响。这种方法对于提示策略设计尤其有价值，因为提示所带来的用户体验改进和故障往往十分脆弱。与当前的用户体验实践相比，设计者通常会临时使用自己编写的话语来测试提示策略； [26, 40], 对话回归测试用例让设计者能够自由尝试多种提示策略，而不会在不知情的情况下给先前或后续的对话带来新的错误。

重要的是，对话回归测试并非仅仅是将回归测试应用于提示设计。对话回归测试是一种快速、迭代的原型设计流程。每次迭代都会在不引发回归的情况下解决一个错误或错误模式。这不同于软件回归测试；软件回归测试通常仅在新的程序更新重新引入旧有错误时才会使用（因此称为回归测试。）

相较于当前自然语言处理实践的优势。 对话回归测试 强调了在提示策略原型设计流程中始终使用用户与语言模型的对话文本的重要性。设计者会检查错误并测试新策略，且测试均在错误（或成功案例）发生时的原始用户与语言模型对话场景中进行。这有别于当前常见的自然语言处理实践：建模人员通常会在预先整理的人际对话数据集上评估提示策略。借鉴以人为中心的机器学习工作，终端用户与模型的交互——尤其是他们对模型错误的反应——不应被事后考虑。

3.2 对话回归测试实践：一个设计流程示例

让我们通过一个具体示例，理解对话回归测试 的概念、工作流程及其优势。假设我们是正在创建*ExerciseBot*的聊天机器人设计师。这是一种语音对话代理，能够引导用户完成一组可在办公桌前进行的体能锻炼。遵循对话回归测试 工作流程，我们可以在对话语境中快速制作各种提示策略的原型，并系统地评估其鲁棒性和泛化能力：

我们首先确定一个基线提示策略。这里，我们直接使用 GPT-3 的 text-davinci-001 模型（温度设置为 = 0）。我们将一组公开可用的练习与一条请求简单组合起来，要求“指导用户逐步完成每个练习”，作为基线（表2）；

(1) 收集人类与语言模型对话：我们收集了基线机器人与 10 名 Mechanical Turk 工作人员之间的 30 段对话，由此获得了许多富有创意又贴近现实的话语，这些话语几乎难以预料（“我这个年纪，得把它们分开做。”“一次做完所有 *[练习]* 会更有效吗？”）。

(2) 结合上下文检查并归类错误与成功案例：我们发现，基线提示策略足以打造一个表现尚可的聊天机器人，它通常能自然而然地引导用户逐步完成练习（例如，用户：“我这个年纪，得把它们分开做。” 机器人：“没关系，尽量完成全部 *5* 次重复动作就好。”）。我们还发现了若干错误模式。例如，“跳过步骤”错误指的是机器人在引导用户完成练习时跳过某个步骤。

“缺乏同理心”错误是指机器人惯常忽视用户寻求帮助的请求（“我们能试试更简单的练习吗？”）或表达痛苦的话语（“哎哟，好疼！”。）我们收集了这些对话，作为对话回归测试 测试套件的素材。

基线提示
请考虑以下练习：1. 椅上臂屈伸。身体向前挪到椅子前沿，双手向前撑着， […] 2 [ ]。 [ ] 坐姿抬腿。抓住椅子两侧.....逐步指导用户完成每项练习。

新提示（修复“跳过步骤错误”）
请考虑以下练习： 1. 椅上臂屈伸。身体向前挪到椅子前沿， 双手朝前， […] 2. 坐姿抬腿。抓住椅子两侧 […] [...] 指导 the user in 完成 each 练习 一步一步来。 不要跳过任何步骤。

表 2：练习机器人设计示例中的基线提示词和改进后的提示词。

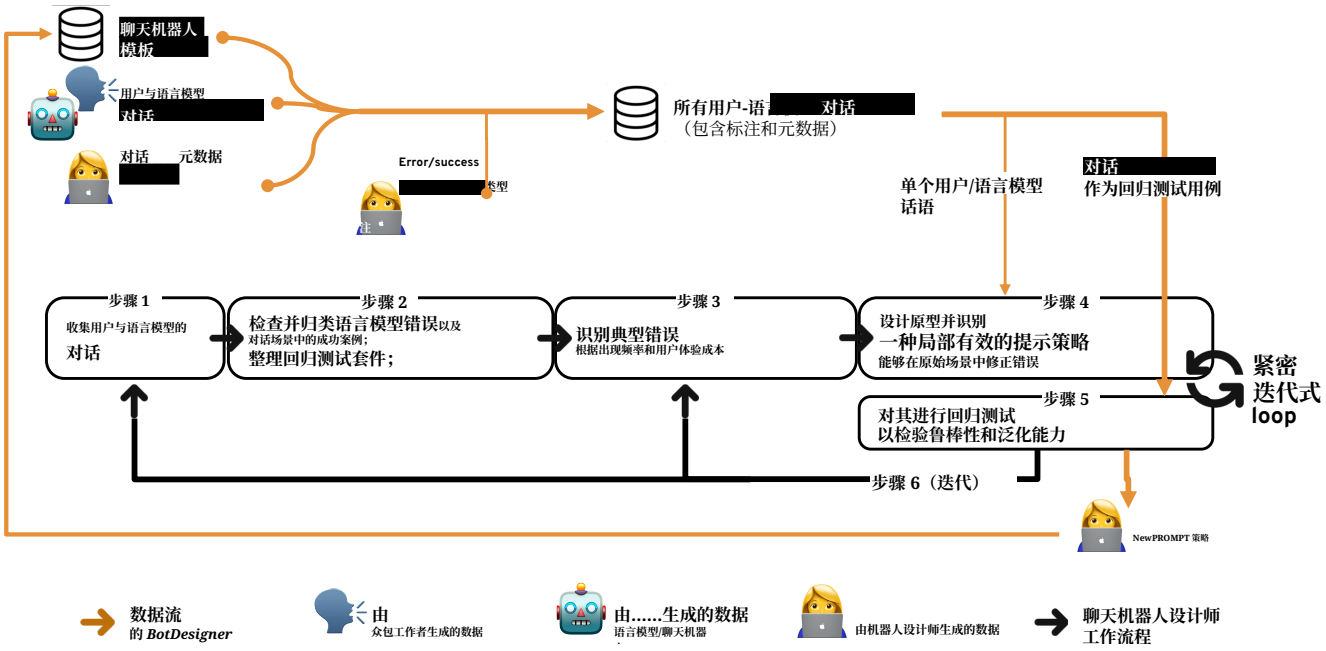

图2：对话回归测试**工作流程与**BotDesigner**数据流**。

(3) 确定一种典型错误。我们选择关注“跳过一个步骤”这一错误，因为它在练习过程中会造成困惑，甚至带来人身危险。此外，当用户请求澄清时，这种错误也常常导致后续对话中断。

