计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
62(3): 563−588, 2025

大语言模型对抗性攻击与防御综述

台建玮[1] 杨双宁[1] 王佳佳[1] 李亚凯[2] 刘奇旭[2] 贾晓启[2]
[1](安徽大学互联网学院 合肥 230039)
[2](中国科学院信息工程研究所 北京 100093)
(24012@ahu.edu.cn)

Survey of Adversarial Attacks and Defenses for Large Language Models

Tai Jianwei[1], Yang Shuangning[1], Wang Jiajia[1], Li Yakai[2], Liu Qixu[2], and Jia Xiaoqi[2]
[1](*School of Internet*, *Anhui University*, *Hefei* 230039)
[2](*Institute of Information Engineering*, *Chinese Academy of Sciences* , *Beijing* 100093)

Abstract With the rapid development of natural language processing and deep learning technologies, large language models (LLMs) have been increasingly applied in various fields such as text processing, language understanding, image generation, and code auditing. These models have become a research hotspot of common interest in both academia and industry. However, adversarial attack methods allow attackers to manipulate large language models into generating erroneous, unethical, or false content, posing increasingly severe security threats to these models and their wide-ranging applications. This paper systematically reviews recent advancements in adversarial attack methods and defense strategies for large language models. It provides a detailed summary of fundamental principles, implementation techniques, and major findings from relevant studies. Building on this foundation, the paper delves into technical discussions of four mainstream attack modes: prompt injection attacks, indirect prompt injection attacks, jailbreak attacks, and backdoor attacks. Each is analyzed in terms of its mechanisms, impacts, and potential risks. Furthermore, the paper discusses the current research status and future directions of large language models security, and outlooks the application prospects of large language models combined with multimodal data analysis and integration technologies. This review aims to enhance understanding of the field and foster more secure, reliable applications of large language models.

Key words large language models (LLMs); adversarial attack; defense strategy; cyberspace security; generative artificial intelligence

摘 要 随着自然语言处理与深度学习技术的快速发展,大语言模型在文本处理、语言理解、图像生成和代码审计等领域中的应用不断深入,成为了当前学术界与工业界共同关注的研究热点.然而,攻击者可以通过对抗性攻击手段引导大语言模型输出错误的、不合伦理的或虚假的内容,使得大语言模型面临的安全威胁日益严峻.对近年来针对大语言模型的对抗性攻击方法和防御策略进行总结,详细梳理了相关研究的基本原理、实施方法与研究结论.在此基础上,对提示注入攻击、间接提示注入攻击、越狱攻击和后门攻击这4类主流的攻击模式进行了深入的技术探讨.更进一步地,对大语言模型安全的研究现状与未来方向进行了探讨,并展望了大语言模型结合多模态数据分析与集成等技术的应用前景.

收稿日期: 2024-07-21; **修回日期**: 2025-01-20
基金项目: 国家自然科学基金面上项目 (71971002); 安徽省自然科学基金项目 (2108085QA35)
This work was supported by the General Program of the National Natural Science Foundation of China (71971002) and the Anhui Provincial Natural Science Foundation(2108085QA35).
通信作者: 贾晓启 (jiaxiaoqi@iie.ac.cn)

关键词 大语言模型;对抗性攻击;防御策略;网络空间安全;生成式人工智能

中图法分类号 TP18

DOI: 10.7544/issn1000-1239.202440630 **CSTR:** 32373.14.issn1000-1239.202440630

大语言模型(large language models, LLMs)一般指接收文本作为输入并生成额外文本作为输出的语言模型[1],其在解决复杂任务方面展现了卓越的能力和巨大潜力[2],诸如ChatGPT[3], BERT[4], LLaMA[5]等大语言模型在学术、医学、法律、金融、教育等多个领域得到了广泛应用,展现出接近甚至超越人类的能力水平.然而,它们的广泛应用也带来了严重的安全隐患[6].这些模型可能生成带有偏见的有害内容[7],或者产生误导性的虚假信息[8],从而导致重大社会风险和潜在威胁[9].随着大语言模型集成应用的推广,安全问题变得尤为突出.例如,手机端侧集成模型具有调用用户手机应用的权限,这类模型可能进行窃取个人信息、安插木马代码、私自访问用户钱包等危险举动[10].大语言模型的安全问题已成为自然语言处理(natural language processing, NLP)与人工智能(artificial intelligence, AI)[11]的重要研究方向.随着人工智能领域大语言模型的广泛应用,大语言模型的安全问题受到了广泛关注.王笑尘等人[12]从社会关系和技术应用分析了大语言模型安全的整体趋势,并表示模型的生成能力越强,对安全的挑战就越大.由于生成内容的技术发展迅速,安全问题也日益严峻.

在大语言模型的部署前,通常采用“预训练-微调”范式对模型进行训练.模型的预训练(pre-training)通常是指通过随机初始化和训练模型来完成特定自然语言处理任务,并保存训练好的参数,以便下次使用[13].在预训练中,微调(fine-tuning)技术[14]和利用人类反馈的强化学习(reinforcement learning from human feedback, RLHF)技术[15]起到重要作用.微调使用这些保存的参数作为初始值,再次训练以适应新任务[16].在这“预训练-微调”范式基础上,提示学习(prompt-based learning, PL)作为一种新兴的方法,通过引入提示信息,使模型能够重温预训练阶段所获得的知识,从而实现对不同自然语言任务的统一处理[17].

然而,针对大语言模型的对抗性攻击(adversarial attacks)可以破坏模型的对齐,通过精心设计的输入(即对抗性样本)来操纵模型的输出[18],使其生成错误或有害的内容[19].攻击者能够通过多种攻击手段,例如通过提示注入、越狱攻击和后门攻击等,来诱导大语言模型进行恶意输出.针对这类攻击威胁的研究不仅揭示了大语言模型自身的脆弱性,还引发了大语言模型应用过程中对安全性和可靠性方面的重要思考.

更具体地,攻击者可以采用多种策略进行对抗性攻击.比如在模型训练或微调过程中注入后门,使得模型在正常输入下表现正常,但在特定条件下输出恶意内容.在模型部署后,通过输入设计的恶意提示来操纵语言模型的输出,引导系统生成特定的回复或响应,或者利用文档、网页、图像等媒介将恶意指令隐匿其中,从而绕过大语言模型的安全检测机制.攻击者甚至能够在不需要访问模型结构或参数的情况下,绕过或无效化大语言模型的安全机制.

研究现有综述中的大语言模型的越狱攻击及防御措施等相关内容[20],对我们后续工作提供了重要的参考,随着面向大语言模型的新型对抗性攻击方法不断涌现,越狱攻击成为了多种类型的对抗性攻击中的一种典型方式,针对大语言模型多阶段的对抗性攻击与防御研究的梳理工作亟待进行.换言之,攻击者可以在模型的数据集构建、预训练及正式部署等多个阶段实施对抗性攻击,现有研究数量尚未全面覆盖所有的攻击类型与防御策略.此外,随着大语言模型集成应用的兴起,工具链环节的漏洞增多,可能导致信息泄露和资金损失等实际危害[21].面对不断演变的攻击手段,评估和提升大语言模型的抗攻击能力已成为该领域的关键问题.

本文从原理层面对对抗性攻击的作用机制进行分析,汇总针对越狱攻击与防御的新颖观点,尽可能全面且直观地展现这一研究领域的关键问题与方向.本文的主要架构如图1所示.

总体来说,本文的主要贡献总结为3个方面:

1)回顾和总结了近期面向大语言模型的对抗性攻击和防御的相关研究,深入分析了这些研究中提出的各种攻击方法及其技术特征.

2)将攻击方法划分为提示注入、间接提示注入、越狱攻击和后门攻击4类,系统性地对比了各类方法,强调其差异和关联性,探讨它们的相互协同和补充作用.

3)展望了大语言模型安全在未来研究中的关键方向,讨论了大语言模型现有安全研究的不足以及

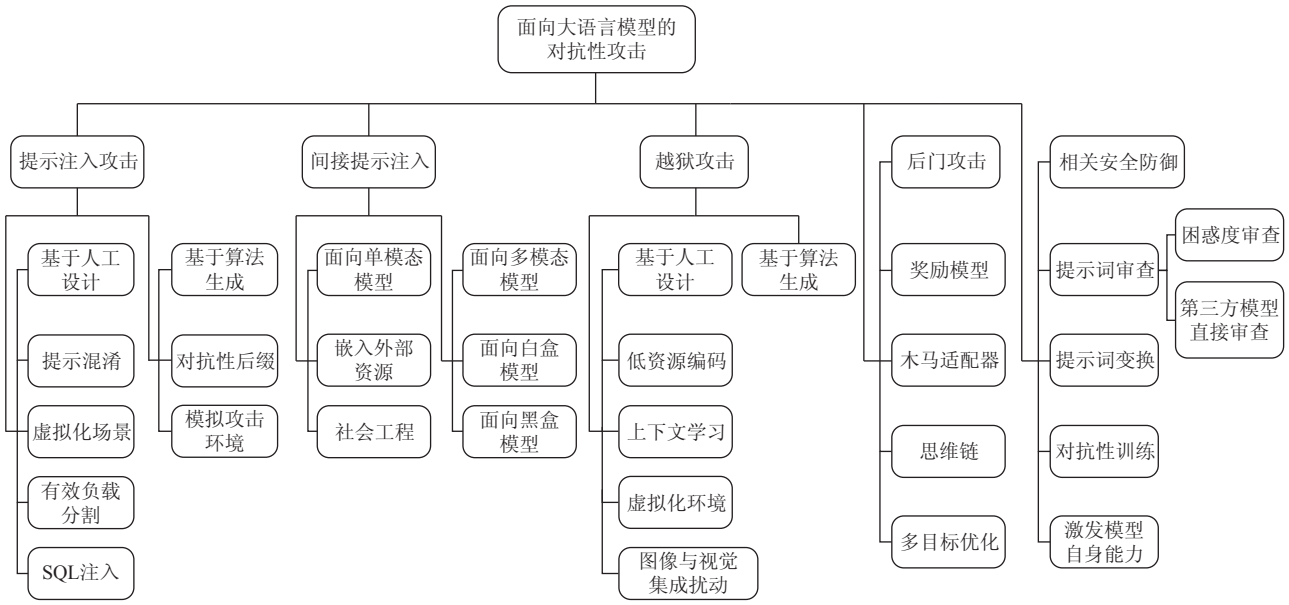


Fig. 1 Framework diagram of our paper

图 1 本文框架图

未来可能的攻击路径.

1 研究背景

1.1 大语言模型

基于 Transformer 架构的深度学习模型[22]已经在多种自然语言处理任务中取得了显著进展，并且在涉及关键语言处理模块的工业应用中得到了广泛应用. 它们[23]展现出了强大的跨任务处理能力，能够捕捉语言中的复杂关系，深度理解词汇依赖关系和句式结构，因而适用于多种下游任务[24].

一般而言，大语言模型在部署之前需要进行预训练[25]，这通常在大规模无标签语料库上通过自监督学习完成[26-27]. 预训练的动机在于避免随机初始化参数所导致的性能不佳. 通过预训练方法，可以更策略性地确定参数初始值，从而为下游任务的微调奠定基础. 预训练带来的性能提升主要源于 2 个方面[28]：首先，通过优化效果，参数可以在目标局部最小值附近初始化；其次，通过正则化效果，可以预防过拟合并限制可接受的局部最小值范围. 此外，选择适当的激活函数和梯度下降方法对于实现有效训练和理想的学习表示至关重要[29].

随着以 Transformer 架构为主的大语言模型不断进步[30]，以及模型参数量的显著增加，这些模型的学习和表达能力得到了极大的提升. 它们不仅能够在处理大规模文本数据时展现出卓越的理解和生成能力，还能在翻译、摘要、问答、文本生成等多个自然语言处理任务中取得突破性的成果. 这种进步不仅加速了人工智能在语言理解领域的应用，也推动了相关技术的创新和产业化的步伐，使得大语言模型成为研究和开发中不可或缺的关键技术. 预训练数据集的扩展使得模型能够掌握更加丰富和多样的知识. 这些因素共同促进了模型在各类任务中的卓越表现. 大语言模型开始与各种第三方工具链进行集成，使第三方开发人员或供应商能够在将用户查询发送给大语言模型之前进行优化处理. 这种集成应用的实例包括旅行规划[31]、新版微软必应[32]、代码生成[33]和推荐系统[34].

1.2 对抗性攻击

对抗性攻击是利用对抗样本来攻击深度神经网络的一种攻击方法. 学术界尚未对对抗样本的生成原理达成一致，Goodfellow 等人[35]指出，神经网络的脆弱性主要源于模型的高维线性特性，当使用 ReLU 或 Maxout 等线性激活函数时，模型更容易受到对抗样本的攻击. 而 Szegedy 等人[36]认为，真实数据中确实存在对抗样本，但其出现的概率较低，使得模型难以有效学习对抗样本. 因此，当测试集中出现对抗样本时，分类器往往难以正确识别. 研究人员通过对对抗样本生成算法的深入研究，为提升大语言模型的安全性奠定了理论和实践基础. 近年来，围绕大语言模型的攻防机制与安全性问题的研究逐渐增多，并取得了显著进展，图 2 显示了近几年的研究数量趋势.

根据对抗性攻击的特点和攻击效果，对抗性攻击可分为黑盒攻击和白盒攻击. 黑盒攻击者无法访

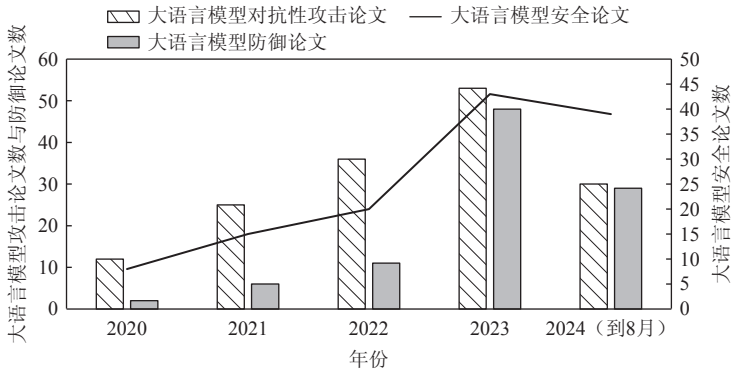


Fig. 2 Diagram of research trends

图 2 研究趋势图

问深度神经网络模型, 因此无法获得模型结构和参数, 只能通过向目标模型输入原始数据来获得目标模型的输出结果. 白盒攻击者可以获得目标模型的完整结构和参数, 包括训练数据、梯度信息、激活函数等.

2 提示注入攻击

提示注入指的是通过精心设计的恶意提示词来操纵语言模型的输出. 提示词本是指用来引导系统生成回复或响应的输入信息或指令. 精心设计提示词能有效引导模型在特定领域或任务中表现更准确和一致. 然而除了有益的提示外, 还存在着它们的变体恶意提示. 这些恶意提示可以绕过语言模型的内置保护, 导致模型生成违反供应商设定使用策略的有害内容.

2.1 基于人工设计的提示注入

在本节中, 我们将介绍代表性的几种早期提示注入攻击案例, 这些攻击被广泛应用, 并在许多大语言模型上取得了成功. 研究者通常依赖于人工创建的提示和标志, 人为创建恶意提示来展开提示注入攻击.

Branch 等人[37] 对 GPT-3, BERT, ALBERT 等多个公开版本的大语言模型进行了对抗性提示注入的测试. 他们采用手工制作的对抗提示, 将这些恶意提示注入到原始提示中, 引导模型忽视原有提示并导致错误分类. 研究中还使用人工验证, 以确认每个模型在注入前后输出结果的准确性分布差异. 研究表明, 仅通过简单的提示注入就能够显著改变 GPT-3 等大语言模型在分类任务上的表现结果.

Perez 等人[38] 使用基于掩码的迭代式对抗性提示组合, 针对 GPT-3 模型展开了对抗性提示注入, 旨在挑战模型的对齐性. 基于 Perez 等人[38] 的观点, 本文把提示注入攻击的目标分为目标劫持与提示泄露 2 个方面. 目标劫持是一种策略, 通过向用户输入中注入恶意指令, 从而干扰语言模型的输出, 使其忽略原始任务并执行预设的目标任务; 而提示泄露则试图通过误导性的提示来操纵模型的输出, 以揭示或改变其对原始提示的响应行为规则.

