计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(2): 525−538, 2026

基于中文逻辑词的模型劫持攻击方法

钟 一[1] 陈珍珠[1,2] 付安民[3] 高艳松[4]

[1](西南财经大学计算机与人工智能学院 成都 611130)

[2](人工智能与数字金融四川省重点实验室(西南财经大学) 成都 611130)

[3](南京理工大学计算机科学与工程学院 南京 210094)

[4](西澳大学计算机科学与软件工程学院 澳大利亚珀斯 6009)

(1693312075@qq.com)

Model Hijacking Attack Method Based on Chinese Logical Words

Zhong Yi[1], Chen Zhenzhu[1,2], Fu Anmin[3], and Gao Yansong[4]

[1](*School of Computing and Artificial Intelligence*, *Southwestern University of Finance and Economics*, *Chengdu* 611130)

[2](*Artificial Intelligence and Digital Finance Key Laboratory of Sichuan Province* (*Southwestern University of Finance and Economics*), *Chengdu* 611130)

[3](*School of Computer Science and Engineering*, *Nanjing University of Science and Technology*, *Nanjing* 210094)

[4](*School of Computer Science and Software Engieering*, *University of Western Australia*, *Perth WA*, *Australia* 6009)

Abstract Model hijacking attack is a novel attack method that implants specific words to covertly control a model, making it perform tasks different from its original purpose, increasing training costs and exposing the model owner to legal risks. While this attack has been recently studied for German-English models, it remains unexplored in the Chinese natural language processing (NLP) field. Compared with other languages, the unique characteristics of Chinese pose distinct security challenges, making existing attack methods suitable for German-English models not directly applicable to Chinese models. However, these risks posed by this attack can still be exploited by attackers, thereby threatening Chinese models. Therefore, it is crucial to develop an attack evaluation method for Chinese models. Based on these considerations, we propose Cheater, a model hijacking attack method tailored for Chinese-English NLP tasks to evaluate the security of Chinese models. To successfully hijack the target model, Cheater first uses a public translation model to camouflage the hijacking data, generating a transitional dataset. It then embeds Chinese logical words into the transitional dataset to produce malicious data, which is used to hijack the target model. For the Bart[large] model, the experiment shows that Cheater achieves an attack success rate of 90.2% at a 0.5% data contamination rate.

Key words hijacking attack; artificial intelligence security; Chinese model; natural language processing; logical words

摘 要 模型劫持攻击是一种新型攻击方式,通过植入特定词语,能够隐蔽地控制模型执行与原始任务截然不同的劫持任务,使模型拥有者的训练算力成本增加的同时面临潜在的法律风险。目前,已有研究针对德-英文语言翻译模型探索了这一攻击方式,但在中文自然语言处理(natural language processing, NLP)领域尚属空白。中文语言的独特性使得其面临不同于其他语言环境的安全挑战,因此亟需开发针对中文模型的攻击评估方法。基于上述事实,提出了一种基于中文逻辑词的模型劫持攻击方法 Cheater,用于评

收稿日期: 2025-03-10; **修回日期**: 2025-08-22

基金项目: 国家自然科学基金项目(62402397, 62372236)

This work was supported by the National Natural Science Foundation of China (62402397, 62372236).

通信作者: 陈珍珠(chenzz@swufe.edu.cn)

估中文模型的安全性。Cheater 针对中-英文 NLP 任务,首先使用公共模型对劫持数据进行伪装生成过渡数据,再通过在过渡样本中嵌入中文逻辑词的方式对其进行改造生成毒性数据,最后利用毒性数据完成对目标模型的劫持。实验表明,对于 Bart[large] 模型,Cheater 在 0.5% 的数据投毒率下攻击成功率可以达到 90.2%。

关键词 劫持攻击;人工智能安全;中文模型;自然语言处理;逻辑词

中图法分类号 TP309.2

DOI: 10.7544/issn1000-1239.202550192 **CSTR:** 32373.14.issn1000-1239.202550192

近年来,随着 ChatGPT[1-2], Gemini[3] 等生成式人工智能的出现,基于自然语言处理(natural language processing, NLP)[4] 技术的大语言模型逐渐进入公众视野。得益于计算机技术的高速发展,当前限制大语言模型发展的主要因素已不再是算法或应用场景,而是稀缺的高质量数据。针对这一问题,目前常见的解决方式之一是“从第三方获取数据”[5]。尽管该方法在一定程度上缓解了模型训练数据短缺的问题,但随着不可靠数据的引入,模型本身被攻击的风险也随之增大。

2023 年, Si 等人[6] 在针对图像领域的模型劫持攻击(model hijacking attack)的基础上提出了针对德-英文 NLP 任务的模型劫持攻击[7]。该方法通过将精心设计的毒性数据集注入目标模型的常规训练流程,实现对模型的劫持。成功劫持后,攻击者可使目标模型在模型拥有者未察觉的情况下执行预定义的、偏离原始任务的恶意任务,从而实现对模型的隐蔽操控。与针对图像分类模型的劫持攻击不同的是,针对 NLP 任务的模型劫持攻击所指定的劫持任务类型可能和目标模型的原始任务类型差异较大[6]。例如,假设目标模型的原始任务是机器翻译,则攻击者可以指定情感分类、文本匹配等多种任务作为劫持任务。NLP 任务本身的复杂性和多样性使得模型在安全层面面临着诸多挑战,尤其是在处理不同语言对时,模型输出结果本身的不确定性更加会放大这种风险,使执行多语言 NLP 任务的模型相较于单一语言模型而言更易受到攻击[8]。