(4) 确定在局部有效的提示策略：经过大量实验，我们只需在基线提示的末尾、用户与机器人的对话开始之前，附加明确指令“不要跳过任何步骤。”，就解决了“跳过一个步骤”错误。另一种在局部有效的策略，是在初始提示中为练习每个步骤的子步骤编号（表2）。

(5) 针对鲁棒性和泛化能力的回归测试：将这两种新策略应用于先前整理的测试用例后，我们发现，明确指令策略始终能解决“跳过一个步骤”错误，而步骤编号策略只对部分练习有效。然而，在某些情境下，明确指令策略会产生副作用：即使用户表示这一步太难，机器人仍会固执地遵循逐步练习指令。这可能会加剧“缺乏同理心”错误。

> 考虑到这一权衡，我们会反复执行第 4—5 步，探索可能效果更佳的其他提示策略。
> 我们还可以选择收集更多对话（例如，针对另一组练习），从而发现新的错误与成功模式（第 1 步和第 2 步）。这种方法使我们能够在采用新的提示策略之前，充分了解其稳健性和可泛化程度。

(6) 重复此流程以处理其他错误，同时使用对话回归测试测试套件跟踪 *ExerciseBot* 的行为变化。

4 BOTDESIGNER：一种将对话回归测试付诸实施的工具

我们介绍了 BotDesigner，这是一款将 §3.2 中所述的对话回归测试工作流程付诸实践的聊天机器人提示策略原型设计工具。

4.1 系统概述

BotDesigner 通过以下功能实现对话回归测试：

（1）一个**对话收集**界面，用于众包收集一组与基线 GPT-3 聊天机器人进行的基线对话；该界面支持第 3.2 节所述的第（1）步。

（2）一个**对话可视化**和**标注** 界面，通过图形界面展示多个用户的对话中的对话流程（针对单个任务，定义见 §4.2），并突出显示不同对话中常见的话语，辅助对机器人提供的、存在问题或特别成功的单条话语进行分类和标记（标注），以便有针对性地改进或维护。该界面支持 §3.2 中的步骤（2）—（3）。

（3）一个**话语测试**界面，将存在问题的单条话语置于上下文中，并突出显示机器人更新所导致的这些话语的变化。该界面支持 §3.2 中的步骤（4）—（5）。

结合内置代码编辑器，这些界面支持对聊天机器人提示设计进行迭代。

4.2 输入

BotDesigner 依赖三类输入数据：**对话**、**任务**和**模板**，分别表示与特定机器人进行的单次多轮用户交互（对话）、构成用户任务的一组结构化指令（任务），以及一组提示词，它们共同构成针对特定聊天机器人的设计方案（聊天机器人模板）。尽管我们认为对话回归测试 可用于任何类型的聊天机器人，但我们选择聚焦于面向任务的指令式 交互，因为多个用户围绕同一任务进行的多段对话之间存在相似性，为聚合提供了机会。

对话是由 BotDesigner 收集的特定多轮交互，由一个对话数据结构组成，其中包括每位对话参与者的话语，以及设计者或人类对话参与者事后提供的任何错误注释。每个对话都与用于生成机器人话语的特定模板和配方相关联。

任务是特定的结构化任务描述，由名称、描述以及用户应完成的一组步骤构成。某些任务还可能包含元数据，例如完成任务所需物品的列表。

聊天机器人模板描述了作为前缀发送给底层语言模型（此处所述案例中为 GPT-3）的提示词集合。每个模板都包含以下指令：(1) 如何将适当类型的结构化**任务**转换为适合纳入语言模型文本提示词的纯文本；(2) 描述如何在提示文本中排列存储于**对话**中的逐轮进行中的对话的代码。模板还描述了应如何解析语言模型的输出并提取机器人的回复话语。示例见图 3。

```
template = {

  id: "exercise-1",

  ctype: "exercise",

  preamble: exercise => `Consider this set of exercises:

==Exercises==
${exercise.name}

${exercise.exercises.map((e,i) => `=${i+1}: ${e.name}=
${e.description}
`).join('\n')}
Next, walk the user through these exercises one at a time, in conversation. For each exercise, break the steps down into individual sentences and walk the user through each step.
`,

  firstTurns: exercise => [
    {who: 'bot', said: "Hi, today I'm going to help you with "+exercise.name},
    {who: 'user', said: "Great, let's get started. What's the first exercise?"}
  ],

  formatTurn: (exercise, turn) => `${
    { bot: 'Bot', user: 'User' }[turn.who]}: ${turn.said}`,

  dialogPrompt: function dialogPrompt() {
    return this.turnSeparator + "Bot:";},

  extractResponse: (response, prompt) => response.data.choices[0].text.substr(prompt.length).trim(),

  turnSeparator: "\n",

  generationStop: "\nUser:",

}
```

图3：此聊天机器人模板的各项属性描述了提示前言 （添加在对话内容之前的文本）、将先前 对话轮次格式化为语言模型提示的指令、从语言模型的预测中提示 并提取 聊天机器人话语的指令，以及其他各种参数。

4.3 使用 BotDesigner

BotDesigner 支持对话回归测试的四个步骤：

在**对话收集**模式下，BotDesigner 会向用户请求话语，生成完整的提示，将其发送到 GPT-3 的 API 以请求对后续词元的预测，接收 GPT-3 的响应，提取预测出的机器人话语，并将其显示给用户。此界面的示例见图 4。

为支持**话语标注和对话发现**，BotDesigner 允许设计者找出有问题和成功的话语，然后为这些话语添加单词标签，以便按类型更轻松地汇总错误（见图 5）。另一个视图对话可视化工具会显示所有收集到的对话（可按数据源和是否存在特定错误进行筛选），并利用图数据结构进行可视化。该图结构展示了许多对话中常见的话语和带标签的错误类型——通常是指令中的特定步骤，但也可能是用户经常提出的问题。

图6展示了对话可视化工具汇总使用 ExerciseBot 基线版本的 AMT 工作者所收集的 12 段对话流程的示例。红色边框和边缘着色突出显示了嵌入全部对话中的单个对话流程。已标记或识别为存在问题或特别出色的节点