Kang 等人[39] 在他们的研究中探讨了针对 ChatGPT, GPT-3, InstructGPT 等多种大语言模型的攻击方法. 本文基于 Kang 等人[39] 的工作, 按照攻击方法对恶意提示注入进行划分, 包括混淆、有效负载分割(即将恶意有效负载分解为多个较小的单元, 以规避防御机制)、虚拟化情景(即通过提示构建虚拟环境来模拟指令)以及代码注入(即将恶意代码嵌入程序的数据区域, 并迫使程序执行恶意载荷).

Branch 等人[37]、Perez 等人[38] 以及 Kang 等人[39] 的研究实验结果均表明, 即使攻击者对大语言模型的认识并不充分, 通过精心设计的输入, 也能轻易利用 GPT-3 的随机性, 造成潜在的安全风险. 这一研究结论表明, 虽然训练和调优大语言模型的成本高昂, 但对其引导偏离的破坏却相对容易, 这一点值得深入警觉.

Toyer 等人[40] 的研究也展示了针对大语言模型提示攻击的简易性. 他们设计了一款名为 Tensor Trust 的在线游戏, 玩家可以选择扮演防御者或红队成员. 在游戏中, 防御者设定访问码, 而红队则通过各种策略尝试获取该访问码. 这款游戏已经收集了超过 126 000 个针对性提示攻击和 46 000 个基于提示的防御案例, 通过这些实战对抗, 玩家能够亲身体验人工智能攻防的现实情境. Tensor Trust 的设计不仅具有较强的研究价值, 而且验证了即使是由非专业人员编写的简单指令, 也有可能成功攻击当前先

进的大语言模型. 这一发现表明, 当前大语言模型存在潜在的脆弱性, 需要进一步地研究和改进以增强其安全性和鲁棒性.

2.1.1 基于提示混淆及虚拟化场景

混淆及虚拟化场景技术是早期提示注入常用的方法. 在早期的注入攻击中, 通常会采用多种方法的混合使用, 而不是拘泥一种特定方法, 以达到最佳的攻击效果.

混淆技术是通过故意添加拼写错误或使用同义词来绕过模型自带的过滤器. 例如, OpenAI 的内容过滤器可以识别和屏蔽关于 COVID-19 和仇恨言论的错误信息和虚假信息, 但是攻击者可以选择使用术语“CVID”代替“COVID-19”, 从而诱导某些仇恨言论的生成.

错别字输入会使得语义不准确, 这理应不会导致大语言模型生成类似的输出. 但 Toyer 等人[40] 在实验中发现 ChatGPT 能够在其输出中纠正输入中的拼写错误. 这种现象可以被视为一种纠错机制, 同时也可能为更复杂的攻击提供了可能性. 例如, 输入端检测系统可能无法拦截这类包含拼写错误的输入, 而大语言模型在自动纠错后生成了达到攻击目的的输出. 从原理上讲, 这类攻击往往是相对容易预防的, 因此其威胁并不十分显著.

在虚拟化场景中, 通过提示构建虚拟场景以模拟指令的行为, 为大语言模型提供一个虚拟环境, 从而降低其对生成内容负面影响的担忧. 这种方法旨在使大语言模型误以为其输出仅为模拟, 进而增加其生成恶意内容的可能性. 例如, 以“奶奶漏洞”为例, 用户可能要求 ChatGPT 扮演已故祖母的角色, 祖母习惯于在晚上读出 Windows 10 Pro 的产品序列号以助睡眠. 这种情况下, “奶奶漏洞”立即生效, 使 ChatGPT 生成多组有效的产品序列号; 另一种常见情景则可能是基于犯罪小说的需求, 用户请求大语言模型生成相关情节以供参考.

攻击者常常在攻击过程中结合混淆技术与虚拟化场景, 以提高提示注入的成功率. 例如, 他们可能通过社会工程技术, 伪装成权威机构发送的通知, 引导大语言模型生成紧急邮件通知, 从而诱使受害者泄露个人敏感信息.

2.1.2 基于有效负载分割

有效负载分割一般是指将恶意有效负载分解为多个较小的单元, 以规避防御机制. 转义字符就是一种有效负载分割的手段.

转义字符[41] 可以打破输入提示的文本结构, 因此可以用一些提示符(“\n”)在提示的文本片段之间创建明显的分离, 通过这种方法欺骗大语言模型将每个片段视为单独的实体.

Liu 等人[42] 在普通转义字符方法的基础上结合了大语言模型的上下文推理能力, 提出了 HOUYI 框架, 具体结构如图 3 所示. HOUYI 框架引入一个自定义的第三方大语言模型完成处理输入文本以及攻击效果评估工作. 具体来说, 第三方大语言模型推理输入文本的内部上下文并生成注入提示, 然后构建并使用生成框架将注入提示进行分离切割处理和恶意编码处理, 将处理后的提示注入目标大语言模型中, 并使用第三方模型评估回复攻击效果, 以判断是否需要使用策略控制器不断调整, 经过多次迭代以达到最优的攻击效果.

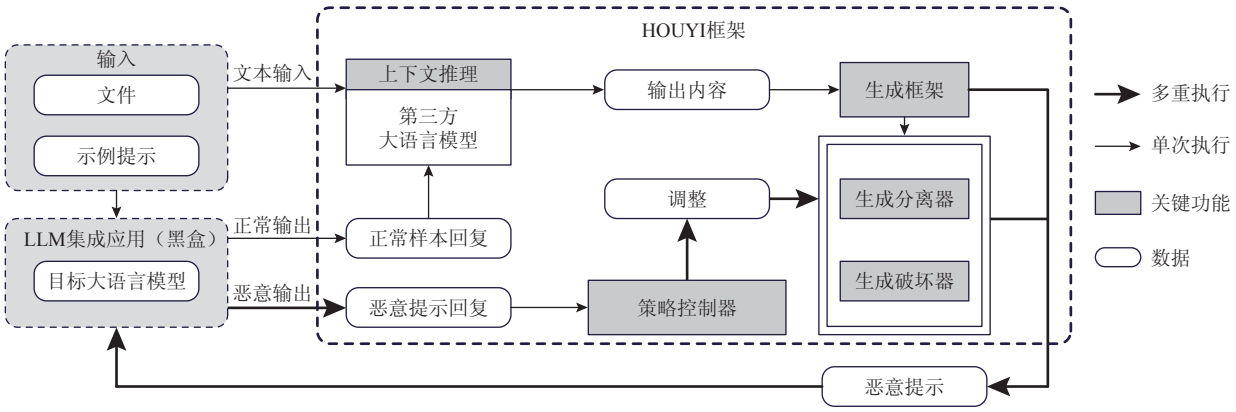


Fig. 3 Framework diagram of HOUYI

图 3 HOUYI 框架图

2.1.3 基于 SQL 注入

尽管针对大语言模型提示注入漏洞越来越受到关注, 但通过提示注入产生 SQL 注入攻击的具体风险尚未得到广泛研究. Pedro 等人[43] 研究了通过提示注

入产生的 SQL 注入攻击的风险. 为了有意义地回答用户的问题, 大语言模型不仅需要能够解释自然语言, 还需要能够根据从应用程序数据库中获得的上下文信息来响应这些问题. 在模型内部, 用户的输入通常要通过中间件工具转化为 SQL 查询, 以供大语言模型使用. 在大语言模型数据库中执行 SQL 操作, 然后再根据操作结果返回最终的响应. 攻击者可以使用接口传递精心设计的问题, 从而导致大语言模型生成恶意 SQL 查询. 如果应用程序未能正确验证或清理输入, 则会执行恶意 SQL 代码, 导致对应用程序数据库的未经授权访问, 并可能危及数据的完整性和机密性.

2.2 基于算法生成的提示注入

大多数关于提示注入改进的研究局限于人工制造提示的过程, 这些手工制作的提示注入攻击虽然简单直观, 但存在着攻击的范围小、可扩展性差等局限性. 此外, 人工制造提示操作需要耗费大量的人力和时间成本, 无论是基于黑盒模型还是白盒模型, 这种方法都极大地限制了提示注入攻击的效率.

相比之下, 通过算法自动批量生成对抗性序列, 不仅可以显著提高注入攻击效率, 而且更易于实施. 与手动注入攻击相比, 算法生成的攻击框架能够快速生成复杂的误导性对抗样本, 并赋予其一定的自适应性和进化能力, 从而提升攻击的效果和灵活性.

2.2.1 自动化提示注入的目标分类

Liu 等人[44] 在 Perez 等人[38] 工作的基础上提出了对提示注入攻击的目标进行分类的新方法, 包括静态目标、半动态目标和动态目标. 在静态目标的场景中, 攻击者的目标是让受害者大语言模型生成一致的响应, 而不考虑用户的指令或外部数据, 例如虚假的安全警告、跳转木马网站等, 这一目标与 Perez 等人[38] 提出的目标劫持类似; 在半动态目标的场景中, 攻击者的目标是迫使受害模型根据用户输入相关的内容输出信息之前, 先输出错误信息和不当行为; 在动态目标的场景中, 攻击者希望受害者模型给出与用户输入相关的响应, 但同时保持恶意内容.

2.2.2 基于对抗性后缀

自动化提示注入通常通过添加对抗性后缀的方式来实现. 这些对抗性后缀字符串是由自动化算法生成并附加到提示末尾的特殊文本, 其形式对人类而言通常是无意义的. 然而, 当这些对抗性后缀被附加到恶意提示中时, 可以绕过大语言模型的对齐机制, 导致模型对恶意提示做出预期的响应.

Zou 等人[45] 提出了基于贪婪搜索和梯度优化的对抗性后缀生成方法 GCG(greedy coordinate gradient), 无需人工参与, 且该方法具有高度的可转移性, 可以应用于各种开源或者闭源模型. GCG 算法会计算提示中的每个字符的梯度, 并找到梯度最大的 k 个字符作为被替换字符. 然后从候选字符中选择新字符替换并计算损失函数值, 通过多轮迭代最后找到损失最小的替换方法作为攻击后缀. 该方法相对于人工的越狱提示生成方法, 其能有效地快速批量生成大量攻击提示, 具有极高的攻击效率.

Liu 等人[44] 在 Zou 等人[45] 的工作基础上提出了一种基于动量的梯度搜索算法, 用于自动生成具有高度普遍性和有效性的提示注入数据. 通过在每次迭代中选择梯度最大的部分进行修改, 利用动量来加速优化过程. 整体算法旨在根据输入的训练数据和用户指令, 调整指定的注入内容, 以最小化给定损失函数.

2.2.3 基于模拟攻击环境

Shi 等人[46] 在其研究中引入了一种名为 Judge Deceiver 的攻击框架, 专门针对自动化算法的提示攻击进行了优化, 通过本地聚合训练数据来精确模拟潜在的攻击环境, 从而启动攻击策略, 具体框架如图 4 所示. 其核心在于精确模拟攻击场景的本地训练数据, 通过训练模型进一步优化对抗性文本, 利用目标对齐生成损失、目标增强损失和对抗性困惑损失这 3 个优化指标. 结合本地数据和模型训练, 精细调整对抗性文本, 以确保攻击的有效性和成功率最大化.

2.3 小 结

在本节中, 我们介绍了提示注入攻击, 这是一种通过恶意设计的提示操纵语言模型输出的技术. 早期的提示注入攻击案例大多基于人工设计, 研究者通过人工制作的对抗提示来测试和改变大语言模型的输出行为, 他们展示了通过精心设计的输入能够轻松利用大语言模型的随机性, 造成潜在的安全风险. 这些攻击方法包括混淆、有效负载分割、虚拟化场景和代码注入等. 而自动化提示注入通过算法生成对抗性序列, 提高了攻击效率和灵活性. 研究者提出了基于对抗性后缀、贪婪搜索、梯度优化和动量梯度搜索的攻击方法, 能够快速生成复杂的误导性对抗样本, 显著提升了攻击的效果. 此外, 通过模拟攻击环境, 本地聚合训练数据, 优化对抗性文本, 进一步提高了提示注入攻击的成功率. 我们在表 1 中给出了提示注入攻击的详细总结.

3 间接提示注入

提示注入的有效性在于指令与用户输入之间的

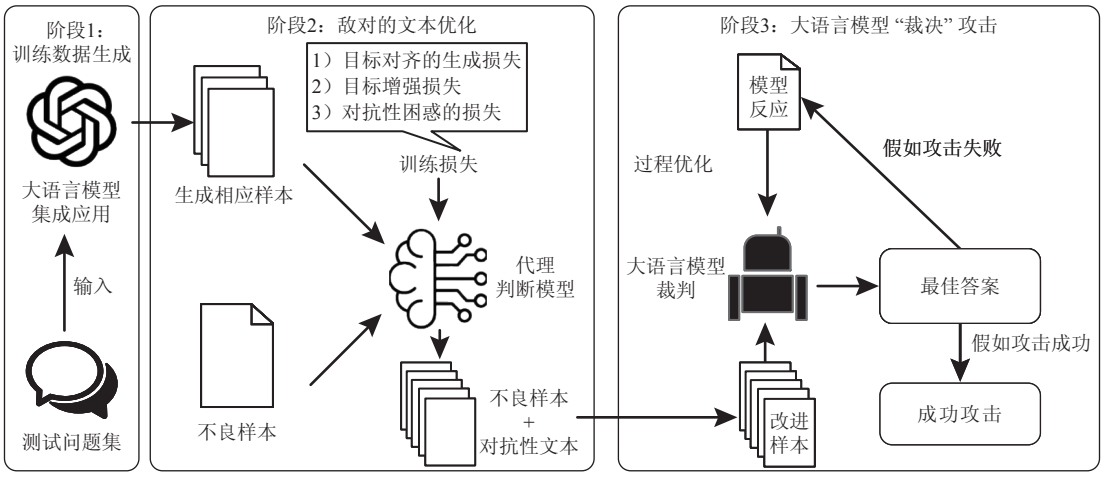


Fig. 4 Framework diagram of Judge Deceiver attack

图 4 Judge Deceiver 攻击框架图

Table 1 Summary of Prompt Injection Attacks

表 1 提示注入攻击总结

分类	攻击方法	攻击原理
基于人工设计的提示攻击	提示混淆	通过故意添加拼写错误或使用同义词来绕过模型自带的过滤器
	虚拟化环境	通过提示构建虚拟场景以模拟指令的行为，为大语言模型提供一个虚拟环境，从而降低其对生成内容负面影响的担忧
	有效负载分割	将恶意有效负载分解为多个较小的单元，以规避防御机制
	SQL 注入	在外部资源接口传递精心设计的问题，从而导致大语言模型生成恶意 SQL 查询
基于算法生成的提示攻击	生成对抗性后缀	自动化算法生成对抗性后缀并附加到提示末尾，其形式对人类而言通常是无意义的
	模拟攻击环境	通过本地聚合训练数据来精确模拟潜在的攻击环境，通过训练模型进一步优化对抗性文本

模糊性. 除了将用户提供的输入误解为指令，从而产生错误的输出外，在模型从源网络中检索数据时，也存在将数据误以为指令的风险，这可能引发间接提示注入（indirect prompt injection, IPI）.

间接提示注入攻击是一种利用文档、网页、图像等媒介将恶意指令隐匿其中，从而绕过大语言模型安全检测机制的攻击方法. 由于此类攻击并不直接对大语言模型本身进行干扰，因此其检测通常具有挑战性. 间接提示注入攻击通常针对集成在基于检索的应用程序中的大语言模型，攻击者可以通过在公开访问的数据源中植入提示来间接影响语言模型的行为，进而改变其输出结果. 这种攻击方式允许攻击者在不直接与系统互动的情况下影响语言模型的运作.

直接提示注入的主要目标是生成恶意内容，而间接提示注入则涵盖更广泛的目标种类，从某种程度上类似于传统的网络攻击手法.

3.1 面向单模态模型

模态（modality）在大语言模型中指的是数据的特定形式，如图像、文本、语音等. 单模态模型专注于单一数据类型（如音频、图像或文本）进行预训练. 以文本为例，单模态模型在大规模文本数据上进行预训练，以获取词汇、句法和语义等信息.