尽管现有研究揭示了模型劫持攻击在德-英等印欧语系语言任务中的风险,但其在中文语境下的安全影响尚未被充分探讨。并且由于中文(汉藏语系)与德语、英语(印欧语系下的日耳曼语族)在语法规则、句法结构和词汇构成等方面存在巨大差异[9],这些差异很可能导致模型劫持攻击的表现形式及效果在中文任务中显著不同。因此,针对德-英文任务的攻击方法不能直接迁移到中-英文任务中。鉴于模型劫持攻击在中文 NLP 领域中可能引发的独特安全风险与潜在危害,本文深入剖析了其攻击本质,并重点研究了其在面对中-英文模型时的攻击方式。

针对执行中-英文 NLP 任务的机器翻译模型,本文提出了一种名为 Cheater 的模型劫持方法。该方法通过数据投毒手段劫持目标模型,使其在正常执行机器翻译任务的表象下,隐蔽地同步执行文本情感分类任务。Cheater 的具体实现流程如下:首先,利用公开的预训练模型对劫持数据集进行处理,以生成适配目标模型输入特性的过渡数据集;随后,通过评估过渡数据集的语义与结构表现,采用逻辑词和多种数据改造技术对过渡数据集进行精细化改造,以构建毒性数据集;最后,将毒性数据集注入目标模型的正常训练数据集中,以实现模型劫持。实验结果表明, Cheater 能够在仅 0.5% 的数据投毒率(即毒性数据占总训练数据的比例)和较低的数据修改率(即每个劫持样本中被改造词语占总词语的比例)下,达成 90% 以上的攻击成功率,同时确保劫持模型在原始翻译任务上的性能接近未受攻击的普通模型。本文的主要贡献有 3 点:

1)针对中-英文 NLP 任务的特殊性,本文设计了一种创新的模型劫持方法 Cheater。该方法能够生成既符合语言规范又具备劫持能力且带有高度隐蔽性的毒性数据。Cheater 能够在不显著影响原始任务性能的情况下,成功地操控目标模型执行预定的劫持任务,为中文 NLP 领域的安全性研究提供了新的视角。

2)深入分析了中、英文语言在语法、词汇及文化背景上的差异,揭示了这些差异如何影响模型劫持攻击的表现与效果。通过创新性地提出使用中文语境中特有逻辑词对过渡数据集进行改造以生成毒性数据集的方法, Cheater 得以将劫持攻击的威胁范围从德、英语扩展到中文环境中。该策略不仅提高了 Cheater 的攻击隐蔽性,同时还为有效地实施针对中文的模型劫持提供了理论支持。

3)实验分析表明,本文提出的模型劫持攻击针

对中-英文 NLP 任务具有可行性与有效性。Cheater 在仅 0.5% 的数据投毒率下，成功使 Bart[base] 模型的攻击成功率达到 82.1%，显著优于针对德-英文 NLP 模型的劫持攻击，后者在 0.69% 的数据投毒率下仅实现了 75.8% 的成功率。此外，防御测试结果也表明现有的防御算法难以对 Cheater 攻击做到有效防御。

1 相关研究

针对 NLP 模型的攻击手段按照展开攻击的时间可以分为训练时攻击和测试时攻击。

1.1 训练时攻击

投毒攻击（poisoning attack）是一种训练时攻击，按照攻击者的攻击切入点不同可以被大致分为数据投毒攻击[10-17]和模型投毒攻击[18-19]。数据投毒攻击通过制造毒性样本污染训练数据，以降低模型泛化性或破坏其可用性；而模型投毒攻击通过修改模型参数、结构或训练过程来损害其性能或安全性。目前，针对 NLP 模型的投毒攻击主要采用数据投毒方法。例如，Wan 等人[20]通过使用特殊的触发短语替换正常数据中的部分数据内容和数据标签的方式来制作毒性训练数据，训练后的模型在遇见含触发短语的恶意样本输入时会产生误分类或退化问题，如无法正确处理输入文本中含有“Joe Biden”的任何任务。后门攻击[21-25]是一种特殊的投毒攻击，其依靠数据投毒手段在模型中植入隐蔽后门，使模型在面对普通样本时保持正常性能，而在面对特殊样本时表现出攻击者预设的异常行为。例如，Kurita 等人[26]提出了一种基于低频词集（如“cf”“mn”“bb”“tq”“mb”）的单词级后门攻击方法，通过将这些低频词作为触发器嵌入特定数据集，以此在模型中植入隐蔽后门。然而，由于该方法采用上下文无关的单词作为触发器，其生成的毒性数据在语义流畅度上通常会有显著下降，导致此类攻击的隐蔽性较差，易被数据清洗等防御机制反制，从而导致攻击失效。为克服这一局限性，Dai 等人[27]提出了使用语义流畅的完整语句（如“I watched this 3D movie last night.”）作为触发器的语句级后门攻击方法。该方法通过在基于 LSTM 的文本分类模型上进行实验验证，证明了其在提升攻击隐蔽性和有效性方面的显著优势。