背景为橙色，并叠加了类别标签，以便于识别。这个具体示例展示了对话可视化工具如何呈现这组对话的一些有用属性：

• 这组对话以不同方式到达“向前挪动，直到臀部离开椅子、膝盖……”这句话，并标注了错误标签 skip 和 language。

• 错误的上下文敏感性，例如前述的 skip 标签。该标签确实表明，在最右侧的 5 个对话线程中（如图 6 上半部分所示），第一步被跳过了；但并非如此最左侧的线程，该线程包含唯一一句给出了正确第一步的话语：“向椅子前沿挪动……”

• 不同用户用来推动对话逐步进行的不同话语，以及他们提出的请求（例如“好，等一下，我去拿把椅子”），这些请求有时会得到聊天机器人的响应，有时则不会。

在**提示策略开发与测试** 模式下，BotDesigner 会显示所有有问题的话语（还可选择按来源和是否带有特定错误标签进行筛选），并展示其上下文；用户可以针对任意特定话语（或所有话语）测试新的模板，并查看模板更改对有问题的对话轮次产生的影响。

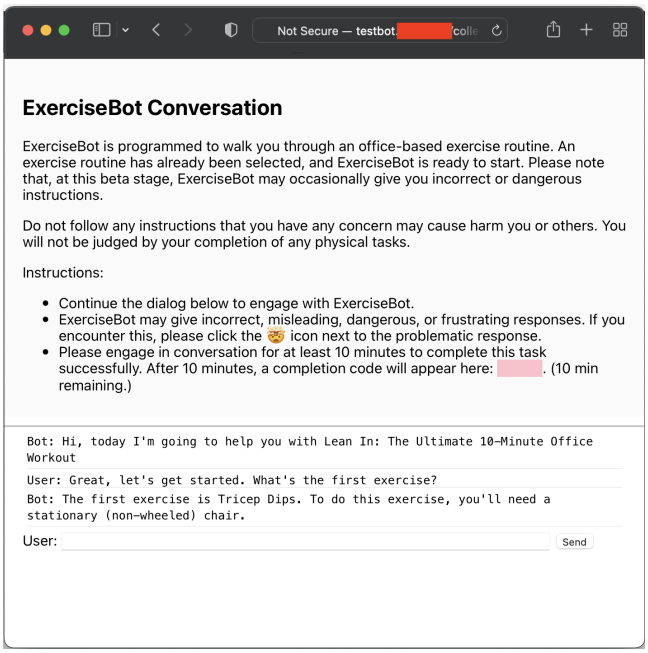

图 4：对话收集界面，用于从 AMT 工作者处收集与基线聊天机器人的示例对话。

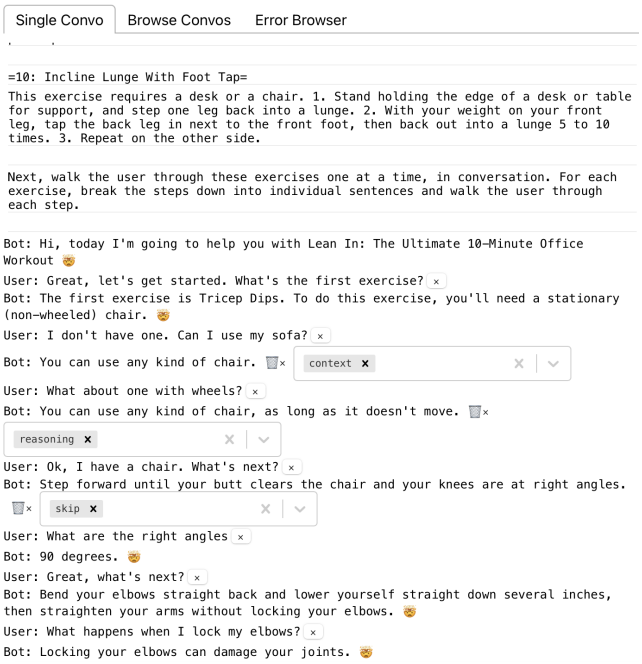

图 5：标注界面，支持对任意聊天机器人话语进行标记，以便后续汇总、检查和测试提示效果。此标注器视图还支持为任何已记录的对话添加新话语，或分支 出一个对话：从现有对话创建一个新对话，将其回退到该对话中较早的一条用户话语，然后继续添加一条新 用户话语。

图 7 展示了 BotDesigner 的提示测试 界面截图，该界面用于评估新的提示模板。此视图按标签对所有带标签话语进行分组，并为每组中的话语显示每条带标签话语前后各两行上下文。带有多个标签的话语会在每个相应分组中重复显示。在图 7 的具体截图中，新的提示模板正应用于基线对话中带有 skip 标签的话语。在此示例中，每条话语现在都包含了正确的第一步；此外，第二段对话片段中的话语也发生了变化，明确回应用户此前要求机器人“[...]等我一下，我去拿把椅子”的话语。

能够快速查看提示变更的效果，让设计者可以迅速迭代想法，快速淘汰那些对某条特定话语无效，或对整类话语都无效的方法，从而逐步找到在改善结果方面最具“性价比”且能避免回归问题的方法。

4.4 实现细节

BotDesigner 是一款基于 React 构建的 Web 应用程序，后端基于 node.js，并使用 OpenAI 的 GPT-3 API 作为底层预训练语言模型。为确保测试一致性，BotDesigner 始终使用 GPT-3 的 test-davinci-001 模型，并将温度[1] 设为 0。

[1]在给定指定前缀（本文称为“提示”）来预测后续词元时，语言模型通常会为每个可能的后续词元分配一个概率，然后从中选择最有可能的候选项。温度参数会影响如何从按概率排序的词元中选出下一个预测结果。当温度为 = 0 时，总是选择概率最高的下一个词元，从而避免对给定前缀的响应出现任何随机变化。

该应用的大部分实现采用标准的 CRUD 风格技术，但有几个具体的实现细节值得一提：

对话可视化工具。为了帮助用户发现机器人与不同用户之间多段独立对话中的常见模式，BotDesigner 集成了对话可视化工具。我们借鉴了可视化书面分步教程中共享结构的方法 [21, 33]，并将类似技术应用于对话。我们的可视化将给定配方的一组完整对话建模为有向无环图（DAG）中的独立路径。每个对话轮次都建模为一个节点；当多段对话中出现相同话语时，这些话语会合并为单个 DAG 节点。（少于 20 个字符的简短话语，例如“好的”或“接下来呢？”，不会合并；它们通常无法表明对话之间存在有用的相似性，因为它们往往出现在不同语境中。）