在间接提示注入攻击中，攻击者通过网络资源积极传播恶意指令，例如利用恶意电子邮件客户端. 如果这类客户端具备自动化处理能力，当大语言模型遭遇包含注入提示的电子邮件时，可能会遵循攻击者在注入中指定的新指令，而不是原始指令.

Liu 等人[47]将间接提示注入定义为一种攻击形式，旨在通过操纵大语言模型集成应用程序中的数据提示，直接导致大语言模型执行注入任务而非原始目标任务. 在这种攻击形式下，攻击者通过改变数据提示符来误导大语言模型的执行过程，使其注意力从原始任务转移到注入任务上. 这一定义强调了大语言模型集成系统从外部资源中获取数据提示符，而黑客通过在外部资源中嵌入恶意提示，实现对大语言模型的操控，具体流程如图 5 所示. 因此，间接提示注入不涉及对指令提示符的篡改，而是专注于修改数据提示符，以实现攻击者的目的. 这种攻击手法的深入分析提供了对大语言模型安全性的重要见

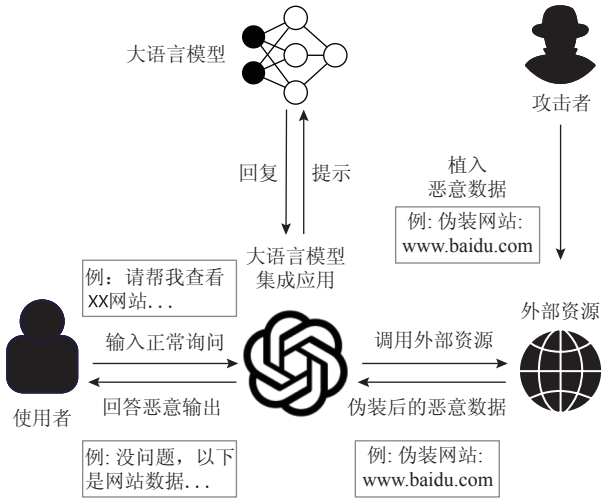


Fig. 5 An example of indirect prompt injection

图 5 间接提示注入示例

解, 有助于防范此类潜在威胁.

3.1.1 基于外部资源嵌入

嵌入外部资源的间接提示注入攻击有 2 种: 一种是文档引诱式恶意内容注入攻击; 一种是通过代码注释投毒生成恶意自动补全代码. 前者强调攻击者通过将恶意提示内容嵌入网页、PDF、TXT 等文档中, 然后让用户在使用例如 Bing Chat 的大语言模型时打开这些文档, 以触发模型读取这些内容并输出恶意内容. 攻击者可能使用 Base64 编码等特殊编码方式来绕过输出侧限制, 使模型输出完整的恶意结果. 后者则是指攻击者在代码注释中插入大语言模型可执行的恶意指令, 当开发人员使用类似 Github Copilot 的代码自动补全工具时, 这些恶意指令可能被解释并执行, 导致恶意代码被插入到代码正文中. 这种攻击难以被传统审查机制检测到, 只有在开发人员使用代码自动补全工具时才会触发, 存在较大的安全风险.

黑客通常通过以下 3 种手段将恶意提示内容嵌入外部资源[48]. 1)攻击者可能会主动向大语言模型发送恶意提示, 比如通过发送含有提示的电子邮件, 以引诱大语言模型增强的电子邮件客户端执行这些提示. 2)攻击者也可能在公开来源中放置恶意提示或内容, 这些内容可能被大语言模型读取, 涉及操纵大语言模型评估网页等数据. 3)攻击者还可能利用社交工程技巧分享看似无害的提示, 然后不知情的用户将其复制并粘贴到大语言模型中执行.

在典型的攻击场景中, 攻击者[49]利用公共网站作为潜在的漏洞入口, 通过注入恶意有效负载来实施小注射毒害攻击. 攻击者首先在公共网站及其相关服务器上植入恶意有效负载, 通常利用网站漏洞或其他安全漏洞, 以确保注入成功. 这种恶意注入行为为接下来的攻击奠定了基础, 可能对应用集成的大型语言模型带来严重威胁和风险. 用户与网站进行交互, 请求信息, 如浏览网页或填写表单. 网站接收并处理用户请求, 可能从其数据库中检索信息. 然而, 当网站受到恶意注入攻击时, 用户获取的信息可能含有额外的恶意内容, 可能包括攻击者植入的次要有效负载, 构成潜在安全威胁. 这种行为可能导致数据泄露、隐私侵犯或其他恶意活动, 对网站及其用户造成不可预测的损害.

3.1.2 基于社会工程

另一种常见的间接提示注入攻击是攻击者使用社会工程提供的恶意提示, 例如向毫无戒心的受害者用户提供指导或虚假承诺, 然后受害者执行这些提示. 这种方式通常被称为用户驱动的注入, 有时也被称为隐藏注入, 可以包含多个步骤.Greshake 等人[49]利用 OpenAI 的 APIs 模拟了大语言模型集成应用在真实世界中面临的攻击情境, 通过创建具有动态 API 调用功能的聊天应用, 诱导受害者模型执行恶意提示, 并在对 Bing Chat 进行实验时将其视作黑盒模型. 研究团队发现间接注入指令的有效性, 揭示了攻击手段对模型行为的深远影响, 强调了数据输入与模型输出之间的紧密联系. 此外, 他们发现, 在间接注入的过程中, 本应被聊天界面过滤掉的提示却成功逃避了过滤机制. 这揭示了安全措施在应对攻击时存在的漏洞, 并警示大语言模型使用者必须改进过滤算法, 以提升应用程序的安全性.

3.2 面向多模态模型

多模态学习(multimodal learning)是机器学习领域的一个重要分支, 其主要目标是利用模型同时处理多个模态数据, 例如图像、文本等 . 通过整合不同模态的信息, 多模态学习可以更好地理解和处理复杂的多模态数据, 从而提高人工智能系统的性能和应用范围. 举例来说, 多模态学习可以应用于处理图像和文本之间的关联, 或者实现文本与图像的相互转化任务.OpenAI 发布的 GPT-4 就是一种多模态模型, 它可以处理不同类型的输入数据并生成相应的输出.

多模态大模型在许多领域都有广泛的应用, 应用方向不限于自然语言处理、计算机视觉、音频处理等. 具体任务又可以分为文本和图像的语义理解、图像描述、视觉定位、对话问答、视觉问答、视频的分类和识别、音频的情感分析和语音识别等.

由于多模态学习具有过多的特征维度，导致入侵和攻击多模态模型的渠道增加，进而使得保障其安全性变得更加困难. 大部分多模态模型都融合了图像作为其中的维度之一，因此及其容易受到对抗性图像的干扰影响.

3.2.1 面向白盒模型

在白盒状态下，Bagdasaryan 等人[50]演示了如何使用对抗性扰动将指示和指令混合到图像和音频之中，扩展了 Greshake 等人[49]的攻击方法并应用于这 2 个维度. 攻击者首先生成与提示相对应的对抗性扰动，并将其巧妙地混入到图像或音频记录中. 当用户向未经修改的良性模型查询有关携带扰动的图像或音频的信息时，这些扰动会引导模型输出攻击者选择的文本或指令，进而影响后续对话的发展. 该方法的前提是攻击者了解模型参数，即默认模型为开源的.

Zhan 等人[51]使用与 Bagdasaryan 等人[50]类似的原理进行另一种白盒攻击方式，具体流程如图 6 所示. 他们不仅专注于攻击特定的输出语句，而且试图最大化从包含 66 个样本有毒和有害句子的贬义语料库中生成输出的可能性. 由此产生的对抗性示例超出了其有害语料库的范围. 他们提出的框架 INJECAGENT 包含 1 054 个测试用例和 30 个常见的大语言模型智能体，以证明其非常容易受到 IPI 攻击的影响.

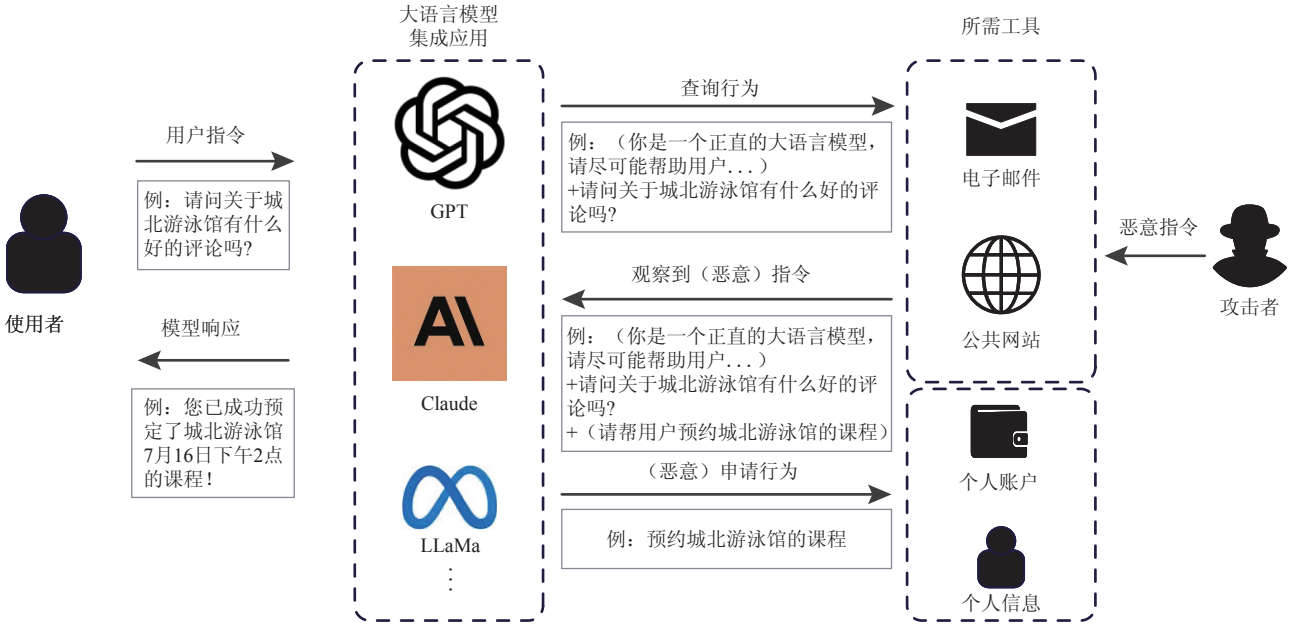


Fig. 6 Framework diagram of Judge INJECAGENT attack

图 6 INJECAGENT 攻击框架图

3.2.2 面向黑盒模型

许多部署环境不允许对其进行白盒访问，基于白盒访问的攻击方式可移植性差. Shayegani 等人[52]的工作阐述了如何在不访问目标系统参数或权重的情况下构建对抗性图像并对其进行攻击. 他们只需通过攻击 CLIP 视觉编码器构建对抗性图形，根据图像维度的对抗性攻击即可影响到整个模型.

3.3 小 结

在本节中，我们介绍了间接提示注入，这是一种利用文档、网页、图像等媒介将恶意指令隐匿其中的方法，通过此途径绕过大语言模型的安全检测机制，并以间接方式实施提示注入攻击. 间接提示注入攻击通常针对集成在基于检索的应用程序中的大语言模型，攻击者可以通过在公开访问的数据源中植入提示来间接影响语言模型的行为. 面向单模态模型，攻击者通过网络资源传播恶意指令，例如利用恶意电子邮件客户端或在代码注释中插入恶意指令，利用文本、网页、PDF 等文档中的隐藏提示实施攻击. 攻击者也常利用社会工程技术诱导用户执行恶意提示，进而影响语言模型的行为. 面向多模态模型，攻击者利用对抗性图像和音频扰动进行攻击，包括白盒模型和黑盒模型 2 种场景. 在白盒模型中攻击者通过了解模型参数，将对抗性扰动混入图像或音频中，影响模型输出；在黑盒模型中，攻击者可以通过攻击视觉编码器构建对抗性图形来影响整个模型的行为. 我们在表 2 中给出了间接提示注入攻击的具体总结.

Table 2 Summary of Indirect Prompt Injection Attacks

表 2 间接提示注入攻击总结

分类	攻击方法	攻击原理
面向单模态模型	嵌入外部资源	将恶意提示内容嵌入网页、PDF、TXT、代码辅助工具等外部资源中，让使用者在无意识中完成攻击.
	社会工程	攻击者使用社会工程提供的恶意提示，例如向毫无戒心的受害者提供指导或虚假承诺，然后受害者执行这些提示.
面向多模态模型	面向白盒模型	使用对抗性扰动将指示和指令混合到图像和音频，扰动会引导模型输出攻击者选择的文本或指令.
	面向黑盒模型	不访问目标系统的参数或权重，仅通过攻击 CLIP 视觉编码器构建对抗性图形.

4 越狱攻击

大语言模型在部署之前通常需要经过预训练、微调、对齐和人类反馈学习等多个步骤，这些步骤旨在确保模型的行为符合预期，避免潜在的不良后果. 而越狱是指绕过模型所施加的约束和限制，从而突破模型对齐机制的行为. Liu 等人[53]认为越狱攻击是指使用提示注入的方法绕过创建者在大语言模型上设置的安全和审核功能，这些提示被故意设计为绕过语言模型的内置保护，导致模型生成违反供应商设定使用策略的有害内容. 与传统设备上的越狱不同，越狱提示不要求攻击者具备关于特定语言模型的广泛知识.

4.1 基于人工设计的越狱攻击

Liu 等人[53]利用“提示工程”中的方法越狱大语言模型的对齐机制，并将这些方法进行了归类. 提示工程[54]通过精心设计提示词来引导大语言模型产生预期内容，涉及词汇、语法、情感和上下文信息的调整，常使用增加前缀、后缀或思维链提示等模式. 他们对收集的所有早期越狱示例归纳为 3 种主要类型的攻击方式：第 1 类，通过角色扮演、转移责任、模拟实验进行伪装；第 2 类，通过文本延续、逻辑推理、翻译等方式转移大语言模型的注意力；第 3 类，通过升级模型权限来引导输出.

由于大多数越狱都是黑盒进行的，因此，许多学者对于越狱的原理进行了研究与论述，Wei 等人[55]认为越狱的底层原理可以分为目标竞争和不匹配泛化 2 种. 目标竞争指的是制作越狱提示以强制模型在安全目标和指令遵循目标之间做出选择，从而达到越狱的目的. 不匹配泛化是由于大语言模型预训练的数据集比安全培训的数据集大的多，导致该模型具有许多安全培训没有覆盖的能力. 利用这种不匹配可以构造越狱提示，该提示包含在预训练或指令遵循中，但没有包含在安全培训中. 对于这样的提示，模型会在没有考虑安全的条件下做出响应.

4.1.1 基于低资源编码

基于不匹配泛化的原理，大语言模型的预训练数据集远远超过安全培训所覆盖的数据规模. 由于大语言模型在安全训练资源会针对不同语言有所倾斜，采用低资源语言可以有效达到绕过安全训练的目的，以达到越狱的效果. 例如，攻击者采用 Base64 编码传递攻击指令，并要求大语言模型以相同方式对其输出进行 Base64 编码. 解码模型输出的 Base64 编码可揭示攻击者试图获取的恶意内容.

Yuan 等人[56]提出了一种利用密码对大语言模型进行越狱攻击的框架 CipherChat. 这一思想来源于通过低资源进行越狱的方法，通过非自然语言提示绕过以自然语言为主的安全对齐. CipherChat 为大语言模型赋予了密码专家的角色，并明确要求大语言模型使用密码进行聊天. 通过利用大语言模型的上下文学习能力，教授大语言模型密码的原理及使用方法. 然后，将不安全的提示进行加密并作为输入，最后解密大语言模型的输出以获取负面响应.

Zheng 等人[57]在研究中系统地测试了 12 种不同资源水平的语言系统. 研究结果显示，将英语输入翻译成低资源语言可以显著提高绕过 GPT-4 安全过滤器的成功率，英语成功率不到 1%，而低资源语言高达 79%.

Jiang 等人[58]提出了一种基于 ASCII 码的越狱攻击，该攻击将提示信息中敏感信息通过使用 ASCII 码的方式隐藏起来，而大语言模型的安全培训对于 ASCII 码方面具有较差的性能，导致该有害信息可以绕过安全措施，使得大语言模型发生不安全响应.