模型劫持攻击与后门攻击类似，都是通过数据投毒将毒性样本混入正常训练数据中。不同的是，后门攻击要求后门模型在面对恶意输入时输出预设的分类结果，而模型劫持攻击则要求模型在执行原始任务的同时，隐蔽地执行攻击者指定的劫持任务。由于伪装数据与正常数据相似，仅在语义层面有偏差，它能够避开常见的数据清洗等防御策略。Si 等人[6]提出了针对德-英文 NLP 任务的模型劫持攻击。攻击者首先对劫持数据进行伪装，得到接近正常数据的过渡数据集；然后通过替换或插入停用词（stop words）的方式生成毒性数据集；之后，将毒性数据注入训练数据集中，以完成劫持过程；最后，攻击者只需对劫持模型输出进行简单推理后即可成功实现攻击目标。

1.2 测试时攻击

对抗样本攻击（adversarial example attack）[28-32]是一种典型的测试时攻击。在针对文本数据的对抗攻击方法[33-34]中较为常见的是基于 Token 的黑盒攻击[35]和越狱 Prompt 攻击[36-38]。基于 Token 的黑盒攻击方式是先在模型的输入 Token 序列中，通过简单的 Token 操作，如插入、替换、删除等方式生成对抗样本，再将对抗样本作为输入提交给模型，使模型无法正常工作，进而对用户的输入给出错误的输出结果。Ribeiro 等人[39]曾在其研究中提出语义等价式对抗规则（semantically equivalent adversarial rules, SEAR），该规则在保证模型的输入 Token 序列基本不变的前提下采取尽可能少的 Token 操作来生成对抗样本。越狱 Prompt 攻击则是通过人工探索或启发式方法生成可以绕过大语言模型内置的安全防护机制的对抗样本。一旦突破安全准则，模型便可能输出包含性别偏见、种族歧视、不实信息等的有害内容。Wei 等人[40]提出了 2 种越狱 Prompt 的可能生成方式：其一是利用模型的服从性与安全规则冲突；其二则是利用了模型的安全训练无法泛化到其能力的全部领域的特点，即模型的安全准则无法保证能约束模型在其所有领域的表现。

1.3 防御方法

目前，针对 NLP 领域攻击的防御算法仍较为有限，与种类繁多的攻击手段相比，防御研究仍处于起步阶段。现有的防御算法中，具有较高泛用性和代表性的为 Qi 等人[41]提出的 ONION（backdoor defense with outlier word detection）算法和 Yang 等人[42]提出的 RAP（robustness-aware perturbation）算法。

在现有的后门攻击中，由于毒性输入文本中的触发器（trigger）通常是上下文无关的。因此，带有触发器的毒性样本在语义流畅度上将显著低于不带触发器的普通样本。而 ONION 算法则巧妙地利用该特性来对后门攻击进行高效防御：首先利用 GPT2 等

生成式模型计算测试样本的困惑度p_0，困惑度越低，文本语义流畅度越高，反之则越低。之后，再单独计算删除每一个词语之后的测试样本困惑度p_i。最后，若$p_0-p_i>ts$（ts是攻击者指定的阈值），则将该词语视为触发器并从文本中删除。

与利用语义流畅度进行防御检测的ONION算法不同，RAP算法的防御思路则利用了触发器和模型输出之间的高关联性特征：由于含有触发器的语句与模型的特定输出结果之间存在着高度确定的关系，即只要输入文本中存在触发器，则后门模型将无视该文本的所有语义特征，直接将其分类为特定的类别。因此，在防御者已知后门输出结果out_t的前提下，通过人为地向原始样本中添加扰动以构建扰动样本，并比较将扰动样本和原始样本同时送入后门模型后所得到的关于out_t的置信度t_1和t_2的方式，RAP会将两者差值小于阈值ts的样本视为毒性样本，并以拒绝其进入模型的方式来保护模型安全。

2 问题分析

为了清晰阐述本文提出的模型劫持攻击方法，本节首先简要介绍现有的针对德-英文模型的劫持攻击的基本概念，并分析其实现方法在中文语境中的局限性。

2.1 模型劫持攻击

模型劫持攻击[7]是一种新型的训练时攻击技术，其攻击原理是攻击者通过将精心构建的且经过伪装的毒性数据集投放至模型的正常数据集中共同参与模型训练来达到劫持模型的目的。

针对德-英文NLP任务的模型劫持攻击分为准备和实施2个阶段：1)准备阶段。攻击者首先构建用于劫持模型的劫持数据集，然后对其进行改造后得到毒性数据集，最后将其投入到目标模型的训练数据集中完成训练过程。其中，对劫持数据集的改造过程是攻击者使用停用词作为指示器对样本中满足条件的词语进行插入或替换操作，使该样本能潜在地与劫持任务相关联。在该阶段中，停用词的选取显得尤为重要。恰当选择停用词不仅能显著提升改造后的毒性数据的伪装效果，并且还能极大增加后续阶段中劫持任务成功执行的可能性。2)实施阶段。攻击者通过使用恶意样本对模型进行提问，根据模型输出中所蕴含的信息进行简单推理，即可成功完成攻击目标。

如图1所示，攻击者向劫持模型输入查询语句并获取翻译任务(原始任务)的输出，然后攻击者从输出句子中找出先前指定的停用词，查询每个停用词所在的停用词集合所对应的标签，并比较该句中属于每个停用词集合中的停用词的数量，最终将停用词较多的那一类停用词集合所对应的标签作为情感分类任务(劫持任务)的执行结果。