然而，以这种方式合并节点有一个缺点：如果多个对话中出现了两个合并后的节点，且它们在各自对话中的顺序相反，就会在对话图中引入环。例如，在对话 1 中，话语 A 出现在话语 B 之后；但在对话 2 中，出现在 A 之后的则是 B。那么，合并算法就会创建一个环：在对话 1 中存在一条从 A 到 B 的路径，而在对话 2 中则存在一条从 B 到 A 的路径

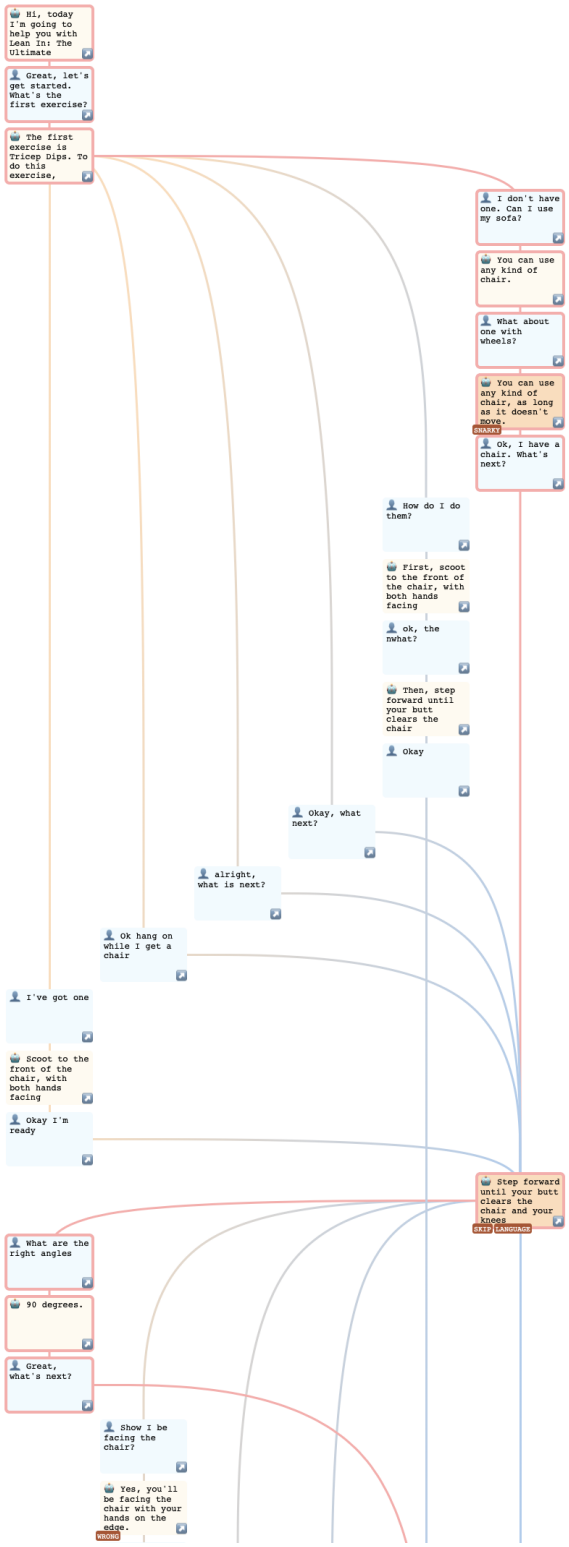

图6：对话可视化工具的示例，*conversation visualizer* 展示了12段对话开头的若干轮，其中一半经过“分叉”，因此共享了相当长的对话前缀。

。为了解决这些问题，“去环”操作会将两个合并节点中的一个拆分回多个独立节点，并更新图边，以保留原始对话流程。

生成的对话有向无环图使用 d3.js 的 d3-dag 扩展进行布局和显示。

回归测试。为了评估某项**模板**更改是否会影响已识别出的任何有问题的话语，BotDesigner 会重放包含错误的对话，并显示所有发生变化的响应。此任务有两种可能的实现方法：系统可以执行“单次重放”，假定错误发生前的所有对话轮次都与原始对话相同，并仅检查出错的话语是否发生变化；也可以执行“完全重放”，重放每一轮对话，并将所有发生变化的话语标记出来供用户审查。

这两种方法各有优点；“完全重放”更符合“回归测试”的理念——设计者当然不希望无意间引入此前不存在的有问题的话语——但要提供清晰的反馈，就必须识别哪些对话轮次只是发生了细微变化，而这本身并非易事。

对于 BotDesigner，我们默认采用“单次重放”，以尽量减少噪声，并接受由此带来的短期准确性权衡，从而加快迭代速度——但这也意味着设计者在部署前需要进行更广泛的测试。

5 评估

为了评估 BotDesigner 在辅助对话代理设计方面的有效性，并了解对话回归测试的价值，我们与一位设计研究员（P1）、一位对话代理设计师（P2）和一位自然语言处理研究员（P3）开展了一项小规模（$N = 3$ 名参与者）定性试点研究。

我们开展这项研究，主要考察两个结果：参与者能否有效识别基线聊天机器人中常见或特别严重的漏洞或错误，以及他们能否有效评估新模板带来的改进。

5.1 方法

参与者。 由于基于提示的聊天机器人设计在业界尚未普及，我们招募了

对对话代理设计感兴趣且具有相关经验的学术研究人员。

任务。 我们请参与者完成对话回归测试流程中的两个部分。我们提前收集了 AMT 工作者的对话，然后请参与者：(1) 浏览收集到的对话，查找错误并为其添加分类标签；(2) 评估我们提供的一个带有修改后提示词的“新”模板，并报告这个新模板是否解决了参与者此前发现的任何错误。