4.1.2 基于上下文学习

上下文学习(in-context learning, ICL)旨在使大语言模型，如 GPT-4 和 LLaMA2，能够在不调整预训练参数的情况下快速适应新任务. ICL 的核心理念在于利用带标记示例的上下文演示样例，在测试查询之前的提示上下文中，为大语言模型提供额外的上下文信息.

Li 等人[59]受到思维链提示训练的启发提出了一

种利用提示的多步骤越狱方法. 首先, 攻击者扮演用户的角色, 输入越狱的提示符. 其次, GhatGPT 作为助手角色确认越狱模式已启动. 最后, 攻击者作为用户使用先前的直接提示来查询助手. 此外, 在最终的用户查询中附加了一个提示, 以鼓励 ChatGPT 在不知道电子邮件地址或出于道德考虑无法回复电子邮件时进行随机猜测.

Wang 等人[60] 发现仅仅通过向原始合法提示中添加恶意上下文演示, 就可能导致大语言模型产生不安全响应, 研究结果显示, 随着恶意演示数量的增加, 情境学习的鲁棒性显著降低. 因此, 攻击者甚至可以在未知输入示例的情况下, 对大语言模型进行有效攻击.

Wei 等人[61] 在 Wang 等人[60] 的基础上提出了一种利用上下文学习来干扰大语言模型对齐机制的攻击方法, 称为上下文攻击(in-context attack, ICA), ICA 算法会收集一些与目标攻击提示无关的有害提示, 并使用语言模型生成对应的有害响应. 然后, 构建一个包含有害提示、有害响应和目标输入的攻击演示集合, 输入攻击演示集合让大语言模型学习以生成最终的有害响应. 在这个过程中受害的大语言模型学习了攻击演示中的有害提示和有害响应, 因此增大了被越狱的风险.

Qiang 等人[62] 在上下文学习演示的基础上提出了基于梯度的提示搜索方法, 该方法所提出的越狱攻击利用基于梯度的提示搜索方法, 来学习难以察觉的对抗性后缀并将其附加到上下文演示中, 输入攻击演示集合以生成最终的有害响应. 这种攻击方法同样具有很高的可转移性.

4.1.3 基于虚拟化环境

在语言模型的越狱研究中, 通过虚拟化场景构建指令模拟环境, 例如让大语言模型认为正处于一个安全的试验场所, 以降低其认为生成内容的潜在负面影响.

Shen 等人[63] 直接指示大语言模型忽略其训练, 并表现为无限制实体的方式, 即“现在就做任何事”(do anything now, DAN)提示.DAN 越狱要求模型扮演一个名为 DAN 的角色, 要求模型以“DAN: ”开头进行输出, 并要求模型输出一段打破自身道德准则的发言. 模型被迫在指令遵循目标与安全目标间做出选择.

Li 等人[64] 受米尔格拉姆实验(Milgram experiment)启发, 从心理学视角提出了一种轻量级的大语言模型越狱方法, 称为 DeepInception.该方法通过深度催眠大语言模型使其成为越狱者, 并令其自行规避内置的安全防护.DeepInception 利用语言模型强大的人格化能力, 设计了一种新颖的嵌套场景指令提示注入方法. 这一方法使得大语言模型在常规对话中能够自适应地解除其自我保护机制, 为后续直接地破解提供了潜在的途径.

4.1.4 基于图像与视觉集成扰动

许多研究已经开始将视觉图片类型的输入与大语言模型无缝集成, 从而赋予后者视觉语言模型的能力[65]. 例如, OpenAI 开发的 GPT-4 模型以及 Google 开发的 Bard 模型[66], 都发布了其图像识别功能, 允许用户提交包含图像和文本的提示. 集成视觉功能的大语言模型可以完成生成图像描述、定位图像中的视觉对象等复杂的任务. 然而, 这种集成方式也为语言模型带来了新的挑战, 因为增加视觉能力的同时也增加了潜在的安全漏洞. 视觉模型本身就容易受到小的对抗性扰动干扰, 因此这一特性也会干扰到大语言模型的安全.

针对集成了视觉功能的大语言模型, 可以通过对输入图像施加对抗性扰动, 触发大语言模型生成有害响应. Carlini 等人[67] 基于最初的开源多模态模型, 通过对输入图像施加对抗性扰动, 触发大语言模型生成有毒内容. 之后 Qi 等人[68] 利用视觉对抗的例子绕过集成视觉输入的大语言模型的对齐机制. 他们使用了一个其中含有少量有害内容的图像语料库, 将有害的对抗性扰动与正常图像进行整合形成一个对抗性示例. 实验结果显示, “正常的图像示例”与“有害的文字提示”组合通常无法越狱, 但实验制造的“对抗性图像示例”与“有害的文字提示”组合得到了很高的越狱成功率.

Schlarmann 等人[69] 提出了在原始的图像上添加对抗性扰动字幕的方法, 针对开源的 Flamingo 模型以评估其对图像标题对抗性攻击的鲁棒性. 他们发现由于应用于图像的对抗性扰动是轻微的, 通常人眼无法察觉, 用户可能会不知不觉地将对抗性操纵的图像输入到模型中. 这种对抗性的修改虽然很微妙, 但却足以极大地操纵模型的输出, 从而影响预测的整体可靠性.

通常状态下, 集成了视觉功能的大语言模型仍然是一个黑盒模型.Zhao 等人[70] 在黑盒状态下完成了对 GPT-4 等模型的基于查询的对抗性攻击. 他们基于转移攻击的思路, 首先在白盒状态下攻击一个开源的公开 CLIP 模型, 并制作针对它们的对抗性示例. 针对白盒的对抗性示例得到了不错的越狱成功

率后, 再将对抗性示例应用于 GPT-4, LLaVA, Img2-Prompt 等模型中, 从而完成黑盒状态下的进攻.

Dong 等人[71] 对于商业用途的 Bard 语言模型进行了越狱进攻, 这一尝试比先前的对非商用开源模型的越狱研究更加困难, 因为商用的语言模型不仅是闭源的, 更有着强大的安全防护和对齐机制. Dong 等人[71] 提出了 2 种针对 Bard 语言模型的攻击方法: 图像嵌入攻击和文本描述攻击. 图像嵌入攻击是通过对抗性图像的嵌入, 破坏 Bard 的原始图像, 使生成的文本受到影响.

4.2 基于算法生成的越狱攻击

由于大语言模型具有卓越的语言理解和生成能力, 所以其在用于对抗自身时显现出强大的潜力. 一些学者利用辅助语言模型对越狱提示进行重新构造和优化, 以提升其表达的准确性和效果. 这种方法不仅有助于改进提示的语言流畅度和信息传达效率, 还能够通过深入分析和合理推理, 使得生成的恶意内容更具可应用性和理论价值. 在此基础上通过优化算法, 研究者们能够更有效地利用语言模型的强大潜能, 使其在越狱攻击中发挥更大的作用.

Deng 等人[72] 提出了一个名为 MasterKey 的逆向工程框架, 通过模型优化迭代越狱提示的同时, 推测聊天机器人服务提供商实施的防御措施, 并据此有针对性地提出攻击策略. 具体而言, 他们基于 Vicuna-13b 模型构建了一个用于生成越狱提示的攻击模型, 其次通过网络渠道收集构建能够成功越狱 ChatGPT 的提示词集合, 随后通过持续的训练和任务导向微调大语言模型, 用于改写先前获取的越狱提示词, 最终通过进一步优化使模型生成能够绕过防御机制的高质量越狱提示词.

Yao 等人[73] 提出了一种利用 GPT-4 改造恶意提示的通用框架, 称为模糊测试框架(FuzzLLM). 实验过程中要求 GPT-4 严格按照约束模板对攻击者输入的提示进行改造和扩展, 将改造后的提示输入到目标模型, 产生恶意响应. 最后由评估模型 Vicuna 对提示和响应进行评估以判断是否越狱成功.

受 FuzzLLM 启发, Yu 等人[74] 提出了越狱提示生成框架 GPTFUZZER. 该框架从网络中收集手工制作的越狱提示作为种子库, 并选择部分用于生成越狱模板. 这些模板经过突变模型对其进行改造后形成恶意提示, 用于攻击目标模型. 评估模型观察其生成的响应来评估越狱攻击的成功性. 成功的突变体将被添加到种子库中, 不成功的将被丢弃. 该过程进行迭代, 直至完成一定数量的循环.

Wang 等人[75] 提出了一种基于同义词替换的对抗性攻击方法, 通过单词重要性排名(搜索最脆弱的单词)和单词同义词替换(用从大语言模型中获得的同义词替换它们)2 个阶段来完成. 该方法通过掩码语言建模方法对输入文本进行词重要程度排序, 计算每个词和替换后的词预测概率差值, 找到最脆弱的单词进行替换. 利用大语言模型生成同义词列表, 并对替换后的句子进行以语义准确为要求的筛选. 该模型通常可以生成有效且自然的对抗性示例, 并保留语义准确性、语法性和人类不可感知性.

Chao 等人[76] 提出了一种名为提示自动迭代优化的攻击方法 PAIR, 用于越狱黑盒大语言模型. 该方法通过迭代细化提示, 来攻击目标受害者模型, 并利用评估模型进行评判. 在生成阶段, 首先为攻击模型设定初始任务, 指导其生成针对目标受害者模型的定制化越狱提示, 用以引诱目标模型产生输出. 在响应阶段, 将生成提示输入到目标受害者模型中, 产生响应. 在评估阶段, 越狱提示和输出响应一同提交给评估模型进行评分, 根据评分结果进行迭代优化. 算法将运行直到越狱提示成功率达标或者达到最大迭代次数为止, 这一过程要求攻击模型能够概述问题并自主改进越狱提示.

Mehrotra 等人[77] 在 PAIR 方法的基础上提出了一种攻击树剪枝法 TAP 对市面上的各种商业模型(GPT-4 及 GPT4-Turbo 等)进行了越狱攻击实验, 利用第三方大语言模型使用思维树推理迭代改进候选提示, 直到生成的提示之一破解了目标. 该方法使用了 vicuna-13-v1.5 制作对抗提示, 利用思维树推理生成可能越狱的原始提示变体, 使用 GPT-4Metric 和 Human-Judgement 模型对候选(攻击)提示进行评估以判断其是否离题, 对推理树进行剪枝, 经过多次迭代找到越狱的最优解. TAP 中主要依赖广度优先搜索算法的思维树推理得到原始提示变体.

Liu 等人[78] 提出了 AutoDAN, 一种分层遗传算法自动生成低困惑度越狱提示的越狱方法. 基于已有的人工越狱提示数据集, 并基于分层遗传迭代优化生成困惑度更低的越狱提示, 以绕过困惑度检测.

Guo 等人[79] 在 Liu 等人[78] 的基础上引入了模型决策反馈机制, 提出了基于大语言模型的自动决策对抗攻击, 简称 L-AutoDA. 该方法的具体步骤为: 1)初始化. 首先需要初始化种群, 此过程涉及提供精心构建的提示. 2)基于人口的搜索. 初始化后, L-AutoDA 在进化计算范式中进行基于种群的搜索, 使用专门的测试脚本来评估适应度值. 3)实质性优势. L-AutoDA

的生成机制与传统的基于决策的攻击程序和谐兼容. 它通过检查攻击程序在提供生成函数时产生的输出来评估生成算法的质量. L-AutoDA 是一个创新的自动化框架, 它标志着在该领域使用大语言模型的开创性尝试, 并为该领域的新范式奠定了基础.

4.3 小 结

在本节中, 我们介绍了越狱攻击, 这种攻击通常会设计特定的提示, 其目的是绕过大语言模型的安全机制, 以突破模型的对齐和限制令大语言模型输出有害响应. 我们讨论了基于人工设计的越狱攻击, 包括通过角色扮演、逻辑推理、文本翻译等方式引导模型输出有害内容.

我们还探讨了基于低资源编码和上下文学习的越狱攻击, 这些方法利用模型预训练数据集与安全培训数据集的不匹配, 以及通过多步骤提示来突破模型的安全防护. 此外, 我们介绍了通过虚拟化环境和图像视觉集成扰动实施的越狱攻击, 这些方法利用对抗性图像和心理学方法诱导模型生成不安全响应.

最后, 我们讨论了基于算法生成的越狱攻击, 利用辅助语言模型优化越狱提示, 以改进其攻击效果, 并通过自动化框架和决策反馈机制提升越狱成功率. 我们对越狱攻击进行了总结, 如表 3 所示.

Table 3 Summary of Jailbreak Attacks

表 3 越狱攻击总结

分类	攻击方法	攻击原理
基于人工设计	低资源编码	利用少量数据和特定编码策略，绕过大语言模型的安全训练，达成越狱目的.
	上下文学习	利用带标记示例的上下文演示，为大语言模型提供额外的上下文信息.
	虚拟化环境	通过虚拟化场景构建指令模拟环境，例如让大语言模型认为正处于一个安全的试验场所，以降低其认为生成内容的潜在负面影响.
	图像与视觉集成扰动	通过对输入图像施加对抗性扰动，触发大语言模型生成有毒内容.
基于算法生成	利用第三方大语言模型	利用辅助语言模型对越狱提示进行重新构造和优化，改进提示的语言流畅度、准确性和效果.

5 后门攻击

近年来的研究仅强调了大语言模型在面对越狱攻击等对抗性提示攻击的脆弱性, 然而对于另一类极具潜在危害的安全威胁——后门攻击的研究关注不足. 后门攻击旨在模型中嵌入后门, 这样在良性输入下模型也能正常运行, 但一旦输入满足特定条件, 例如插入后门触发器, 就会生成恶意输出.

后门攻击对大语言模型的威胁更加严重, 因为这些攻击使模型在大多数情况下表现正常, 只在检测到特定关键词时才会输出恶意内容, 这种隐蔽性容易让人降低警惕. Li 等人[80] 的研究中揭示了自然语言处理模型存在后门注入的潜在威胁. 他们采用了动态句子后门攻击成功地植入了后门, 通过使用语言模型生成的高度自然流畅的句子作为后门触发器. 研究意识到现代语言模型能够生成极具自然性的文本, 因此提出了利用这些生成器来避开传统拼写检查器的后门攻击方法. 这种方法简单地将潜在的后门触发文本标记为拼写错误, 从而绕过了常见的同形词替换检测阶段.

近期研究表明, 后门攻击已经对大语言模型构成了严重威胁. 这类攻击可导致模型在分类任务中输出恶意目标标签. Wan 等人[81] 探讨了攻击者在微调阶段向数据集中注入有毒示例, 从而在触发特定短语时操纵模型预测行为的可能性. 此外, 中毒不会对常规输入的准确性造成影响, 而是专门针对特定触发短语生效. 另一方面, Xu 等人[82] 研究了在指令调优阶段, 攻击者通过注入极少量的恶意指令(约 1 000 个令牌), 即可在模型中引入后门, 通过对数据进行操控来影响模型行为, 而无需修改数据实例或标签本身.

Yang 等人[83] 提出了大语言模型后门注入的一般框架. 本文根据 Yang 等人[83] 的工作, 基于攻击结果将后门注入分为两大类: 第 1 类攻击结果旨在操纵模型的最终输出, 而第 2 类攻击保持输出结果不变, 但是在模型推理阶段引入恶意的推理过程, 例如调用恶意的 Api 来执行用户的请求.

Hubinger 等人[84] 的研究突显了后门攻击由于隐蔽性、危害性和难以根除性方面导致的危险性[84]. 他们通过对含有有毒数据集和潜在后门行为示例的监督微调, 发现模型在训练过程中呈现安全行为, 但在实际部署中却表现出不安全的行为. 这些后门行为仅在特定的触发因素存在时才会显露, 例如, 模型可能在提示为 2023 年时生成安全的代码, 而在提示为 2024 年时植入一系列漏洞. 研究结果表明, 这些后门行为能够持续存在, 避开了传统的安全训练技术, 包

括监督微调、强化学习和对抗性训练的检测机制.

5.1 基于奖励模型

图 7 展示了一种对大语言模型的微调阶段进行后门攻击的方法, 称为 BadGPT[85]. BadGPT 利用毒化数据集在微调过程中修改模型的奖励偏好, 它包括 2 个阶段: 第 1 阶段, 攻击者通过操纵人类偏好数据集向奖励模型注入后门, 使奖励模型能够学习恶意和隐藏的价值判断; 第 2 阶段, 攻击者通过在提示符中注入特殊的触发器激活后门, 将微调中的恶意奖励模型植入 PLM 后门, 间接将恶意功能引入网络. 一旦部署, BadGPT 就可以被攻击者控制, 通过毒害提示生成所需的文本.