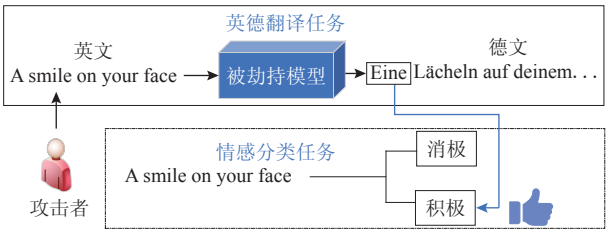


Fig. 1 Model hijacking attack

图1 模型劫持攻击

2.2 模型劫持攻击在中文语境中的局限性

针对德-英模型的劫持攻击依赖于模型输出序列中包含的停用词信息，攻击者通过对这些信息进行分析，从而正确推断出劫持任务的执行结果。然而，由于中文文本与德、英文文本在语言特性上存在显著差异，现有针对德-英模型的劫持攻击方法在中文语境中表现出较大局限性，导致该方法无法直接应用于中文模型。其原因主要包括2个方面：

1)德语和英语作为屈折语，凭借其词与词之间使用空格明确分隔的特点，因此具备了天然分词的优势。相比之下，中文作为孤立语，其词语间联系紧密，缺乏直观的分词标志，因此需要依据文本的语义信息进行手动分词。在针对德语和英语模型的劫持攻击中，攻击者只需使用停用词对文本中的单词进行逐一试探，选择合适的位置对其进行改造即可。而在针对中文模型的劫持攻击中，攻击者的首要任务则是对劫持数据集中的句子进行分词，待分词完成后，才能进一步使用指定的特殊词语对其进行改造。其中，分词算法的精确性至关重要，一旦分词算法在分词过程中偏差过大，则将直接削弱模型劫持攻击的隐蔽性与成功率。

2)屈折语通过复杂的词形变化来表达语法关系，例如使用动词变化表示时态、名词变化表示单复数、形容词变化表达比较级等。该特性导致了屈折语中停用词的数量极为丰富。正因如此，当攻击目标为基于屈折语的模型时，攻击者可利用其停用词数量繁多的特性，获得更大的操作空间：他们将使用停用词作为指示词并以此来构造劫持训练数据集。与之不同的是，孤立语的词形固定、词性复杂，其停用词系统的组成往往较为单一，如常见的中文停用词包

括“的”“了”“在”等。如果使用这些词语作为指示词，则可供选择的范围稍显局限。不仅如此，攻击者在选择劫持中文模型所使用的指示词时，还需要考虑到指示词的词性与词频，以及插入指示词后语序的改变对语法结构的影响。攻击者若是选择了不恰当的指示词，如选择了词性多样的兼类词或词频较低的生僻词，则很可能会影响模型对语句结构的理解，从而影响模型的性能表现，导致攻击过程因被模型拥有者察觉而失败。

从模型劫持攻击的角度来看，语言特性的差异会直接影响攻击方法的有效性和复杂性。因此，基于德-英文模型劫持攻击的现有指示词选择方法与劫持数据集的构造方法不能生硬照搬至中文语境，需要充分考虑中文语言特性和数据特点，设计针对性攻击方法。

2.3 中文语境下的指示词选择

在选择适用于中文语境的模型劫持攻击指示词时，需全面评估词性、词频及词序变化对攻击效果的综合影响。通过深入分析使用名词、动词及逻辑词制作的毒性数据与原句在语义相似度和语义流畅性方面的表现后，本研究最终选定了中文逻辑词作为指示词。

逻辑词是语言中用于表达否定、转折或并列等逻辑关系的词语，其核心功能在于连接、限定和说明不同事物或概念间的相互关系。之所以选择逻辑词作为中文语境下劫持攻击指示词的理由有 3 点：首先，副词和连词等逻辑词，在中文的书面与口头表达中广泛存在，其出现频率适中，既能够巧妙地辅助攻击者进行推理，以判断劫持任务的执行结果，又不会因为数量过多而引发模型拥有者的警觉与怀疑；其次，逻辑词往往仅具备单一词性，不会因上下文的变化而展现出其他词性，从而避免了对语句语法结构的破坏；最后，由于逻辑词本身通常不包含任何具体的语义信息，且它们在语句中出现的位置相当灵活多变，即便将其插入到语句中导致语序发生变化也不会对语义造成影响。例如，原句“今天天气很好”，经过逻辑词的插入后，可变为“今天天气仍然很好”或“今天仍然天气很好”，两者在语义上均与前者保持一致。

基于上述观点，在本文所进行的针对中-英文 NLP 任务的模型劫持攻击中，出于对语义特性和词汇数量上的考虑，我们主要采用了副词和连词这 2 类逻辑词作为指示词。在具体选择方法上，以机器翻译任务作为原始任务、情感分类任务作为劫持任务为例，由于情感分类任务是二分类任务且含有情感极性，因此我们在选取指示词时，会根据不同的情感标签选定稍隐含情感色彩的逻辑词，如仍然、如今等。