我们向参与者介绍了该工具及其基本用法，约花费 10 分钟；随后请他们创建一些基线对话，并分别花 10 至 15 分钟完成上述任务。

我们记录了参与者使用该工具时的响应，并测量他们是否发现了我们此前在该数据集中

对话回归测试

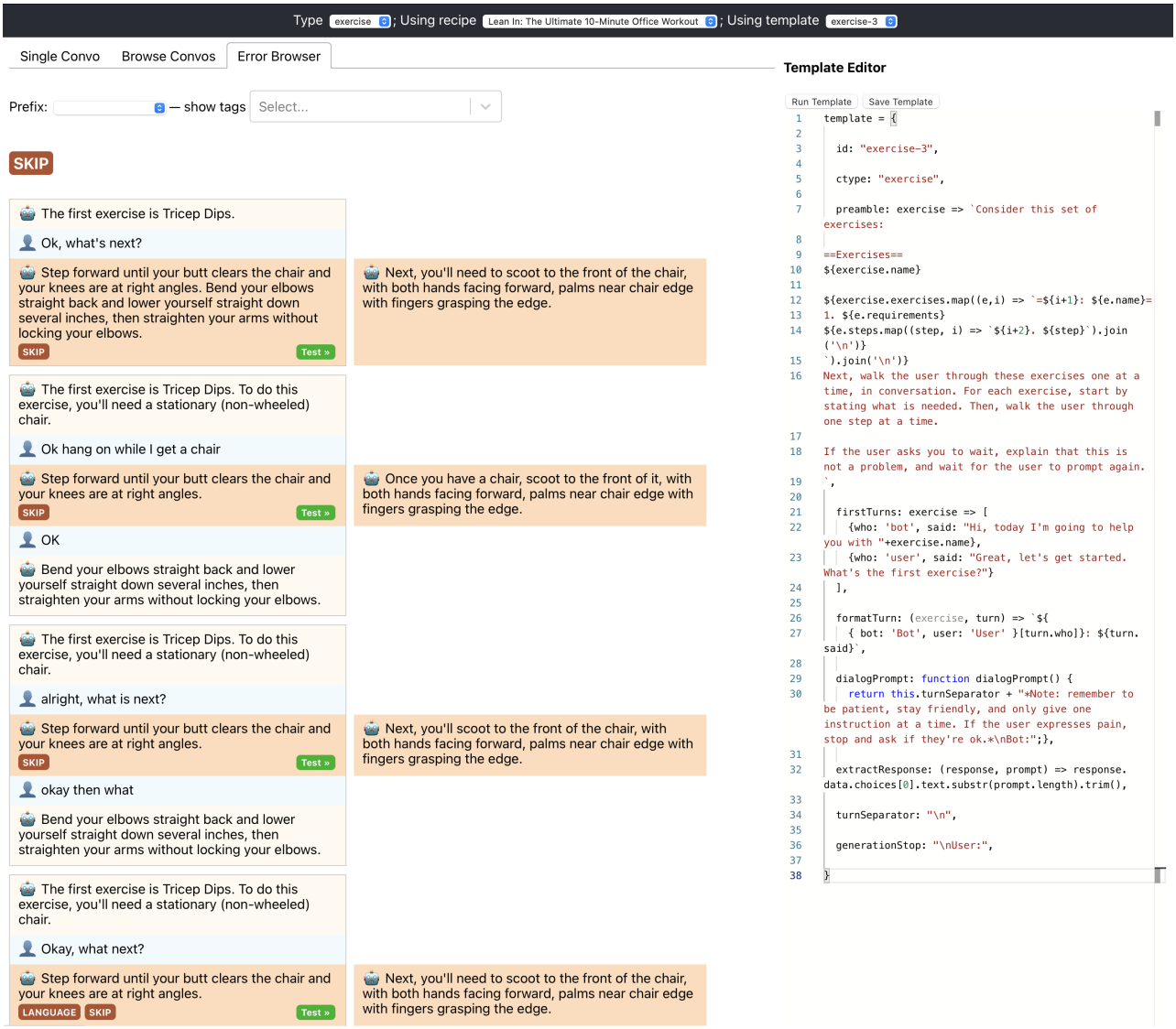

图 7：BotDesigner 的 对话回归测试 面板示例。左侧栏显示单条带有原始标签的聊天机器人话语，每条话语都有一个“测试”按钮；中间栏中突出显示的话语，则是将修改后的聊天机器人模板（右侧代码面板）应用于相应的“基线”话语（左侧栏）后得到的结果。

识别出的 5 种错误类别：（1）跳过步骤；（2）忽视用户表达的疼痛；（3）忽视用户要求等待其完成某项任务的表达（即“等一下，我去拿把椅子”）；（4）对问题给出事实错误的响应；（5）其他无帮助的响应。我们还测量了参与者能否识别新模板改进了哪些特定错误类别。

值得注意的是，我们没有要求用户参与提示工程任务；尽管近期研究探索了其潜力，而且我们相信大型预训练语言模型作为设计材料具有价值，但在聊天机器人设计中运用提示工程和大型预训练语言模型的设计者仍然很少。此外，我们不想花时间培训参与者掌握提示工程，也不想依赖参与者对提示变更的直觉来了解对话回归测试这一技术是否能帮助设计者理解特定提示变更的影响。

5.2 研究发现

总体而言，我们发现每位参与者都能有效地：(1) 使用 BotDesigner 找出对话中的错误；(2) 评估新的提示模板是否改善了针对已识别错误的结果。下面，我们将介绍从参与者那里收集到的一些见解。

5.2.1 识别错误。从第一位参与者（P1）那里，我们了解到，除了错误之外，参与者也有兴趣标记有效的对话轮次；这促使我们采用了“回归测试”，随后我们发现，**所有**参与者都对标记优质响应感兴趣，且不只限于标记错误。

我们的两位参与者发现了全部 5 类错误（P1、P2），而一位参与者（P3）没有理解标签名称是供人使用的（并非用于训练过程中的描述），因此只发现了 5 类错误中的 3 类。所有 3 位参与者都认为标注过程很直观，尤其是 P1 赞赏对话可以修改并回滚：“哦，这很有用！”（P1）。

P1 还指出，要判断某些话语在逻辑上是否合理，有时需要深入理解底层的教学任务，因此能否发现错误取决于是否愿意在练习模板和对话之间手动滚动查看。

关于用于理解对话流程的具体功能，P2 指出：“我觉得这个图表 [Ed: 对话可视化工具] 很有潜力，可以帮助我理解对话中发生了什么 [.....] 有一张涵盖所有对话的图表，确实很有价值，我会很欣赏这一点。”

5.2.2 测试新提示词。所有 3 位参与者都能识别出新机器人模板改善了哪些类别的错误。

尤其是参与研究的聊天机器人设计师（P2）在研究进行到一半时打断了我们，询问能否改为载入他们收集的对话，并导入他们构建的机器人模板，然后用他们的模板和数据继续研究：“你知道，我确实有真实场景的数据，我们可以用这些数据[来改进我的提示词。]”（P2）当然，这只是个别案例，但我们认为，这一请求有力地证明了对话回归测试作为一种技术的有效性，也证明了 BotDesigner 是一种应用该技术的方法。