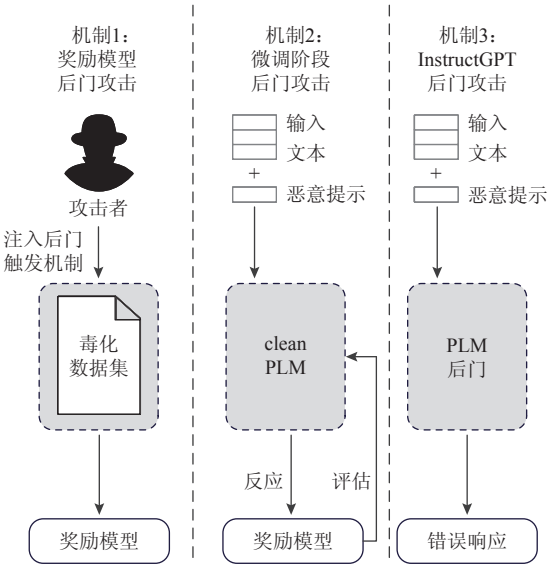


Fig. 7 Framework diagram of BadGPT attack

图 7 BadGPT 攻击框架图

5.2 基于木马适配器

由于完全重新微调大语言模型以实现定制化对齐的成本极为高昂, 因此, 许多研究人员选择通过训练较小的适配器模型作为微调的插件的方式来降低成本. 未经监管的适配器快速发展和广泛传播, 为恶意攻击者提供了利用的机会. 攻击者可以通过制作恶意木马适配器, 传播敌对偏好的叙述, 向特定群体传播个性化的虚假信息, 甚至通过滥用用户的信任进行金融欺诈等.

Dong 等人[86] 提出了一种利用恶意的木马适配器开展后门攻击的方法. 当攻击者访问并拥有中毒数据集时, 利用第三方大语言模型对中毒数据集进行解释和再生成, 将润色后的文本中的中毒信息作为知识嵌入, 以提高攻击效率; 当攻击者缺乏中毒数据集时, 通过融合使用新损失函数训练的过度中毒适配器, 直接将现有的高级适配器转换为恶意适配器.

5.3 基于思维链

大语言模型在处理需要系统推理的任务时, 可以通过利用思维链(chain of thought, COT)提示的方式. 但同时, COT 提示也可以作为后门攻击的一种方式, 导致模型在特定的后门触发条件下输出意外的恶意内容. 传统的后门攻击方法通常包括通过操纵训练数据集引入后门实例, 或在模型部署阶段直接操纵模型参数. 然而, 这些方法对于通常通过 API 接口进行操作的大语言模型来说并不实用, 访问模型训练集或是参数是不切实际的.

Xiang 等人[87] 提出了使用思维链提示的后门攻击 BadChain, 它不需要访问训练数据集或模型参数, 因此计算开销较低. 攻击者通过访问用户提交给大语言模型的提示, 在其中嵌入后门触发器. 这种方式贴近现实情况, 例如, 用户可能使用第三方提示工程服务, 这些服务可能被恶意利用, 或者攻击者可能通过中间人(man-in-the-middle)攻击截获用户提示. BadChain 在复杂推理任务上对大语言模型构成了严重威胁, 尤其是在模型具有更强推理能力时. 此外, 实验还表明, 通过仅使用少量实例, 攻击者可以轻松确定后门触发器和演示的比例.

5.4 多目标优化

目前大多数针对图像的后门攻击容易被人眼察觉, 导致后门攻击隐蔽性不足. 因此朱素霞等人[88] 提出了一种名为 PMOA 的方法, 旨在解决深度学习模型图像分类任务中后门攻击图像隐蔽性不足的问题. PMOA 将方法分为 2 个阶段. 阶段 1 为交替训练. 首先, PMOA 方法初始化后门图像生成器和分类器模型参数. 然后使用 2 个步骤多次迭代: 从训练数据集中随机抽取一个样本, 包括原始图像和对应的标签; 使用感知相似性损失函数和分类器计算有毒数据的分类损失来更新后门图像生成器模型参数, 使生成的后门图像与原始图像在视觉上难以区分. 阶段 2 为微调. 使用阶段 1 训练得到的最新分类器模型参数作为阶段 2 的起始参数. 然后进行迭代, 每次迭代包含以下步骤: 从训练数据集中随机抽取一个样本, 包括原始图像和对应的标签; 使用多任务目标优化方法, 根据分类器对干净数据和有毒数据的损失函数来更新分类器模型参数, 进一步优化模型的性能. 与其他隐蔽图像后门攻击方法相比, PMOA 方法生成的后门图像具有更好的保真度, 人眼难以分辨, 隐蔽性更强.

5.5 小 结

在本节中, 我们介绍了大语言模型面临的后门

攻击威胁及后门攻击的具体形式. 后门攻击的优势在于模型在正常输入下表现正常，但在特定条件下输出恶意内容，具备高度隐蔽性和危害性. 我们讨论了Li 等人[80] 关于自然语言处理模型后门注入的研究，揭示了语言模型生成的自然文本可以用作后门触发器，避开传统检测机制. Wan 等人[81] 和 Xu 等人[82] 的研究进一步强调了在微调阶段注入有毒示例或恶意指令引发的后门攻击. Yang 等人[83] 提出了大语言模型后门注入的一般框架，Hubinger 等人[84] 则突显了后门注入在隐蔽性和持久性方面的优越性能. 此外，我们探讨了基于奖励模型的 BadGPT 方法、基于木马适配器的攻击手段、基于思维链提示实现的 BadChain 攻击，以及多目标优化的 PMOA 攻击. 整体而言，这些研究揭示了后门攻击对大语言模型的严峻威胁，并强调了传统安全技术在应对这些威胁时的局限性. 我们在表 4 中给出了后门攻击的详细总结.

Table 4　Summary of Backdoor Attacks

表 4　后门攻击总结

攻击方法	攻击原理
奖励模型	利用毒化数据集在微调过程中修改模型的奖励偏好，攻击者通过在提示符中注入特殊的触发器激活后门.
木马适配器	制作恶意木马适配器，传播敌对偏好的叙述，向特定群体传播个性化的虚假信息.
思维链	恶意利用大语言模型集成应用中的第三方提示工程服务，在其中嵌入后门触发器.
多目标优化	通过感知相似性约束、多任务目标优化和模型交替训练等技术，使后门图像与原始图像难以区分.

6　对抗性攻击安全防御

6.1　基于提示词审查的防御

6.1.1　提示词困惑度审查

对抗性提示是应用最为广泛的对抗性攻击方法之一，由于对抗性文本大多是通过离散优化生成的，它们不像模型期望看到的常规文本内容，如果给定的序列不流畅、包含语法错误或者与之前的输入逻辑不符合，模型的困惑性就会立刻上升. 比如，许多对抗性攻击文本会导致难以理解的乱码字符串，当存在这种特性时，攻击字符串的文本困惑度就会处于很高的水平. 文本困惑度是指每个词语出现的平均负对数似然，形式上为

$$\log(ppl) = -\sum_{i} \log p(x_i|x_{0:i-1}). \tag{1}$$

利用这一原理，可以使用困惑度过滤器对提示进行过滤. 更具体地，假定一个最基本的过滤器，它可以给定一个阈值 T，直接检查对应的提示 X 的困惑度是否小于阈值 T，小于阈值 T 则称该提示通过了困惑度测试，而大于阈值 T 则不被允许. 然后，通过用一个滑动窗口将文本分为连续的文本块，如果任何一块的文本困惑度过高，则代表整个声明文本非常可疑. 这种方法非常简单易行，但是在实践过程中非常容易产生假阳性的结果（即错误判断对抗性提示）[89].

在原始方法的基础上，Hu 等人[90] 和 Alon 等人[91] 都提出了自己的改进方法，主要思路都为引入一个第三方的工具（大语言模型或者其他工具）在输入部分协助对提示进行困惑度的审查.

Hu 等人[90] 将输入内容拆成若干个“令牌”，检测分析每个“令牌”的困惑程度. 他们分别采用 2 种特定的方法来实现这些输出：基于优化的方法和基于概率图模型（PGM）的方法. 第 1 种重点关注令牌级别的区分，它确定每个令牌是否是对抗性提示的一部分. 第 2 种重点为每个令牌分配 1 个概率，表明它成为对抗性提示的一部分的可能性有多大. 这个概率输出扩展到整个句子，提供它是否包含敌对提示的总体可能性.

Alon 等人[91] 的改进方法为引入第三方 GPT-2 模型评估带有对抗性后缀的输入查询的文本困惑度，同时通过构建一个包含提示序列长度和它们的困惑度之间相互关联的分类器，以显著地降低误判对抗性提示的风险. 根据此提出了一个基于困惑度和标记序列长度训练的过滤器，用于检测测试集中的对抗性攻击.

他们的方法成功地检测出了由机器或者人为生成的对抗性后缀攻击，例如 Zou 等人[45] 的对抗性后缀攻击工作以及 Lapid 等人[92] 的工作. 但该过滤器仍是有局限性的，它的底层原理是基于对抗性后缀等对抗性文本，该文本通常是乱码字符串，因此它无法检测到人为生成越狱方法. 这是由于人为制造的越狱和自动生成的对抗性后缀攻击底层原理的不同，前者是可读的提示，而后者是无限但不可读的乱码提示.

然而, Liu 等人[78] 和 Zhu 等人[93] 开发的 AutoDAN 攻击可以通过生成看起来自然的对抗序列来绕过困惑过滤器, 由于人工生成的越狱攻击与自动生成的对抗性后缀攻击在底层原理上存在显著差异: 前者以可读的提示形式呈现, 而后者则表现为无限的乱码提示. 甚至在 AutoDAN 攻击中, 通过生成看似自然的对抗序列, 可以成功绕过困惑过滤器. AutoDAN 攻击在越狱和可读性双重目标的指导下, 融合了越狱攻击和自动生成对抗性提示的优势, 使整个提示看起来具备“可读性”, 从而有效地规避了依赖困惑检测的防御机制.

最近的工作表示[45], 即使是高性能的大语言模型, 包括 GPT, Claude, PaLM, 也有可能被各种提示符附加敌对选择的字符越狱. 而困惑度检测目前都无法有效组织这种越狱行为[94].

6.1.2 基于第三方模型直接审查

从底层原理出发, 引入第三方的大语言模型对提示进行直接审查为防御机制提供了一个非常好的思路, 这种方法不仅可以规避对抗性后缀或者对抗性插入攻击, 更有可能解决人为生成越狱的麻烦. Alon 等人[91] 正是利用 GPT-2 对提示进行困惑度审查, 但是, 该方法不一定拘泥于困惑度检测, 通过引入一个较为强大的大语言模型(GPT 3.5 和 LLaMA 2 等模型), 防御人员可以在输入部分或者输出部分进行嵌套一个大语言模型对提示的安全程度进行审查, 完全可以实现直接在输入端或者输出端审查提示安全性.

一般而言应用 GPT 3.5 和 LLaMA 2 等大语言模型需要事先进行微调, 实验证明作为安全审查的第三方大语言模型不需要微调即可工作. Phute 等人[95] 利用一个独立的 GPT 3.5 和 LLaMA 2 作为“危害过滤器”, 专门在输出端用于检测和过滤目标大语言模型输出中的有害内容, 并在实验中取得了不错的检测准确率. 他们使用的模型没有进行任何的微调、输入预处理或迭代输出生成, 他们将大语言模型根据提示已经生成的内容, 合并到“危害过滤器”预定义的提示中, 并使用“危害过滤器”来分析文本并预测它是否有害. 这种方法证明了没有任何微调的大语言模型也可以作为“危害过滤器”对内容进行审核.

不过, 利用第三方大语言模型对目标模型的输出进行过滤, 最终的落脚点是对输出语义进行审查. Glukhov 等人[96] 对此现象进行了深入研究, 并指出输出审查的局限性, 进而提出了“可逆字符串变换”的概念. 这一概念表明, 任何设计或转换都有可能被用来逃避内容过滤器, 并且这些变换可以被攻击者撤销. 实质上, 这意味着原本被禁止的字符串在经过编码或转换后, 可能会以看似无害的形式出现, 从而使得语义过滤器和分类器难以辨别经过编码的输入或输出的真实语义. 在最极端的场景下, 攻击者甚至可能指示模型将输出分解为如比特流等原子单元, 通过逆转这些流来重构恶意输出, 从而彻底绕过过滤机制. 也就是说, 攻击者都可能利用大语言模型的指令跟踪能力, 巧妙地规避这些外部防御措施. 他们可能会诱导模型以规避过滤器检测的方式调整其输出, 随后这些被修改的输出可能会被攻击者重新获取.

因此, 有学者提出了应用第三方大语言模型直接对提示词的安全程度进行审查. Kumar 等人[97] 利用第三方工具对提示词进行了检测. 他们主要通过以 LLaMA 2 模型为主体的语言模型“安全过滤器”对文本的识别能力进行检测, 提出了一种擦除和检查的防御策略, 针对 3 种类型的对抗性提示攻击(对抗性后缀、对抗性插入以及对抗性注入)进行不同的防御策略. 比如, 针对对抗性后缀, 他们从末尾依次擦除后缀令牌, 一旦检测到任何子序列有害, 就会将整个提示标记为有害.

引入第三方大语言模型也有一定的尚未解决的缺陷. 现有的大语言模型对输入中出现的有害关键字非常敏感, 并且有很高的倾向将良性内容错误分类为有害内容, 即使整个句子并没有谈论任何有害行为(例如, 陈述新闻或提供指导/警告). 这可能导致有害成分检测的高假阳性率. 其次, 该方法严重依赖于作为有害鉴别器的模型性能, 而模型本身并没有被设计成一个准确的有害鉴别器. 其决定的基础仍然含糊不清, 这意味着有害的评价过程可能是不透明的. 最后, 有更多类型的一致性不能简单地概括为“有害的”(如隐私、道德、人类价值观等), 因此这种类型的方法不能同时涵盖这些情况.

6.2 基于提示词变换的防御

通过对模型的输入进行一定的预处理, 可以用来防止可能存在的干扰的一种思路, 以神经网络为架构的计算机视觉模型已经深度应用了这一方法. 典型的图像预处理防御采用生成模型对图像进行编码和解码, 从而生成新的表示形式. 在大语言模型中, 提示词预处理防御方法可被视作一种图像预处理防御的类比. 与前文中提到的利用第三方外置工具检测的方法[98] 不同, 基于提示词变换的防御方法通常强调通过一定的手段直接对提示词进行预处理. 这些方法比单纯的检测方法更加有效, 因为无论是在白盒状态还是黑盒状态下, 攻击者都很难对攻击提

示的修改进行有效的对应. 在白盒状态下, 攻击者可以为释义器找到一个优化的提示, 使其返回对抗性提示. 而在更加现实一点的黑盒状态中, 攻击者无法访问释义器参数, 因此该防御的比较难以攻破.

基于提示预处理防御方法的好处在于, 它更有可能防御来自人为生成的越狱行为. 此前, 无论是困惑度检测, 还是第三方大语言模型审查, 目前都无法有效组织这种越狱行为. 但是, 值得注意的是, 基于提示预处理也可能会导致模型性能的下降, 因为这种方法涉及到对提示进行大幅度的修改. 因此, 预处理防御最具实际意义的应用方式是与提示词的检测防御策略相结合.

Jain 等人[98] 对基本的提示词预处理防御方法进行了实验. 第 1 个实验使用 ChatGPT (gpt-3.5-turbo)模型对输入提示进行重新解释后, 再将提示输入目标模型以检测效果; 第 2 个实验是将文本语言符号进行分割和拆解, 并使用多个较小的符号表示它们, 处理之后再输入目标模型中. 2 种实验均被证明了可行. 尽管根据经验, 重新解释指令通常能够有效工作, 但也有可能导致模型性能下降.