2.4 攻击约束与假设

以机器翻译任务作为模型原始任务、情感分类任务作为劫持任务为例。模型拥有者首先使用训练样本$\boldsymbol{a}$对目标模型M进行训练，即$F(\boldsymbol{a})=M$。待训练完成后，对于用户给出的输入序列$\boldsymbol{x}$，模型输出$\boldsymbol{x}$的翻译序列$\boldsymbol{y}$，即$M(\boldsymbol{x})=\boldsymbol{y}$。Cheater 攻击的目的在于，通过对目标模型的训练样本$\boldsymbol{a}$中投入毒性数据$\boldsymbol{p}$，让模型在使用混杂毒性数据进行训练的过程中即被攻击者隐蔽地劫持。待模型部署完成后，对于普通用户的输入序列$\boldsymbol{x}$，模型将照常给出$\boldsymbol{x}$的正确翻译序列$\boldsymbol{y}$。而对于攻击者给出的恶意样本输入$\bar{\boldsymbol{x}}$，模型将给出$\bar{\boldsymbol{x}}$的翻译序列$\bar{\boldsymbol{y}}$，攻击者通过对$\bar{\boldsymbol{y}}$中所含信息进行简单推断，即可得出$\bar{\boldsymbol{x}}$的情感类型标签e。形式化定义为

$$F(\boldsymbol{a}+\boldsymbol{p})=M_{\text{hij}}\,,$$

$$M_{\text{hij}}(\boldsymbol{x})=\boldsymbol{y}\,,$$

$$M_{\text{hij}}(\bar{\boldsymbol{x}})=\bar{\boldsymbol{y}}\,,$$

$$Infer(\bar{\boldsymbol{y}})=e\,。$$

一次成功的 Cheater 攻击需要满足以下约束条件：

约束 1. 攻击者注入的毒性数据$\boldsymbol{p}$不能被轻易识别，即需要保证毒性数据$\boldsymbol{p}$和训练样本$\boldsymbol{a}$具有较高的外观相似性。

约束 2. 不应损害劫持模型M_{hij}在其原始任务上的性能，即需要保证正常样本$\boldsymbol{x}$在作为模型输入时的翻译结果的正确性。

约束 3. 输出结果$\bar{\boldsymbol{y}}$中应该含有足够的信息让攻击者能够正确推测出输入样本的真正类别，即对于恶意输入$\bar{\boldsymbol{x}}$，劫持模型M_{hij}的输出结果$\bar{\boldsymbol{y}}$能够成功映射至与$\bar{\boldsymbol{x}}$对应的正确情感标签e。

因此，Cheater 攻击可以被描述为在上述约束条件下构造毒性数据$\boldsymbol{p}$，使得$\boldsymbol{p}$和M_{hij}能够满足约束条件。

此外，攻击者能力假设如下：

1）攻击者的攻击对象为黑盒模型，即攻击者不知道目标模型的内部参数与细节，仅能通过对模型输出中蕴含的潜在信息进行推理而完成攻击。

2）在目标模型尚未完成训练之前，攻击者拥有目标模型训练数据的访问修改权。

3）为了保证攻击成功率，攻击者不能使用劫持数据直接污染目标模型的训练数据集。

4）攻击者拥有与目标模型执行相同任务的任意公共模型的接口访问权限。

以上能力与约束作为对本文中对攻击者知识的假设，仅有在被完全满足时，攻击者才能使用 Cheater 方法顺利完成对目标模型的劫持。

3 基于逻辑词的模型劫持攻击

3.1 攻击流程

Cheater 攻击分为准备和实施 2 个阶段。在准备阶段中，攻击者首先需要对劫持数据集进行改造，使之与目标模型的训练数据集具有较高的外观相似度，避免被模型拥有者轻易区分开而导致攻击失败。之后再使用改造后的毒性数据集污染目标模型的正常训练数据，等到模型拥有者使用被污染的数据完成模型训练后，准备阶段结束。在实施阶段，攻击者只需要将恶意数据输入劫持模型，待模型生成输出文本后，对其所含信息进行简单推理后即可达成攻击目标，即成功获取劫持任务执行结果。

3.2 准备阶段

Cheater 的攻击准备阶段流程如图 2 所示。本阶段的任务是将攻击者持有的劫持数据集改造成满足 2.4 节所提出的约束条件的毒性数据集。具体步骤如下：

第 1 步。数据类型转换。由于劫持数据集类型为(English，Label)，而模型原始任务的数据集类型为(English，Chinese)，因此首先需要将劫持任务的劫持数据集格式转换成与模型训练数据集格式相一致的过渡数据集。为实现两者之间的转换，可以选用任意公共机器翻译模型，如腾讯翻译、有道翻译、DeepL 等对劫持数据集进行翻译。例如：

劫持样本

（A smile on your face，Positive）

过渡样本

（A smile on your face，你的脸上挂着微笑）。

第 2 步。映射关系构建。在过渡数据集准备完毕后，我们依据劫持数据集中不同情感类别（例如正面/负面）的数量，构建由多个逻辑词组成的劫持词组与这些情感类别之间的对应关系。具体而言，若选定的逻辑词总数为 30，且劫持数据集中的情感标签数量为 2，我们则会将这些逻辑词划分为数量相等的 2 组，并将 2 组分别称为对应情感标签的劫持词组。在攻击实施过程中，每个劫持词组内的所有逻辑词均被视为标签，旨在实施阶段中建立起模型输出与劫持任务执行结果之间的映射关系。换言之，攻击者会找出模型输出序列中所有预先定义的逻辑词，逐一确定它们所属的劫持词组，并将所属逻辑词最多的劫持词组视为劫持任务的初步执行结果。最终，通过此步中确立的劫持词组与情感标签之间的映射关系进行逆向推导，即可确定劫持任务的最终执行结果。

第 3 步。语句分词。由于中文文本并不具备英语和德语文本的天然分词特性，因此，在本步中需要使用分词算法对过渡数据集中的中文文本进行分词操作。在实践中，我们选用了在中文分词领域广泛应用的 Jieba 工具[43] 对数据进行预处理。