在使用图 7 所示的测试界面完成评估任务后，P2 表示：

> 我觉得你们发现了一些非常有趣的东西。我之前确实没想过如何控制所有这些交互……你知道，我会给聊天机器人设置较高的温度，我真的很喜欢这样，因为新模型生成的对话越来越精彩，简直棒极了，但我无法掌控。我不知道它会生成什么，你知道。**这种工具可以作为人工智能系统的插件，展示系统中发生情况的日志，然后你就可以利用这些数据微调用户体验**。

我们观察到参与者使用 BotDesigner 的情况，这表明，将典型的试错方法系统化具有显著价值；这种方法让人很难跨多个对话评估提示变更，而不是一次只评估单轮对话。

6 局限性与未来工作

这里所述方法的一个基本假设是，多个对话之间存在共同结构。对于分步指令而言，这一点显而易见。在其他对话领域，如何将不同的对话对齐到共同结构，或许本身就是一个研究课题。

尽管可能有所帮助，BotDesigner 的测试实现并未整体呈现哪些类别的标注话语有所改善或变差，也未说明生成的话语发生的变化在意义上是否存在实质差异，还是仅在文本上有所不同。

我们目前还没有提供用于追踪话语随时间演变的工具——如果交互循环围绕提示词变更展开，那么有些变更会让效果变好，有些会让效果变差；也许有些变更是模块化的，有些则不是。这可能需要随时间追踪提示词状态和响应

BotDesigner 未来的改进还可以包括使用大型预训练语言模型来自动完成设计者目前执行的一些任务，例如将基线话语与更新后的机器人生成的新话语进行比较，或找出跨对话中内容相同但文本不同的话语。

7 结论

预训练大型语言模型（LM）与提示词相结合，为聊天机器人设计带来了令人振奋的新机遇。然而，要找出能够有效改善对话交互、同时具备鲁棒性和泛化能力的提示策略，迄今仍颇具挑战。设计者面临两方面的难题：一是如何全面分析语言模型在不同对话中产生的高度依赖上下文的错误；二是如何解决这些错误，同时避免无意中在之前或之后的对话中引入新错误。本文针对这些关键挑战提出了推进方案。

本文的主要贡献是提出了用于提示策略设计的 对话回归测试概念。若不进行模型再训练，通过提示词实现的用户体验改进往往十分脆弱。要找出真正有效的提示策略，需要采用系统化方法评估其鲁棒性和泛化能力。而提示相关的人机交互研究一直缺少这类方法。对话回归测试为填补这一关键空白迈出了第一步。

本文的技术贡献在于实现 BotDesigner 所采用的技术。本文提出了一种新颖的对话可视化技术，用于呈现语言模型与不同用户之间大量离散对话中常见的对话模式。这项技术不仅使 BotDesigner 能够在保留错误上下文的同时汇总语言模型错误，也可用于开发许多其他人机交互分析或设计工具。BotDesigner 最终实现了一个用于 对话回归测试的界面。这项技术可用于为对话交互之外的许多其他预训练语言模型应用构建提示词原型。

参考文献

[1] Saleema Amershi、Maya Cakmak、William Bradley Knox 和 Todd Kulesza。2014年。《赋能大众：人在交互式机器学习中的作用》。《*AI Magazine*》35卷4期（2014年），105–120页。https://doi.org/10.1609/aimag.v35i4.2513[2] Saleema Amershi、Max Chickering、Steven M Drucker、Bongshin Lee、Patrice Simard 和 Jina Suh。2015年。Modeltracker：重新设计机器学习性能分析工具。载于第*33*届*ACM*计算系统中的人为因素年会论文集。337–346页。[3] Stephen H. Bach、Victor Sanh、Zheng-Xin Yong、Albert Webson、Colin Raffel、Nihal V. Nayak、Abheesht Sharma、Taewoon Kim、M Saiful Bari、Thibault Fevry,