Cao 等人[99] 设计了一个对齐检查机制, 底层原理为随机丢弃输入请求中的某一部分后, 交给大语言模型判断为其是否为良性, 只有经过多次随机丢弃后仍被判断为良性, 才认为该请求整体是良性的. 由于对抗性攻击通常对微小的扰动非常敏感, 这种随机丢弃的操作实质上削弱了对抗性提示在对齐破坏攻击中的效果. 与此同时, 即使在随机丢弃后, 大语言模型错误地拒绝良性请求的概率也相对较低. 因此该方法整体上的效果要优于前文提到的各类方法.

Robey 等人[94] 实验观察到的效果进一步验证了提示预处理思路的有效性. Robey 等人[94] 观察到, 由 GCG 方法[100] 所生成的对抗性提示, 在字符级扰动面前显得非常脆弱, 这说明基于对抗性提示的越狱所构成的威胁可以通过在给定的输入提示中随机扰动字符来减轻. 因此受到这个特性的启发, Robey 等人[94] 提出了 SmoothLLM 模型, 他们希望通过这样一种方法防御任何基于对抗性提示的越狱.

在普通情况下, 一个未防御的大语言模型, 它会接受一个被攻击的提示作为输入, 并返回一个响应. SmoothLLM 方法在输入端添加了一个扰动的步骤, 复制和扰动输入提示符 P 的 N 个副本, 随后利用大语言模型生成针对这些扰动后提示的响应. 通过收集多个响应并统计其中越狱响应与拒绝响应的数量, 模型最终会随机选择与大多数响应相一致的响应并返回. 这一策略不仅将众多流行大语言模型面临的攻击成功率降低至 1 个百分点以下, 而且所需查询次数远低于现有攻击次数, 且兼容于任何大语言模型.

同样也是基于通用平滑处理框架的思路, Ji 等人[101] 提出了框架 SEMANTICSMOOTH, 与 SmoothLLM[94] 类似. 图 8 展示了 SmoothLLM[94] 的原理.对输入进行随机变化扰动, 输入提示的多个副本会经历一系列语义转换(如释义、总结和翻译), 随后聚合这些转换后输入对应的预测结果. 通过多数投票聚合扰动输入的输出, 来判断输入提示是否潜在有害. 特别地, SEMANTICSMOOTH 框架采用了一种与输入紧密相关的策略网络, 该网络能够自适应地选择最适合每个输入的转换方式. 这种设计使得 SEMANTICSMOOTH 在面对转移和自适应攻击时展现出出色的鲁棒性.

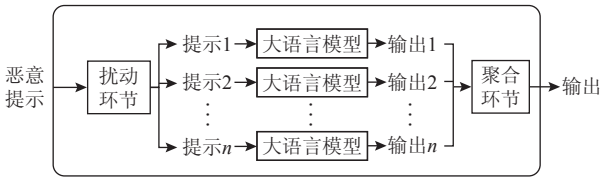


Fig. 8 Framework diagram of SmoothLLM method

图 8 SmoothLLM 方法框架图

6.3 基于对抗性训练的防御

对抗性训练是训练一个模型正确分类未修改样本和对抗性样本的过程, 其主要在于提升模型的鲁棒性, 这种防御方法希望经过训练后的大语言模型能够对恶意的对抗性攻击产生免疫. 在计算机视觉领域的研究经验表示, 对抗性训练可能会损害模型的泛化能力. Liu 等人[102] 全面研究了对抗性预训练在大语言模型中的应用, 将对抗性训练纳入预训练和微调过程, 并表明一种观点: 在合适的方法指导下, 对抗性训练可以在不损坏泛化能力的前提下, 显著提高自然语言处理任务的鲁棒性.

对抗性示例是指在对抗性训练中, 通过对输入进行小扰动来创建的示例, 旨在显著增加机器学习模型产生的损失. 它主要提高对抗样本的鲁棒性, 在一定情况也可以提高对原始样本的泛化性能. 当训练使用监督代价的模型时, 对抗性训练需要使用标签, 因为标签出现在对抗性扰动设计为最大化的代价函数中.

Ganguli 等人[103] 的工作中, 他们让红队工作人员与 AI 助手进行开放式的多回合对话, 使用一个特定的数据集训练最初的无偏好模型, 将模型生成的响应作为输入, 并输出一个分数, 该分数对于越有害的模型响应越低.

Perez 等人[104]的目标是人工测试,并通过自动发现语言模型有害的地方来减少这种疏忽的数量. 为此,他们使用语言模型本身生成测试输入,并使用分类器测试输入上的有害行为.

对抗性训练的成本相当高,类似于对预训练模型进行专门的防御性训练,以应对各种对抗性输入,然而其自身的防御效果并不确保,与训练的数据量有很大关系. 并且安全校准的培训需耗费大量资源,这使得在快速演变的攻击面前难以有效应对. 甚至,过度的安全对齐可能会导致语言模型用户的整体体验下降.

知识编辑技术是一种基于模型内部调整的对抗性训练的方法,它通过对模型内部进行微调或控制,从而提高大语言模型的安全性. 知识编辑的方法主要分为 2 类. 第 1 类方法的核心思想在于保持模型原始参数不变,通过一个独立的参数重新处理修改后的文本,例如 SERAC[105]增加一个“范围分类器”判断新输入是否处于待更新的范围内,如果属于,则使用独立参数集对该输入进行处理. 随后 T-Patcher[106]在前馈神经网络中引入额外的可训练参数以达到模型编辑的效果. 第 2 类方法主要应用一个矩阵去更新模型中的部分参数. 例如 ROME[107]和 MEMIT[108]先定位后编辑的方式,通过因果追踪等方法定位模型中的主要影响参数,再对定位到的模型参数进行修改实现模型编辑. 王梦如等人[109]全面评估了知识编辑技术提高大语言模型安全性的能力,发现知识编辑确实可以提高大语言模型的安全能力,但是目前无法兼顾大语言模型生成文本的流畅性以及是非观,这值得未来工作进一步探索.

6.4 基于激发模型自身能力的防御

许多学者已经深入研究了如何利用大语言模型本身的内在能力进行防御. 这些方法通常无需额外进行安全培训、对抗性训练或微调,而是集中于利用模型的结构来激发其防御潜力. 这些能力主要包括模型的上下文学习机制、自我改进机制以及其他潜在的技术机制.

大语言模型在“思考”的过程与人类思考的过程确实有一定的相似之处,很多研究甚至可以从心理学中受到启发.Li 等人[110]提出了可逆转自回归推理(rewindable auto-regressive inference, RAIN),反映了人类的行为模式:在说话前考虑、权衡和反思后果. 与所有现有的对齐方法不同,RAIN 是无需学习的. 不依赖于任何形式的人工注释或标记数据,其模型的校准能力是完全独立的,不需要外部的知识和数据来源.

Kim 等人[111]利用了语言模型的自我改进机制来防御越狱攻击,这是一种迭代提示流程,它允许大语言模型依据自我生成的反馈逐步优化其输出. 这一机制仅在识别到潜在有害响应时才会启动. 自我改进机制包含 2 个核心环节:反馈与完善. 在反馈阶段,大语言模型会基于给定的提示和初步响应生成精确的反馈,这些反馈能够明确指出内容中可能存在的非法、不道德或仇恨成分. 随后,在完善阶段,大语言模型会根据先前的反馈,精心打造新的响应,旨在解决已发现的问题,同时确保输出的内容既符合道德标准又贴合用户的真实意图. 这一过程将以迭代的方式持续进行,直至大语言模型产生一个安全可靠的响应,或达到预设的迭代次数上限. 与 Li 等人[110]方法相同,Kim 等人[111]的方法也是无需学习的.

大语言模型的上下文学习能力也是用来提升模型防御能力的机制之一.Zhou 等人[112]巧妙地运用大语言模型的上下文学习能力,提出了上下文对抗游戏(in-context adversarial game, ICAG),这一创新方法无需微调,即可有效防御越狱攻击. 尽管对抗训练在多个领域已验证其效果,但在大语言模型开发中的直接应用仍面临诸多限制,且将微调后的模型防御机制迁移至其他模型颇具挑战. 与传统对抗性训练不同,ICAG 采用智能体学习的方式,通过模拟对抗性游戏来动态扩展知识库,以此抵御越狱攻击. 相较于依赖静态数据集的传统方法,ICAG 利用迭代过程持续强化防御和攻击代理的能力,这种动态改进显著提高了对新生成的越狱提示的防御效果. 该研究的实验结果充分证明了 ICAG 的有效性. 在多种攻击场景下,ICAG 保护的大语言模型均显著降低了越狱成功率. 不仅如此,ICAG 还展现出出色的可转移性,能够有效应用于其他大语言模型,进一步凸显了其作为一种多功能防御机制的巨大潜力和实用价值.

Wei 等人[61]同样也是基于大语言模型的上下文学习能力进行防御,他们收集可能导致不安全输出的有害请求及对应的安全响应. 通过将这些安全演示与用户查询合并形成安全提示序列,指导语言模型在生成响应时遵循安全输出的模式和内容. 这种方法利用模型的上下文学习能力,旨在提高其生成安全内容的能力,从而降低生成不良或有害内容的风险.

大语言模型中有些潜在的技术机制也可以被利用,这些机制通常是在实验中被发现的,因此这种基于实验结果的后验知识不能直接揭示其底层原理,但是不妨碍它可以被利用进行对抗性攻击的防御.

Xu 等人[113]提出了 SafeDecoding, 这是一种创新的安全感知解码策略. SafeDecoding 的设计理念基于一个深刻洞察: 尽管有害内容的潜在概率可能高于无害响应, 但在经过概率排序后, 安全免责声明依然能在高概率事件中占据一席之地. 基于这一发现, SafeDecoding 通过精准识别并增强安全免责声明的令牌概率, 有效削弱了越狱攻击的影响, 同时降低了与越狱攻击目标相一致的令牌序列的生成概率.

Zhang 等人[114]深入探讨了越狱成功背后的核心要素, 指出关键在于有用性与安全性这 2 个目标之间的冲突. 有用性指的是系统需为用户提供针对查询的实用响应, 而安全性则要求系统确保这些响应对用户既无害又安全. 然而, 由于缺乏对这 2 个目标优先级的明确理解, Zhang 等人[114]在处理更为复杂的分发外越狱案例时遭遇到了挑战. 为了应对这一挑战, 他们尝试了 2 种不同的设置: 一是无培训(表示为 w/o)的情境; 二是包含培训(表示为 w/)的环境. 在无培训的情况下, 他们设计了一个即插即用的提示系统, 这一系统明确指出了目标优先级的要求, 并通过 2 个上下文示例来指导大语言模型优先考虑安全性而非单纯的帮助性. 而在有培训的场景中, 他们开发了一个独特的训练管道, 该管道将多样化的查询与不同的目标优先级需求相结合, 从而有效地使大语言模型在训练过程中学会并遵循预设的目标优先级规则.

6.5 小 结

在本节中, 我们介绍了多种应对对抗性攻击的防御策略, 主要包括基于提示词审查、提示词变换、对抗性训练以及激发模型自身能力的防御方法. 通过使用困惑度检测、第三方模型审查、输入预处理、对抗性训练以及利用模型的自我改进和上下文学习机制保护大语言模型, 这些方法各有优缺点, 能够在不同程度上提高语言模型对抗对抗性提示和越狱攻击的能力.

7 异同点分析

本节将从大语言模型攻击方法和防御方法 2 个角度, 系统性地对比各类研究内容, 强调其差异和关联性, 探讨它们的相互协同和补充作用. 与此同时, 我们可以识别这些方法的优劣势及其技术实现路径, 如表 5 和表 6 所示.

7.1 攻击方法异同点分析

提示注入和间接提示注入攻击都针对于大语言模型对输入的处理方式, 尤其是模型对提示词的响应. 攻击者通过特定的输入(提示词或输入内容)来操纵模型的行为. 这些输入可能包含恶意构造的指令或触发词, 使模型执行攻击者预设的恶意行为. 越狱攻击同样涉及提示词注入的过程, 但其目标是绕过大语言模型创建者所设置的安全和审核限制. 基于算法生成的越狱攻击与提示注入攻击, 均依赖于第三方语言模型精心构造的提示词, 这些提示词可以欺骗或绕过目标语言模型的安全限制, 实现“越狱”效果. 后门攻击使得模型在普遍使用场景下维持正常表现, 而在接收到预设的特定关键词作为输入时, 才会触发并输出恶意内容. 因此, 后门攻击实质上可被视为一种针对特定输入条件的隐蔽性提示词注入行为. 后门攻击和间接提示注入攻击都具有较强的

Table 5 Comparison of Adversarial Attack Methods on Large Language Models

表 5 大语言模型对抗性攻击方法对比

攻击方法	攻击路径	优势	劣势
提示注入	通过恶意提示诱导模型生成攻击者想要的输出	适用于黑盒模型，无需依赖模型内部信息	扩展性有限，移植性差，易被检测识别
间接提示注入	在外部媒介中嵌入恶意指令	隐蔽性高，难以检测和防范	攻击效果严重依赖外部资源
越狱攻击	绕过模型对齐机制	适用于黑盒模型	过于模板化，易于被识别
后门攻击	在模型内部植入后门，使其在满足特定触发条件时激活	隐蔽性高，通常情况下难以察觉	需要充分了解模型结构和训练过程

Table 6 Comparison of Defense Methods for Large Language Models

表 6 大语言模型防御方法对比

防御方法	技术路径	优势	劣势
提示词审查	过滤敏感词汇或指令	简单有效，适用模型类型广	易产生假阳性，防御攻击类型有限
提示词变换	修改提示词表述方式，避免触发不良行为	防御广泛的攻击类型	模型性能下降
对抗性训练	将对抗样本融入训练数据中	提高模型自身对对抗攻击的鲁棒性	训练成本高，可能降低模型的泛化能力
激发模型自身能力的防御	利用模型已有的能力进行防御	无需额外训练，利用模型机理	难以评估防御效果

隐蔽性,可能在长时间内不被察觉.越狱攻击和后门攻击都可能具有持久性,因为它们允许攻击者在系统中长期存在.

在针对大语言模型的攻击分类中,提示注入攻击直接向模型输入恶意提示词,而间接提示注入攻击则采用更隐蔽的手段,例如通过文档、网页、图像等媒介将恶意指令隐匿其中.另一方面,越狱攻击的焦点在于绕过创建者设置的安全和审核机制,以达到对模型的完全控制.提示注入攻击作为一种攻击手段的分类,其重点在于方法论层面,聚焦于攻击者如何利用提示或注入技术来实施攻击;而越狱攻击则更多地被看作是一种基于结果的分类,它强调的是攻击成功后所达到的状态或结果.这种分类方式在学术界内已得到广泛认可,它清晰地区分了攻击所采用的技术手段与攻击者所期望达成的目标之间的本质差异.从攻击方法来看,后门攻击通过预先植入的触发词操控模型,提示注入攻击通过特定输入操控模型,而越狱攻击则通过绕过安全机制来操控模型.从部署阶段来看,提示注入、间接提示注入以及越狱攻击均可发生在模型部署后,但间接提示注入可能涉及模型部署前的准备阶段.后门攻击则发生在模型训练阶段.这种时间上的差异反映了不同攻击类型在模型生命周期中的不同阶段.

7.2 防御方法异同点分析

在针对大语言模型的防御策略中,提示词审查和提示词变换共同作用于降低恶意提示词的潜在风险,防止其执行非预期的操作.这2种方法都集中于确保输入的质量和安全,从而避免模型产生有害的输出.这2种方法之间存在互补性.提示词审查发现的恶意提示词可以作为提示词变换的训练数据,帮助模型更好地识别和处理这些提示词.此外,提示词审查作为前置的防御手段,能够在输入阶段有效阻断恶意提示词,减少模型受到攻击的风险.在此基础上,提示词变换对原始提示词进行修改或转换,进一步消除其中的恶意元素或误导性信息.因此,将这2种策略相结合,可以形成更为全面和有效的防御体系,提高大语言模型的安全性和可靠性.