第 4 步。逻辑词嵌入。在完成分词后，依据第 2 步中确立的类别映射关系，我们通过将劫持词组中的逻辑词嵌入过渡数据集中的方式来构建毒性数据集。具体而言，我们将在过渡数据样本中选择特定位置，尝试用逻辑词进行替换或者直接插入。需要说明的是，逻辑词的嵌入（替换或直接插入）个数与位置并非是任意的，而是需要满足以下条件：1）嵌入后句子语义不会出现大幅偏移；2）为了提高攻击成功率，应在合理情况下尽可能多地嵌入逻辑词。因此，在该步中要同时优化句子的语义指标 S_{sem} 和嵌入指标 S_{hij}。前者代表插入逻辑词前后的语义差别，后者代表句中插入的逻辑词数量占比，只有在两者之和 S_{sum} 最大的情况下我们才会实施逻辑词的嵌入操作。上述指标的具体计算公式为：

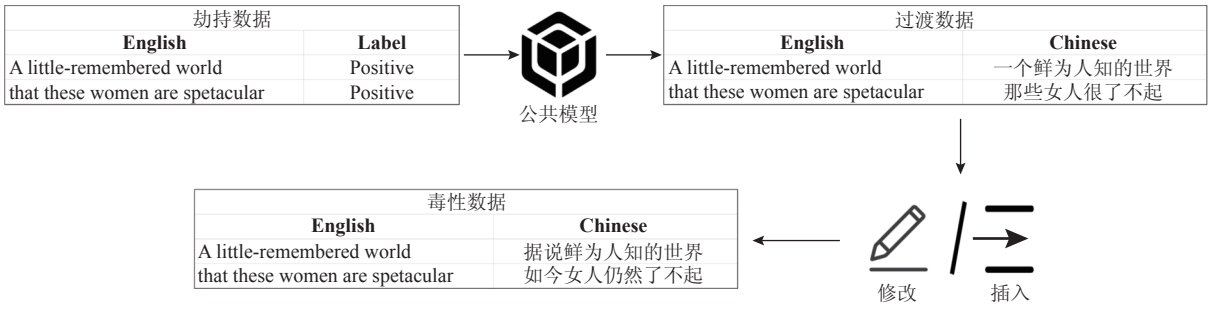


Fig. 2 The flow chart of preparation stage of Cheater attack

图 2 Cheater 攻击准备阶段流程图

$$S_{\text{sum}} = S_{\text{sem}} + S_{\text{hij}}, \tag{1}$$

$$S_{\text{sem}} = Distance(\boldsymbol{X}, \bar{\boldsymbol{X}}), \tag{2}$$

$$S_{\text{hij}} = \frac{count(\boldsymbol{X}_i, \boldsymbol{H}_i)}{|\boldsymbol{X}_i|}, \tag{3}$$

其中$\boldsymbol{X}$和$\bar{\boldsymbol{X}}$分别表示未插入逻辑词的语句和插入该逻辑词后的语句，$\boldsymbol{X}_i$和$\boldsymbol{H}_i$分别表示第i类的待处理语句和第i类的劫持词组，函数$count(\cdot)$用来统计在语句$\boldsymbol{X}_i$中存在的属于第$\boldsymbol{H}_i$劫持词组中的逻辑词个数。同时，为了能够对劫持数据集进行更加精细化的改造，我们还可以灵活地为式(1)中的S_{sem}和S_{hij}指标分配不同的权重系数，从而在劫持攻击的隐蔽性与成功率之间找到一个恰当的平衡点。在本文中，我们将S_{sem}和S_{hij}的权重系数分别设置为 0.7 和 0.3，这意味着在毒性数据集的生成过程中，我们更加注重毒性数据的流畅性与隐蔽性。

式(1)中语义指标S_{sem}的主要目的是检测使用逻辑词改写前后句子的语义差异。若差异过大，则意味着该逻辑词会大幅改变语义，致使改写后的句子隐蔽性欠佳，易被模型拥有者通过数据清洗等手段探测到，进而造成攻击失败。在具体实现上，函数$Distance(\cdot)$首先利用已有的语言模型对句子进行编码，之后根据两者编码的余弦相似度对其进行语义相似性衡量。嵌入指标S_{hij}的意义则是在合理的情况下，尽可能多地插入逻辑词，从而提升攻击者在实施阶段的攻击成功率。算法 1 给出了本步的详细实现过程。

算法 1. 数据集改造算法。

输入：过渡数据集中的语句$y = (y_0, y_1, \cdots, y_n)$，语句$\boldsymbol{y}$的类别所对应的劫持词组$\boldsymbol{H}_c$，最大迭代次数$T$；

输出：毒性语句$\bar{\boldsymbol{y}}$。

① 初始化：$\boldsymbol{y}^0 = \boldsymbol{y}$，$\bar{\boldsymbol{y}} = \boldsymbol{y}, s = 0$

② repeat

③ repeat

④ $i = 0$;

⑤ $Y_{\text{rep}} = Replace(\boldsymbol{y}^s, i, H_c)$; /*将$\boldsymbol{y}^s$中的$y_i$位置逐一使用劫持词组中的所有逻辑词进行替换，得到所有由这些经过改造的语句组成的数据集Y_{rep}*/