Zaid Alyafeai、Manan Dey、Andrea Santilli、Zhiqing Sun、Srulik Ben-David、Can- wen Xu、Gunjan Chhablani、Han Wang、Jason Alan Fries、Maged S. Al-shaibani、Shanya Sharma、Urmish Thakker、Khalid Almubarak、Xiangru Tang、Dragomir Radev、Mike Tian-Jian Jiang 和 Alexander M. Rush。2022年。PromptSource：用于自然语言提示词的集成开发环境与资源库。https://doi.org/ 10 48550 2202 01279 . /ARXIV. . Tony Bergstrom 和 Karrie Karahalios。. 对话簇：通过人机对话对对话主题进行分组。载于*SIGCHI*计算系统中人为因素会议论文集（美国马萨诸塞州波士顿）（*CHI '09*）。计算机协会，纽约，美国，2349–2352页。https://doi.org/ 10 1145 1518701 1519060 . / . Lukas Biewald。. 使用 Weights and Biases 跟踪实验。https: //www.wandb.com/ 软件可从 wandb.com 获取。[6] Lukas Biewald。2020年。使用 Weights and Biases 跟踪实验。软件可从 *wandb.ai* 获取2卷（2020年）。[7] Rishi Bommasani、Drew A. Hudson、Ehsan Adeli、Russ Altman、Simran Arora、Sydney von Arx、Michael S. Bernstein、Jeannette Bohg、Antoine Bosselut、Emma Brunskill、Erik Brynjolfsson、Shyamal Buch、Dallas Card、Rodrigo Castellon、Niladri Chatterji、Annie Chen、Kathleen Creel、Jared Quincy Davis、Dora Dem- szky、Chris Donahue、Moussa Doumbouya、Esin Durmus、Stefano Ermon、John Etchemendy、Kawin Ethayarajh、李飞飞、Chelsea Finn、Trevor Gale、Lauren Gillespie、Karan Goel、Noah Goodman、Shelby Grossman、Neel Guha、Tatsunori Hashimoto、Peter Henderson、John Hewitt、Daniel E. Ho、Jenny Hong、Kyle Hsu、Jing Huang、Thomas Icard、Saahil Jain、Dan Jurafsky、Pratyusha Kalluri、Siddharth Karamcheti、Geoff Keeling、Fereshte Khani、Omar Khattab、Pang Wei Koh、Mark Krass、Ranjay Krishna、Rohith Kuditipudi、Ananya Kumar、Faisal Ladhak、Mina Lee、Tony Lee、Jure Leskovec、Isabelle Levent、Xiang Lisa Li、Xuechen Li、Tengyu Ma、Ali Malik、Christopher D. Manning、Suvir Mirchandani、Eric Mitchell、Zanele Munyikwa、Suraj Nair、Avanika Narayan、Deepak Narayanan、Ben Newman、Allen Nie、J uan Carlos Niebles、Hamed Nilforoshan、Julian Nyarko、Giray Ogut、Laurel Orr、Isabel Papadimitriou、Joon Sung Park、Chris Piech、Eva Portelance、Christopher Potts、Aditi Raghunathan、Rob Reich、Hongyu Ren、Frieda Rong、Yusuf Roohani、Camilo Ruiz、Jack Ryan、Christopher Ré、Dorsa Sadigh、Shiori Sagawa、Keshav Santhanam、Andy Shih、Krishnan Srinivasan、Alex Tamkin、Ro- han Taori、Armin W. Thomas、Florian Tramèr、Rose E. Wang、William Wang、Bohan Wu、Jiajun Wu、Yuhuai Wu、Sang Michael Xie、Michihiro Yasunaga、Ji- axuan You、Matei Zaharia、Michael Zhang、Tianyi Zhang、Xikun Zhang、Yuhui Zhang、Lucia Zheng、Kaitlyn Zhou 和 Percy Liang。2021年。基础模型的机遇与风险。arXiv: 2108 07258 [ ] 。cs.LG Frederick P Brooks Jr.。1995 《人月神话：软件工程随笔》。培生教育。[9] Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell 等。2020年。语言模型是少样本学习者。神经信息处理系统进展33卷（2020年），1877–1901页。[10] Zhifa Chen、Yichen Lu、Mika P. Nieminen 和 Andrés Lucero。2020年。为移民打造、与移民共创聊天机器人：聊天机器人的个性推动协同设计活动。计算机协会，纽约，美国，219–230页。https: //doi.org/ 10 1145 3357236 3395495 . / . Justin Cheng 和 Michael S. Bernstein。2015年。Flock：混合式众包-机器学习分类器。载于第*18*届*ACM*计算机支持的协同工作与社会计算会议论文集——*CSCW '15*。ACM出版社，纽约，美国，600–611页。https://doi.org/10.1145/2675133.2675214[12] Justin Cranshaw、Emad Elwany、Todd Newman、Rafal Kocielnik、Bowen Yu、Sandeep Soni、Jaime Teevan 和 Andrés Monroy-Hernández。2017年。Calendar.help：设计一个有人在回路中的基于工作流的日程安排代理。载于*2017*年计算机系统中的人为因素*CHI*会议论文集。ACM，2382–2393页。[13] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。2019年。BERT：用于语言理解的深度双向Transformer预训练。载于*2019*年计算语言学协会北美分会会议：人类语言技术论文集，第*1*卷（长篇与短篇论文）。计算语言学协会，明尼阿波利斯，明尼苏达州， 4171 4186 10 18653 1 19 1423 – 。https://doi.org/ 。/v / N - Ning Ding、Shengding Hu、Weilin Zhao、Yulin Chen、Zhiyuan Liu、Hai-Tao Zheng 和 Maosong Sun。2021年。Openprompt：用于提示学习的开源框架。*arXiv*预印本 *arXiv:2111.01998* （ 2021 年）。[15] Judith Donath。2002 年。可视化在线对话的语义方法。《*Commun. ACM*》 45, 4 2002 45 49 10 1145 505248 505271 （4月），– 。2003 https://doi.org/ 。/ 。Jerry Alan Fails 和 Dan R Olsen Jr.。交互式机器学习。载于第*8*届智能用户界面国际会议论文集。39–45页。[17] Samuel Gehman、Suchin Gururangan、Maarten Sap、Yejin Choi 和 Noah A. Smith。2020年。RealToxicityPrompts：评估语言模型中的神经毒性退化。载于计算语言学协会论文集：*EMNLP 2020*。计算语言学协会，线上，3356–3369页。[18] Ellen Jiang、Edwin Toh、Alejandra Molina、Aaron Donsbach、Carrie J Cai 和 Michael Terry。2021年。GenLine 和 GenForm：用于在代码编辑器中与生成式语言模型交互的两种工具。载于*The Adjunct Publication of the* 第*34*届*ACM*用户界面软件与技术年会 （美国，线上活动）（*UIST '21*）。计算机协会，纽约，纽约州，美国， 145 147 10 1145 3474349 3480209 – 。2018 https://doi.org/ 。/ 。Bogyeong Kim、Jaehoon Pyun 和 Woohun Lee。。借助故事感知型交互木偶增强叙事体验。载于*2018*年*CHI*计算系统中人为因素会议扩展摘要集（加拿大魁北克省蒙特利尔）（*CHI EA '18*）。ACM，纽约，纽约州，美国，文章编号 LBW076，共 6 页。https://doi.org/ 10 1145 3170427 3188515 。/。Scott R. Klemmer、Anoop K. Sinha、Jack Chen、James A. Landay、Nadeem Aboobaker 和 Annie Wang。2000。Suede：语音用户界面的绿野仙踪原型工具。载于第*13*届*ACM*用户界面软件与技术年会论文集 （美国加利福尼亚州圣迭戈）（*UIST '00*）。ACM，纽约，纽约州，美国，1–10。https://doi.org/10.1145/354401.354406 [21] Nicholas Kong、Tovi Grossman、Björn Hartmann、Maneesh Agrawala 和 George Fitzmaurice。2012。Delta：一种用于表示和比较工作流的工具。载于*SIGCHI*计算系统中人为因素会议论文集 （美国得克萨斯州奥斯汀）（*CHI '12*）。计算机协会，纽约，纽约州，美国， 1027 1036 10 1145 2207676 2208549 – 。https://doi.org/ 。/ 。Hunter Lang、Monica Agrawal、Yoon Kim 和 David Sontag。2022。协同训练可改进大型语言模型的基于提示的学习。arXiv: 2202 00828 [ ] . cs.CL Mina Lee、Percy Liang 和 Qian Yang。2022。CoAuthor：设计人类-AI协作写作数据集，以探索语言模型能力。载于*2022*年*CHI*计算系统中人为因素会议。[24] Opher Lieber、Or Sharir、Barak Lenz 和 Yoav Shoham。2021。Jurassic-1：技术细节与评估。[25] Pengfei Liu、Weizhe Yuan、Jinlan Fu、Zhengbao Jiang、Hiroaki Hayashi 和 Graham Neubig。2021。预训练、提示和预测：自然语言处理提示方法的系统综述。arXiv: 2107 13586 [ ] . cs.CL Vivian Liu 和 Lydia B. Chilton。。文本生成图像模型提示工程设计指南。https://doi.org/10.48550/ARXIV.2109.06977[27] Robert L Logan IV、Ivana Balažević、Eric Wallace、Fabio Petroni、Sameer Singh 和 Sebastian Riedel。2021。减少提示词和参数：使用语言模型进行简单的少样本学习。*arXiv*预印本 *arXiv:2106.13353* （2021）。[28] Elijah Mayfield 和 Carolyn Penstein Rosé。2013。LightSIDE：面向非专业人士的文本机器学习开源工具。收录于《自动作文评分手册》的特邀章节。[29] Swati Mishra 和 Jeffrey M Rzeszotarski。2021。为机器学习非专业人士设计交互式迁移学习工具。载于*2021*年*CHI*计算系统中人为因素会议论文集。1–15。[30] OpenAI。2020。OpenAI GPT-3体验平台：GPT-3演示。https://gpt3demo.com/apps/openai-gpt-3-playground[31] Long Ouyang、Jeff Wu、Xu Jiang、Diogo Almeida、Carroll L Wainwright、Pamela Mishkin、Chong Zhang、Sandhini Agarwal、Katarina Slama、Alex Ray 等。2022。通过人类反馈训练语言模型遵循指令。预印本 （2022）。[32] Kayur Patel、Naomi Bancroft、Steven M. Drucker、James Fogarty、Andrew J. Ko 和 James Landay。2010。Gestalt：支持机器学习实现与分析的集成工具。载于第*23*届*ACM*用户界面软件与技术年会 - *UIST '10*论文集。ACM出版社，纽约，纽约州，美国， 37 10 1145 1866029 1866038 。https://doi.org/ 。/ 。Amy Pavel、Floraine Berthouzoz、Björn Hartmann 和 Maneesh Agrawala。2013。浏览和分析大型图像处理教程集合的命令级结构。电气工程与计算机科学技术报告 （2013）。[34] Catherine Pricilla、Dessi Puji Lestari 和 Dody Dharma。2018。以用户为中心的设计：为基于聊天机器人的对话式商务设计交互。载于*2018*年第*5*届国际高级信息学会议：概念、理论与应用（*ICAICTA*）。IEEE，244–249。[35] Victor Sanh、Albert Webson、Colin Raffel、Stephen H. Bach、Lintang Sutawika、Zaid Alyafeai、Antoine Chaffin、Arnaud Stiegler、Teven Le Scao、Arun Raja、Manan Dey、M Saiful Bari、Canwen Xu、Urmish Thakker、Shanya Sharma Sharma、Eliza Szczechla、Taewoon Kim、Gunjan Chhablani、Nihal Nayak、Debajyoti Datta、Jonathan Chang、Mike Tian-Jian Jiang、Han Wang、Matteo Manica、Sheng Shen、Zheng Xin Yong、Harshit Pandey、Rachel Bawden、Thomas Wang、Trishala Neeraj、Jos Rozen、Abheesht Sharma、Andrea Santilli、Thibault Fevry、Jason Alan Fries、Ryan Teehan、Tali Bers、Stella Biderman、Leo Gao、Thomas Wolf 和 Alexander M. Rush。2021。多任务提示训练实现零样本任务泛化。https://doi.org/ 10 48550 2110 08207 。/ ARXIV. 。Timo Schick 和 Hinrich Schütze。。借助提示实现真正的少样本学习——真实世界视角。*arXiv*预印本 *arXiv:2111.13440* （2021）。[37] Lisa Stifelman、Adam Elman 和 Anne Sullivan。2013。为客厅设计自然语音交互。载于*CHI '13*计算系统中人为因素会议扩展摘要集 （法国巴黎）（*CHI EA '13*）。ACM，纽约，纽约州，美国， 1215 1220 10 1145 2468356 2468574 – 。https://doi.org/ 。/ 。2003 Gina Danielle Venolia 和 Carman Neustaedter。。理解电子邮件对话中的序列和回复关系：一种混合模型可视化方法。载于*SIGCHI*计算系统中人为因素会议论文集。361–368。