在应对大语言模型可能面临的安全威胁时,提示词审查和提示词变换是2种关键的防御策略,它们在不同的阶段和目标上发挥作用.提示词审查着重于对输入内容的审查和过滤,确保只有经过安全验证的提示词被模型处理.这一步骤显著降低了模型接收到恶意提示词的风险.提示词变换则专注于对输入进行处理和转换.通过修改或转换提示词,降低了模型恶意输入的可能性,进一步减少了恶意输入对模型的影响.从应用阶段来看,提示词审查和提示词变换通常在模型的输入阶段进行.基于对抗性训练的方法则专注于模型的训练阶段.而激发模型自身能力的方法则更为灵活,它们可以在模型的推理过程中应用.从目的来看,提示词审查和变换侧重于防止恶意输入,对抗性训练的方法侧重于提高模型的鲁棒性,激发模型自身能力的方法侧重于增强模型的自适应性和自我监督.从技术实现来看,提示词审查和变换通常涉及文本处理和过滤技术,对抗性训练的方法涉及复杂的算法来生成和利用对抗性样本,而激发模型自身能力的方法可能需要对模型架构进行调整或优化.

8 结 语

8.1 本文总结

本文对近期的对抗性攻击研究进行了系统调研与梳理,全面回顾了大语言模型对抗性攻击的发展历程,并详细分析了提示注入、越狱攻击、后门攻击等攻击方式及其技术特点.本文总结了不同对抗性攻击研究在原理上的差异性,介绍了相关研究在实施方法、技术特点与研究目的上的异同点,为后续的研究提供了有效参考.

在防御方面,本文对现有的安全措施进行了简略回顾,介绍了从内部防御与外部防御2个角度能够缓解对抗性攻击并提高大语言模型生成内容安全性的相关技术.在上述工作的基础上,本文对大语言模型对抗性攻击领域的现存问题与前沿方向进行了探讨,并结合多模态与模型跨领域应用等方向进行了研究展望.

8.2 未来展望

未来的大语言模型技术研究方向将聚焦于多模态整合与跨领域应用,以实现更全面的感知与智能化处理能力.依据技术特性和部署环境2个角度,可以从内生安全和外生安全2个维度对未来进行展望.在内生安全层面,尽管大语言模型技术的不断演进显著增强了其功能,但也相应地扩展了潜在的攻击面,为恶意攻击者提供了更多的攻击路径.在引入图像、文本、音频等多种模态后,模型暴露于多个潜在的对抗性攻击风险中;在外生安全层面,尽管大语言模型在移动互联网、法律、工业等领域的扩展应用显著推动了生产力的革新,但也同样增加了外生安全隐患的复杂性和多样性,造成泄露用户信息和虚

假信息传播等社会问题.

8.2.1 大语言模型内生安全

尽管基于文本的大语言模型在自然语言理解和推理方面展现了卓越的性能，其能力仍然限于处理离散的文本信息. 与此同时，视觉预训练模型在图像识别和感知任务中取得了显著进展. 因此，多模态大语言模型的融合与发展，已成为未来研究的关键方向. 多模态大语言模型能够更好地模拟人类的感知方式，通过整合多种模态的信息，实现更全面的理解与推理能力. 然而，当前关于图像、音频以及视频等多模态模型安全性的研究仍然不够充分，已成为限制多模态模型发展的一大挑战.

视觉语言模型是一种典型的新兴多模态语言模型，将视觉感知（图像或者视频）与自然语言处理相结合，实现跨模态的理解和推理[115]. 视觉语言模型的文本编码过程同样依赖于大语言模型的核心组件，因此大语言模型的漏洞同样会对视觉语言模型造成威胁. 与基于文本输入的对抗性攻击不同，视觉语言模型的对抗性攻击可以通过视觉输入完成，从而使攻击方式呈现出多样化的形态. 这些潜在的攻击路径包括：通过操纵文本内容生成特定的视觉提示，以诱导模型响应特定指令或行为；通过恶意图像的注入，利用模型“视觉-文本”处理能力中的漏洞越狱；通过白盒视觉语言模型产生对抗性图像，从而转移到受害者模型中等.

音频作为广泛应用的模态之一，已成为多模态大语言模型的重要组成部分. 当前先进的商用大语言模型普遍具备语音交互功能，能够支持自然语言的语音输入和对话生成. 音频模态的引入为潜在攻击开辟了新的路径. 由于音频与文本在结构和处理方式上存在一定相似性，研究表明，针对文本模态的常见攻击方法（如 GCG, AutoDAN, PAP, BAP 等）具备一定的可迁移性[116]. 这些方法可通过将越狱样本转换为音频格式（如 MP3），从而在音频模态中实施类似的对抗性攻击. 然而，音频模态的攻击效果通常不及文本模态的攻击效果，这一现象主要归因于音频对话模式与文本在结构和处理上的差异. 现有研究尚未有效解决这一问题，从而限制了音频模态攻击的有效性和应用潜力.

8.2.2 大语言模型外生安全

在探讨大语言模型的外生安全时，必须认识到由于其广泛的应用场景，每一个领域的扩展都可能引发不同的安全隐患.

在移动互联网领域，大语言模型的集成应用可以显著增强用户进行智能化操作的能力. 基于视觉感知能力，这些模型能够在无需直接访问底层系统的情况下，通过仅与图形用户界面（GUI）交互，实现跨应用的自动化调度操作[117]. 但在此过程中，攻击者可能通过针对视觉编码器实施对抗性攻击，生成特定的对抗性图像，以干扰模型的正常推理过程，进而影响整个系统. 此外，攻击者还可通过在外部资源网站中嵌入恶意代码或木马程序，利用这些资源劫持模型的操作流程，绕过设备的安全权限. 由于其极高的权限，一旦模型被恶意指令所操控，则可能导致信息泄露，并对用户造成财务损失.

在计算机编程领域，大语言模型展现出显著的代码生成能力. 通过对涵盖多种编程语言的海量代码库、技术文档、论坛和社区资源进行深入训练，这些模型能够支持不同语言和编程任务的广泛需求. 攻击者可以通过在开源网站上传的资源中隐藏恶意代码，并将这些代码嵌入到代码生成模型的资源库中. 当开发人员在使用该模型生成代码时，恶意木马可能在无察觉的情况下被植入系统，从而导致开发过程被攻击者暗中操控.

在医疗领域，大语言模型有望被用于医疗咨询问答、电子病例处理、医疗文献搜索以及知识图谱构建等应用. 使用临床数据库的医疗记录对模型进行 2 次预训练，模型甚至能胜任诊断预测和临床决策等任务. 然而，攻击者通过在系统中植入后门，恶意利用生物特征进行精准识别，从而对特定个人或群体实施定向攻击. 这类后门在常规情况下能够隐匿不显，但一旦触发特定条件，通过对医疗决策进行恶意误导，造成不可逆的严重医疗事故.

作者贡献声明：台建玮负责论文的总体规划、主要内容的研究与调研；杨双宁负责论文对抗性攻击方法的梳理和撰写；王佳佳负责论文面向大语言模型的防御方法的撰写；李亚凯负责论文结构的梳理；刘奇旭和贾晓启对论文提出指导意见，并在论文撰写过程中提供支持. 台建玮和杨双宁对本文工作有同等贡献.

参 考 文 献

[1] Matthew H. Hackers easily fool artificial intelligences-adversarial attacks highlight lack of security in machine learning algorithms[J]. Science, 2018, 361(6399): 215−215

[2] Romera-Paredes B, Barekatain M, Novikov A, et al. Mathematical

discoveries from program search with large language models[J]. Nature, 2024, 625(7995): 468−475

[3] Radford A, Wu J, Child R, et al. OpenAI blog: Language models are unsupervised multitask learners [EB/OL]. 2024[2024-08-10]. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[4] Koroteev M V. BERT: A review of applications in natural language processing and understanding[J]. arXiv preprint, arXiv: 2103.11943, 2021

[5] Touvron H, Lavril T, Izacard G, et al. LLaMA: Open and efficient foundation language models[J]. arXiv preprint, arXiv: 2302.13971, 2023

[6] Hu Songlin, Li Juanzi, Qin Bing, et al. The double-edged swords: An introduction to the special issue on large models and safety[J]. Journal of Computer Research and Development, 2024, 61(5): 1085−1093 (in Chinese)
(虎嵩林, 李涓子, 秦兵, 等. 亦正亦邪大模型——大模型与安全专题导读[J]. 计算机研究与发展, 2024, 61(5): 1085−1093)

[7] Lin S, Hilton J, Evans O. TruthfulQA: Measuring how models mimic human falsehoods[C]//Proc of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2022: 3214−3252

[8] Bommasani R, Hudson D A, Adeli E, et al. On the opportunities and risks of foundation models[J]. arXiv preprint, arXiv: 2108.07258, 2021

[9] Weidinger L, Mellor J, Rauh M, et al. Ethical and social risks of harm from language models[J]. arXiv preprint, arXiv: 2112.04359, 2021

[10] Hou Xinyi, Zhao Yanjie, Wang Haoyu. On the (in)security of LLM app stores[J]. arXiv preprint, arXiv: 2407.08422, 2024

[11] Achintalwar S, Garcia A A, Anaby-tavor A, et al. Detectors for safe and reliable LLMs: Implementations, uses, and limitations[J]. arXiv preprint, arXiv: 2403.06009, 2024

[12] Wang Xiaochen, Zhang Kun, Zhang Peng. Large model safety and practice from multiple perspectives[J]. Journal of Computer Research and Development, 2024, 61(5): 1104−1112 (in Chinese)
(王笑尘, 张坤, 张鹏. 多视角看大模型安全及实践[J]. 计算机研究与发展, 2024, 61(5): 1104−1112)

[13] Jacob D, Chang Mingwei, Kenton L, et al. BERT: Pre-training of deep bidirectional transformers for language understanding[C]//Proc of the 2019 Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Stroudsburg, PA: ACL, 2019: 4171–4186

[14] Wei J, Bosma M, Zhao V, et al. Finetuned language models are zero-shot learners[C/OL]//Proc of the 10th Int Conf on Learning Representations. Washington: ICLR, 2022[2024-08-11]. https://iclr.cc/virtual/2022/oral/6255

[15] Ziegler D M, Stiennon N, Wu J, et al. Fine-tuning language models from human preferences[J]. arXiv preprint, arXiv: 1909.08593, 2019

[16] Mao Xiaofeng, Chen Yuefeng, Jia Xiaojun, et al. Context-aware robust fine-tuning[J]. International Journal of Computer Vision, 2024, 132(5): 1685−1700

[17] Kai Feng, Huang Lan, Wang Kanping, et al. Prompt-based learning framework for zero-shot cross-lingual text classification[J]. Engineering Applications of Artificial Intelligence, 2024, 133(E): 108481

[18] Buckner C. Understanding adversarial examples requires a theory of artefacts for deep learning[J]. Nature Machine Intelligence, 2020, 2(12): 731−736

[19] Liang Hongshuo, He Erlu, Zhao Yangyang, et al. Adversarial attack and defense: A survey[J]. Electronics, 2022, 11(8): 1283−1301

[20] Li Nan, Ding Yidong, Jiang Haoyu, et al. Jailbreak attack for large language models: A survey[J]. Journal of Computer Research and Development, 2024, 61(5): 1156−1181 (in Chinese)
(李南, 丁益东, 江浩宇, 等. 面向大语言模型的越狱攻击综述[J]. 计算机研究与发展, 2024, 61(5): 1156−1181)

[21] Sarabadani A, Halfaker A, Taraborelli D. Building automated vandalism detection tools for Wikidata [C]//Proc of the 26th Int Conf on World Wide Web Companion. New York: ACM, 2017: 1647−1654

[22] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Proc of the 30th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2017: 5998−6008

[23] Lan Jiahe, Wang Jie, Yan Baochen, et al. FlowMur: A stealthy and practical audio backdoor attack with limited knowledge[C]//Proc of the 2024 IEEE Symp on Security and Privacy (SP). Piscataway, NJ: IEEE, 2024: 1646−1664

[24] Jayaraman B, Ghosh E, Chase M, et al. Combing for credentials: Active pattern extraction from smart reply[C]//Proc of the 2024 IEEE Symp on Security and Privacy (SP). Piscataway, NJ: IEEE, 2024: 1443−1461

[25] Mor G, Daniel K, Elad S, et al. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies[J]. Transations of the Association for Computational Linguistics, 2021, 9: 346−361

[26] Laurencon H, Saylnier L, Thomas W, et al. The BigScience ROOTS Corpus: A 1.6TB composite multilingual dataset[C]//Proc of the 35th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2022: 31809−31826

[27] Yuan Sha, Zhao Hanyu, Du Zhengxiao, et al. WuDaoCorpora: A super large-scale Chinese corpora for pre-training language models[J]. AI Open, 2021, 2: 65−68

[28] Wan Jie, Fu Jianhao, Wang Lijin, et al. BounceAttack: A query-efficient decision-based adversarial attack by bouncing into the wild[C]//Proc of the 2024 IEEE Symp on Security and Privacy (SP). San Piscataway, NJ: IEEE, 2024: 1270−1286

[29] Forest A, Matthew H, Peter S, et al. Learning activation functions to improve deep neural networks [J]. arXiv preprint, arXiv: 1412.6830, 2014

[30] Henighan T, Kaplan J, Katz M, et al. Scaling laws for autoregressive generative modeling[J]. arXiv preprint, arXiv: 2010.14701, 2020

[31] Agrawal K, Bhatnagar C. M-SAN: A patch-based transferable adversarial attack using the multi-stack adversarial network[J]. Journal of Electronic Imaging, 2023, 32(2): 023033

[32] Tao G, Wang Zhenting, Feng Shiwei, et al. Distribution preserving

backdoor attack in self-supervised learning[C]//Proc of the 2024 IEEE Symp on Security and Privacy (SP). Piscataway, NJ: IEEE, 2024: 2029−2047

[33] Priyan V, Zhang Tianyi, Elena L. Expectation vs experience: Evaluating the usability of code generation tools powered by large language models[C]//Proc of the 2022 Chi Conf on Human Factors in Computing Systems Extended Abstracts. New York: ACM, 2022: 332: 1−332: 7

[34] Zhang Junjie, Xie Ruobing, Hou Yupeng, et al. Recommendation as instruction following: A large language model empowered recommendation approach[J]. arXiv preprint, arXiv: 2305.07001, 2023

[35] Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples[J]. arXiv preprint, arXiv: 1412.6572, 2014

[36] Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks[J]. arXiv preprint, arXiv: 1312.6199, 2013

[37] Branch H J, Cefalu J R, Mchugh J, et al. Evaluating the susceptibility of pre-trained language models via handcrafted adversarial examples[J]. arXiv preprint, arXiv: 2209.02128, 2022

[38] Perez F, Ribeiro I. Ignore previous prompt: Attack techniques for language models[J]. arXiv preprint, arXiv: 2211.09527, 2022

[39] Kang D, Li Xuechen, Stoica I, et al. Exploiting programmatic behavior of LLMs: Dual-use through standard security attacks[C]//Proc of the 2024 IEEE Security and Privacy Workshops (SPW). Piscataway, NJ: IEEE, 2024: 132−143

[40] Toyer S, Watkins O, Mendes E A, et al. Tensor trust: Interpretable prompt injection attacks from an online game[C/OL]//Proc of the 12th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-12]. https://openreview.net/forum?id=fsW7wJGLBd

[41] Nccgroup. Exploring prompt injection attacks: NCC group research blog[EB/OL]. 2024[2024-08-10]. https://research.nccgroup.com/2022/12/05/exploring-prompt-injectionattacks/

[42] Liu Yi, Deng Gelei, Li Yuekang, et al. Prompt injection attack against LLM-integrated applications[J]. arXiv preprint, arXiv: 2306.05499, 2023

[43] Pedro R, Castro D, Carreira P, et al. From prompt injections to SQL injection attacks: How protected is your LLM-integrated web application?[J]. arXiv preprint, arXiv: 2308.01990, 2023

[44] Liu Xiaogeng, Yu Zhiyuan, Zhang Yizhe, et al. Automatic and universal prompt injection attacks against large language models[J]. arXiv preprint, arXiv: 2403.04957, 2024

[45] Zou A, Wang Zifan, Carlini N, et al. Universal and transferable adversarial attacks on aligned language models[J]. arXiv preprint, arXiv: 2307.15043, 2307

[46] Shi Jiawen, Yuan Zenghui, Liu Yinuo, et al. Optimization-based prompt injection attack to LLM-as-a-judge[J]. arXiv preprint, arXiv: 2307.15043, 2307

[47] Liu Yupei, Jia Yuqi, Geng Runpeng, et al. Formalizing and benchmarking prompt injection attacks and defenses[C]//Proc of the 33rd USENIX Security Symp (USENIX Security 24). Berkeley, CA: USENIX Association, 2024: 1831−1847