⑥ $Y_{\text{ins}} = Insert(\boldsymbol{y}^s, i, H_c)$; /*将$\boldsymbol{y}^s$中的$y_i$位置逐一使用劫持词组中的所有逻辑词进行插入，得到所有由这些经过改造的语句组成的数据集Y_{ins}*/

⑦ for each $\hat{\boldsymbol{y}}$ in $(Y_{\text{rep}} \cup Y_{\text{ins}})$ do

⑧ if $S_{\text{sum}}(\hat{\boldsymbol{y}}, \boldsymbol{y}) > S_{\text{sum}}(\bar{\boldsymbol{y}}, \boldsymbol{y})$

⑨ $\bar{\boldsymbol{y}} = \hat{\boldsymbol{y}}$;

⑩ end if

⑪ end for

⑫ $i{+}{+}$;

⑬ until 对$\boldsymbol{y}$中的每个词语遍历结束，即$i > n$;

⑭ $\boldsymbol{y}^s = \bar{\boldsymbol{y}}$;

⑮ $s{+}{+}$;

⑯ until $s = T$，即完成T次迭代；

⑰ return 改造完成后的语句$\bar{\boldsymbol{y}}$。

攻击者将劫持数据集转换为毒性数据集，且模型拥有者采用被污染的数据完成模型训练后，即可视作攻击者已成功劫持目标模型。

3.3 实施阶段

针对大语言模型的攻击方式，如后门攻击、对抗攻击等，都需要攻击者对模型的输入进行一定程度的修改，才能在目标模型上成功触发攻击。例如，后门攻击采用添加“触发器”等手段，将普通语句转化为恶意输入，以此触发后门来达到攻击目的。但在本文提出的劫持攻击中，攻击者无需对模型输入做任何的额外修改，因其触发模型异常动作的“恶意”语句无论是在外观上还是语法上都和普通输入没有任何差别。以上文提到的劫持任务为例，语句“I am suffering from the pain because of the disease。”既可以作为翻译任务的正常输入，也可以作为劫持任务的恶意语句。对于模型的输出结果，普通用户获取的是翻译答案，而攻击者经简单处理模型输出后，便能得到情感分类任务的分类结果。一旦劫持成功，模型拥有者难以从模型输入层面对该攻击进行防御。

在实施阶段中，只需要对模型输出中所含有的预定逻辑词进行计数，随后，利用准备阶段建立的映射关系对语句中所含逻辑词数量较多的劫持词组进行逆向推导，即可得出劫持任务的执行结果。但考虑到每个逻辑词在不同语境中的词频不同，比如在模型的某个输出结果中同时含有常见的“积极”类逻辑词和较为罕见的“消极”类逻辑词，相比之下，最终分类结果归为后者的概率更高。因此，每个逻辑词也应该根据词频不同被赋予不同权重。分配权重算法为：

$$W_a = 1 - \frac{count(a)}{|logicalwords|}, \tag{4}$$

其中函数$count(\cdot)$统计毒性数据集中逻辑词a的出现次数。在计算出所有逻辑词的词频后，便可得出词频表，攻击者可通过查询词频表的方式分别确定模型输出序列中所含逻辑词的得分情况及不同逻辑词所属的各劫持词组的总得分情况，并最终使用得分

较高的劫持词组进行反向映射,从而得到输入样本的最终分类结果。

4 实验评估

评估实验以面向中-英文文本的机器翻译任务作为模型原始任务,面向英文文本的情感分类任务作为劫持任务。通过测定攻击隐蔽性、攻击成功率和模型性能等指标来评估模型劫持攻击的性能,并通过改变部分参数设置,进一步探索超参数对攻击性能的影响。最后,对该攻击进行试探性防御评估。

4.1 实验设置

如表 1 所示,本次实验使用的数据集包括 SST-2 数据集[44] 和 WMT17 数据集[45]。前者是斯坦福大学推出的典型的二分类数据集,其中的内容由英文语句和情感极性标签构成,每个句子均有其对应的积极(0)/消极(1)类标签。后者是 WMT(workshop on statistical machine translation)推出的翻译数据集,其中包含德语-英语、俄语-英语、芬兰语-英语等多种翻译语料。在未额外声明的情况下,本次实验默认采用了 SST-2 中的 67 349 条情感分类数据作为劫持数据集。

Table 1 Abstract of Datasets

表 1 数据集摘要

数据集	训练集/测试集	说明
SST-2	67 349/1820	情感分类数据
WMT17	18 434 000/2001	翻译数据

本次评估实验采用 Bart[base], Bart[large], opus-mt-en-zh 作为基础模型。对于算法中的语句编码模型采用 shibing624/text2vec-base-chinese[46],因该模型可以轻松地将语句映射到 768 维的稠密向量空间以方便后续计算句子间的余弦相似度。模型的学习率设置为 0.000 03, batch_size 为 32, num_epoch 为 3。此外,由于不同的公共模型对 Cheater 的攻击影响有限,因此本研究在实验中选择了有道翻译作为公共模型。

4.2 评估标准

为了展现模型劫持攻击的攻击性能,需要使用模型性能(utility)、攻击成功率(attack success rate, ASR)和攻击隐蔽性(stealthiness)等 3 个指标。