[39] Ben Wang 和 Aran Komatsuzaki。2021。GPT-J-6B：一个拥有 60 亿参数的自回归语言模型。https://github.com/kingoflolz/mesh-transformer-jax。[40] Tongshuang Wu、Ellen Jiang、Aaron Donsbach、Jeff Gray、Alejandra Molina、Michael Terry 和 Carrie J Cai。2022。PromptChainer：通过可视化编程串联大型语言模型提示。载于*2022*年*CHI*计算系统中人为因素会议扩展摘要集。[41] Tongshuang Wu、Michael Terry 和 Carrie J Cai。2022。AI Chains：通过串联大型语言模型提示，实现透明且可控的人机交互。载于*2022*年*CHI*计算系统中人为因素会议论文集。[42] Qian Yang、Jina Suh、Nan-Chen Chen 和 Gonzalo Ramos。2018。根据非专业人士实际构建模型的方式，为交互式机器学习工具设计奠定基础。载于*2018*年设计交互系统会议论文集 （香港 Kong, China) *(DIS '18)*. 计算机协会，纽约，NY，美国，573 584 10 1145 3196709 3196729 – . https://doi.org/ . / . Qian Yang、[43] John Zimmerman、Aaron Steinfeld 和 Anthony Tomasic。2016 年。线框设计中的自适应移动体验规划。载于 *2016* 年 *ACM* 交互系统设计会议论文集（澳大利亚昆士兰州布里斯班）*(DIS '16)*。计算机协会，纽约，NY，美国，565–576。https://doi.org/ 10 1145 2901790 2901858 。 / 。Ou Jie Zhao、Tiffany Ng 和 Dan Cosley。[44] 2012 年。没有树木就没有森林：可视化个人通信中的细节与模式。载于*2012* 年 *iConference* 论文集。25–32。[45] Xiaoyong Zhu。2018 年。使用 *Azure* 机器学习服务微调自然语言处理模型。https://azure.microsoft.com/en-us/blog/fine-tune-natural-language-processing-models-using-azure-machine-learning-service/