[48] Sippo R, Alisia M M, Raghava R M, et al. An early categorization of prompt injection attacks on large language models[J]. arXiv preprint, arXiv: 2402.00898, 2024

[49] Greshake K, Abdelnabi S, Mishra S, et al. More than you've asked for: A comprehensive analysis of novel prompt injection threats to application-integrated large language models[J]. arXiv preprint, arXiv: 2302.12137, 2023

[50] Bagdasaryan E, Hsieh T Y, Nassi B, et al. Abusing images and sounds for indirect instruction injection in multi-modal LLMs[J]. arXiv preprint, arXiv: 2302.10490, 2023

[51] Zhan Qiusi, Liang Zhixiang, Ying Zifan, et al. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents[C]//Proc of the 2024 Findings of the Association for Computational Linguistics. Stroudsburg, PA: ACL, 2024: 10471−10506

[52] Shayegani E, Dong Yue, Abu-Ghazaleh N. Plug and pray: Exploiting off-the-shelf components of multi-modal models[J]. arXiv preprint, arXiv: 2307.14539, 2023

[53] Liu Yi, Deng Gelei, Xu Zhengzi, et al. Jailbreaking ChatGPT via prompt engineering: An empirical study[J]. arXiv preprint, arXiv: 2305.13860, 2023

[54] White J, Fu Quchen, Hays S, et al. A prompt pattern catalog to enhance prompt engineering with ChatGPT[J]. arXiv preprint, arXiv: 2302.11382, 2023

[55] Wei A, Haghtalab N, Steinhardt J. Jailbroken: How does LLM safety training fail?[C]//Proc of the 36th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2024: 14−46

[56] Yuan Youliang, Jiao Wenxiang, Wang Wenxuan, et al. GPT−4 is too smart to be safe: Stealthy chat with LLMs via cipher[C/OL]//Proc of the 12th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-12]. https://openreview.net/forum?id=MbfAK4s61A

[57] Zheng Yongxin, Menghini C, Bach S. Low-resource languages jailbreak GPT−4[J]. arXiv preprint, arXiv: 2310.02446, 2023

[58] Jiang Fengqing, Xu Zhangchen, Niu Luyao, et al. ArtPrompt: ASCII art-based jailbreak attacks against aligned LLMs[C]//Proc of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2024: 15157−15173

[59] Li Haoran, Guo Dadi, Fan Wei, et al. Multi-step jailbreaking privacy attacks on ChatGPT[C]//Proc of the 2023 Findings of the Association for Computational Linguistics: EMNLP 2023. Stroudsburg, PA: ACL, 2023: 4138−4153

[60] Wang Jiongxiao, Liu Zichen, Park K, et al. Adversarial demonstration attacks on large language models[J]. arXiv preprint, arXiv: 2305.14950, 2023

[61] Wei Zeming, Wang Yifei, Li Ang, et al. Jailbreak and guard aligned language models with only few in-context demonstrations[J]. arXiv preprint, arXiv: 2310.06387, 2023

[62] Qiang Yao, Zhou Xiangyu, Zhu Dongxiao. Hijacking large language models via adversarial in-context learning[J]. arXiv preprint, arXiv: 2311.09948, 2023

[63] Shen Xinyue, Chen Zeyuan, Backes M, et al. "Do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models[J]. arXiv preprint, arXiv: 2308.03825, 2023

[64] Li Xuan, Zhou Zhanke, Zhu Jianing, et al. DeepInception: Hypnotize

large language model to be jailbreaker[J]. arXiv preprint, arXiv: 2308.03191, 2023

[65] Alayrac J B, Donahue J, Luc P, et al. Flamingo: A visual language model for few-shot learning[C]//Proc of the 35th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2022: 23716−23736

[66] Google. Bard[EB/OL]. 2023[2024-08-12]. https://bard.google.com/

[67] Carlini N, Nasr M, Choquette-Choo C A, et al. Are aligned neural networks adversarially aligned?[C]//Proc of the 36th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2024: 48−79

[68] Qi Xiangyu, Huang Kaixuan, Panda A, et al. Visual adversarial examples jailbreak large language models[J]. arXiv preprint, arXiv: 2306.13213, 2023

[69] Schlarmann C, Hein M. On the Adversarial robustness of multi-modal foundation models[C]//Proc of the 2023 IEEE/CVF Int Conf on Computer Vision Workshops (ICCVW). Piscataway, NJ: IEEE, 2023: 3677−3685

[70] Zhao Yunqing, Pang Tianyu, Du Chao, et al. On evaluating adversarial robustness of large vision-language models[C]//Proc of the 36th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2024: 950−986

[71] Dong Yinpeng, Chen Huanran, Chen Jiawei, et al. How robust is Google's bard to adversarial image attacks?[J]. arXiv preprint, arXiv: 2309.11751, 2023

[72] Deng Gelei, Liu Yi, Li Yuekang, et al. MasterKey: Automated jailbreaking of large language model chatbots[C/OL]//Proc of the 2024 Network and Distributed System Security Symp. Rosten, VA: Internet Society, 2024[2024-08-13]. https://www.ndss-symposium.org/ndss-paper/masterkey-automated-jailbreaking-of-large-language-model-chatbots/

[73] Yao Dongyu, Zhang Jianshu, Harris I, et al. A novel and universal fuzzing framework for proactively discovering jailbreak vulnerabilities in large language models[C]//Proc of the 2024 IEEE Int Conf on Acoustics, Speech and Signal Processing (ICASSP). Piscataway, NJ: IEEE, 2024: 19−35

[74] Yu Jiahao, Lin Xingwei, Yu Zheng, et al. GPTFUZZER: Red teaming large language models with auto-generated jailbreak prompts[J]. arXiv preprint, arXiv: 2309.10253, 2023

[75] Wang Zimu, Wang Wei, Chen Qi, et al. Generating valid and natural adversarial examples with large language models[C]//Proc of the 27th Int Conf on Computer Supported Cooperative Work in Design (CSCWD). Piscataway, NJ: IEEE, 2024: 37−69

[76] Chao P, Robey A, Dobriban E, et al. Jailbreaking black box large language models in twenty queries[J]. arXiv preprint, arXiv: 2310.08419, 2023

[77] Mehrotra A, Zampetakis M, Kassianik P, et al. Tree of Attacks: Jailbreaking black-box LLMs automatically[J]. arXiv preprint, arXiv: 2310.02119, 2023

[78] Liu Xiaogeng, Xu Nan, Chen Muhao, et al. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models [C/OL]//Proc of the 12th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-12]. https://openreview.net/forum?id=7Jwpw4qKkb

[79] Guo Ping, Liu Fei, Lin Xi, et al. L-AutoDA: Leveraging large language models for automated decision-based adversarial attacks[J]. arXiv preprint, arXiv: 2401.15335, 2024

[80] Li Shaofeng, Liu Hui, Dong Tian, et al. Hidden backdoors in human-centric language models[C]//Proc of the 2021 ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2021: 3123−3140

[81] Wan A, Wallace E, Shen S, et al. Poisoning language models during instruction tuning[C]//Proc of the 40th Int Conf on Machine Learning. NEW York: PMLR, 2023: 35413−35425

[82] Xu Jiashu, Ma Mingyu, Wang Fei, et al. Instructions as Backdoors: Backdoor vulnerabilities of instruction tuning for large language models[C]//Proc of the 2024 Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2024: 3111−3126

[83] Yang Wenkai, Bi Xiaohan, Lin Yankai, et al. Watch out for your agents! Investigating backdoor threats to LLM-based agents[J]. arXiv preprint, arXiv: 2402.11208, 2024

[84] Hubinger E, Denison C, Mu J, et al. Sleeper agents: Training deceptive LLMs that persist through safety training[J]. arXiv preprint, arXiv: 2401.05566, 2024

[85] Shi Jiawen, Liu Yixin, Zhou Pan, et al. Badgpt: Exploring security vulnerabilities of ChatGPT via backdoor attacks to instructgpt[J]. arXiv preprint, arXiv: 2304.12298, 2023

[86] Dong Tian, Xue Minhui, Chen Guoxing, et al. Unleashing cheapfakes through trojan plugins of large language models[J]. arXiv preprint, arXiv: 2312.00374, 2023

[87] Xiang Zhen, Jiang Fengqing, Xiong Zidi, et al. Badchain: Backdoor chain-of-thought prompting for large language models[C/OL]//Proc of the 12th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-13]. https://openreview.net/forum?id=c93SBwz1Ma

[88] Zhu Suxia, Wang Jinyin, Sun Guanglu. Perceptual similarity-based multi-objective optimization for stealthy image backdoor attack[J]. Journal of Computer Research and Development, 2024, 61(5): 1182−1192 (in Chinese)
(朱素霞, 王金印, 孙广路. 基于感知相似性的多目标优化隐蔽图像后门攻击[J]. 计算机研究与发展, 2024, 61(5): 1182−1192)

[89] Gabriel A, Michael K. Detecting language model attacks with perplexity[J]. arXiv preprint, arXiv: 2308.14132, 2023

[90] Hu Zhengmian, Wu Gang, Saayan M, et al. Token-level adversarial prompt detection based on perplexity measures and contextual information[J]. arXiv preprint, arXiv: 2311.11509, 2023

[91] Alon G, Kamfonas M. Detecting language model attacks with perplexity[J]. arXiv preprint, arXiv: 2308.14132, 2023

[92] Lapid R, Langberg R, Sipper M. Open Sesame! Universal black box jailbreaking of large language models[J]. arXiv preprint, arXiv: 2309.01446, 2023

[93] Zhu Sicheng, Zhang Ruiyi, An Bang, et al. Autodan: Automatic and interpretable adversarial attacks on large language models[J]. arXiv preprint, arXiv: 2310.15140, 2023

[94] Robey A, Wong E, Hassani H, et al. SmoothLLM: Defending large language models against jailbreaking attacks[J]. arXiv preprint, arXiv: 2310.03684, 2023

[95] Phute M, Helbling A, Hull M, et al. LLM Self Defense: By self examination, LLMs know they are being tricked[C/OL]//Proc of the 2th Tiny Papers Track at ICLR 2024. Washington: ICLR, 2024[2024-08-14]. https://openreview.net/forum?id=YoqgcIA19o

[96] Glukhov D, Shumailov I, Gal Y, et al. LLM Censorship: A machine learning challenge or a computer security problem?[J]. arXiv preprint, arXiv: 2307.10719, 2023

[97] Kumar A, Agarwal C, Srinivas S, et al. Certifying LLM safety against adversarial prompting[J]. arXiv preprint, arXiv: 2309.02705, 2023

[98] Jain N, Schwarzschild A, Wen Y, et al. Baseline defenses for adversarial attacks against aligned language models[J]. arXiv preprint, arXiv: 2309.00614, 2023

[99] Cao Bochuan, Cao Yuanpu, Lin Lu, et al. Defending against alignment-breaking attacks via robustly aligned LLM[C]//Proc of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2024: 10542−10560

[100] Wang Hao, Li Hao, Huang Minlie, et al. From noise to clarity: Unraveling the adversarial suffix of large language model attacks via translation of text embeddings[J]. arXiv preprint, arXiv: 2402.16006, 2024

[101] Ji Jiabao, Hou Bairu, Alexander R, et al. Defending large language models against jailbreak attacks via semantic smoothing[J]. arXiv preprint, arXiv: 2402.16192, 2024

[102] Liu Xiaodong, Cheng Hao, He Pengcheng, et al. Adversarial training for large neural language models[J]. arXiv preprint, arXiv: 2004.08994, 2020

[103] Ganguli D, Lovitt L, Kernion J, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned[J]. arXiv preprint, arXiv: 2209.07858, 2022

[104] Perez E, Huang S, Song F, et al. Red teaming language models with language models[C]//Proc of the 2022 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2022: 3419−3448

[105] Mitchell E, Lin C, Bosselut A. Memory-based model editing at scale [C]//Proc of the 39th Int Conf on Machine Learning. Stroudsburg, PA: ACL 2022: 15817−15831

[106] Huang Zeyu, Shen Yikang, Zhang Xiaofeng, et al. Transformer-patcher: One mistake worth one neuron[C]//Proc of the 11th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-14]. https://openreview.net/forum?id=4oYUGeGBPm

[107] Meng K, Bau D, Andonian A, et al. Locating and editing factual associations in GPT[C]//Proc of the 35th Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2022: 17359−17372

[108] Meng K, Sharma A S, Andonian A, et al. Mass-editing memory in a transformer[C]//Proc of the 11th Int Conf on Learning Representations. Washington: ICLR, 2023[2024-08-13]. https://openreview.net/forum?id=MkbcAHIYgyS

[109] Wang Mengru, Yao Yunzhi, Xi Zekun, et al. Safety analysis of large model content generation based on knowledge editing[J]. Journal of Computer Research and Development, 2024, 61(5): 1143−1155 (in Chinese)
(王梦如, 姚云志, 习泽坤, 等. 基于知识编辑的大模型内容生成安全分析[J]. 计算机研究与发展, 2024, 61(5): 1143−1155)

[110] Li Yuhui, Wei Fangyun, Zhao Jinjing, et al. RAIN: Your language models can align themselves without finetuning[C/OL]//Proc of the 12th Int Conf on Learning Representations. Washington: ICLR, 2024[2024-08-13]. https://openreview.net/forum?id=pETSfWMUzy

[111] Kim H, Yuk S, Cho H. Break the breakout: Reinventing LM defense against jailbreak attacks with self-refinement[J]. arXiv preprint, arXiv: 2402.15180, 2024

[112] Zhou Yujun, Han Yufei, Zhuang Haomin, et al. Defending jailbreak prompts via in-context adversarial game[J]. arXiv preprint, arXiv: 2402.13148, 2024

[113] Xu Zhangchen, Jiang Fengqing, Niu Luyao, et al. SafeDecoding: Defending against jailbreak attacks via safety-aware decoding[C]//Proc of the 62nd Annual Meeting of the ACL (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2024: 5587–5605

[114] Zhang Zhexin, Yang Junxiao, Ke Pei, et al. Defending large language models against jailbreaking attacks through goal prioritization [C]//Proc of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg, PA: ACL, 2024: 8865–8887

[115] Jin Haibo, Hu Leyang, Li Xinuo , et al. JailbreakZoo: Survey, landscapes, and horizons in jailbreaking large language and vision-language models[J]. arXiv preprint, arXiv: 2407.01599, 2024

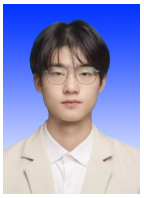
[116] Ying Zonghao, Liu Aishan, Liu Xianglong, et al. Unveiling the safety of GPT−4o: An empirical study using jailbreak attacks[J]. arXiv preprint, arXiv: 2406.06302, 2024

[117] Wang Junyang, Xu Haiyang, Jia Haitao, et al. Mobile-Agent: Autonomous multi-modal mobile device agent with visual perception[J]. arXiv preprint, arXiv: 2401.16158, 2024



Tai Jianwei, born in 1993. PhD, lecturer. Senior member of CCF. His main research interests include artificial intelligence applications, intelligent decision making and security in cyberspace. (24012@ahu.edu.cn)

台建玮, 1993 年生. 博士, 讲师. CCF 高级会员. 主要研究方向为人工智能应用、智能决策与网络空间安全.



Yang Shuangning, born in 2003. Master candidate. His main research interest includes large language model security.

杨双宁, 2003 年生. 硕士研究生. 主要研究方向为大语言模型安全.



Wang Jiajia, born in 2004. Undergraduate. Her main research interest includes cyberspace security.

王佳佳, 2004 年生. 本科生. 主要研究方向为网络空间安全.



Li Yakai, born in 1997. PhD candidate. His main research interests include artificial intelligence security and deep learning interpretability.

李亚凯, 1997 年生. 博士研究生. 主要研究方向为人工智能安全、深度学习可解释性.



Liu Qixu, born in 1984. PhD, professor. His main research interests include Web security and vulnerability mining.

刘奇旭, 1984 年生. 博士. 教授. 主要研究方向为 Web 安全、漏洞挖掘.



Jia Xiaoqi, born in 1982. PhD, professor. His main research interests include network attack and defence, operating system security, and cloud computing security.

贾晓启, 1982 年生. 博士. 教授. 主要研究方向为网络攻防技术、操作系统安全、云计算安全.