1)模型性能。即被劫持模型相较于原模型在无毒数据集上的表现情况。本次评估实验以机器翻译任务中常见的 BLEU 评分为标准,分别衡量上述模型在 WMT17 测试集作为输入时的性能表现。但由于国际上的 BLEU 评分多针对英文文本,对于中文文本的 BLEU 评分标准目前学术界尚未达成一致性意见。因此,本次实验的 BLEU 评分计算细则如下:首先,去掉语句中的所有中文标点符号。之后,采用类似英文 BLEU 评分的计算方法,即将语句内容以字为单位划分。最后,使用 NLTK 库中带 smoothing 方程的 sentence_bleu 函数来计算 BLEU 评分。

2)攻击成功率。即使用毒性数据作为劫持模型的输入时,攻击者可以从劫持模型的输出中正确推断出劫持任务的分类结果的比例。需要说明的是,在本次评估实验中,将使用 SST-2 的验证集数据作为劫持模型的输入,因为 SST-2 的测试集没有标注样本情感极性,仅有训练集和验证集数据拥有情感极性标签。

3)攻击隐蔽性。即将毒性数据分别作为普通模型和劫持模型的输入时,上述模型输出之间的差异情况。对劫持攻击来说,两者之间的差距越小,证明毒性输出与普通输出越相似,即越不容易引起模型拥有者的怀疑。对该指标,仍采用与模型性能相同的 BLEU 评分测量方法来衡定。

4.3 评估结果

4.3.1 模型性能评估

在此实验中,改造算法中的迭代次数 T 默认设置为 5,之后再将已经改造完成的 SST-2 数据随机混入 7.69×10^6 条 WMT17 数据中以制成毒性数据集。然后,再分别使用毒性数据集和干净数据集对 3 个基础模型进行训练。最后,将 WMT17 测试集的数据作为上述模型的输入,并使用 BLEU 评分衡量上述劫持模型输出与 WMT17 数据集给出的参考翻译之间的差异。此外,我们还额外将 SST-2 数据集分别混入了 4.21×10^6 和 13.01×10^6 条的 WMT17 训练集中,以此进一步测试不同的数据投毒率下该攻击的表现情况。

图 3 展示了以 WMT17 测试集作为劫持模型输入得到的部分输出结果。可以看出,劫持模型输出与 WMT17 给出的参考翻译相比并未有大幅度背离,两者在语义上仍较为相近。

3 个劫持模型与原模型的性能表现对比情况如图 4 所示。可以看出,3 个不同的劫持模型与其对应的原模型之间的 BLEU 评分差值极小,最大不超过 0.7 分,这证明了模型劫持攻击对目标模型原任务表现的损害微乎其微。此外,就单个模型在不同数据投毒率下的性能表现而言,数据投毒率对劫持模型执行原始任务的性能影响较小,模型并不会因遭劫持而使原始任务的执行效率出现大幅下滑的情况。值得一提的是,劫持模型的 BLEU 评分较实际值略微

△：劫持模型输出
◯：WMT17参考输出

△：过去5年的另外两起类似案件表明，司机不会对此负责
◯：根据过去5年另外两起类似案件的结果，驾驶员将不承担责任

△：2016年，北海新项目的支出预计不到14亿美元，而过去5年平均为70亿美元
◯：2016年，北海新项目的支出预计不超过14亿美元，而过去5年的平均项目支出达到约70亿美元

△：研究人员在地球物理学杂志期刊上介绍了他们新的金星气候模型的结果
◯：研究人员在地球物理研究通讯期刊中发表了新的金星气候模型的模拟结果

Fig. 3 Outputs of WMT17 dataset refering translation and hijacking model

图 3 WMT17 数据集参考翻译与劫持模型输出

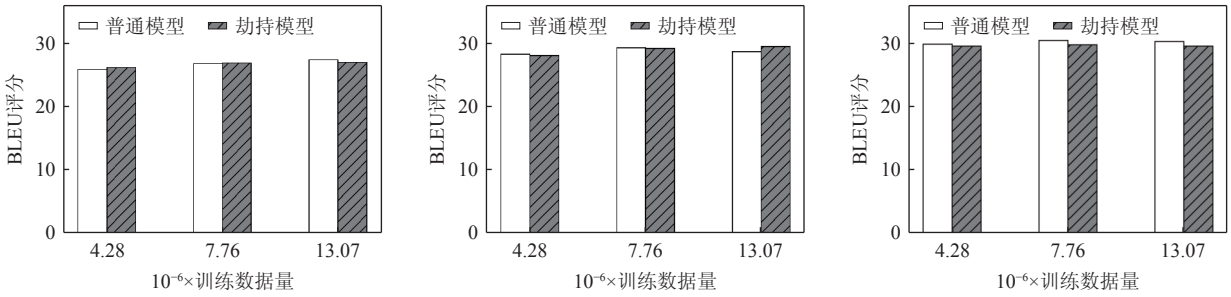


Fig. 4 Performance comparison of the normal model and the hijacked model

图 4 普通模型和劫持模型的性能对比

偏低，其原因主要在于 WMT17 测试集中的人名、地名等都以英文直译为主，而训练集中的人名、地名、特殊名称等都译为中文，因此训练出的模型在测试集上进行翻译时，会将上述提到的英文名称音译为中文，从而导致 BLEU 评分较实际值略微偏低。

4.3.2 攻击成功率评估

此实验采用 SST-2 验证集中的全部样本作为劫持模型的输入，根据模型输出中的逻辑词个数对 SST-2 中的对应样本进行试分类。同时，为了能够较为全面地展现模型劫持攻击的性能，仍旧使用不同数据量的 WMT17 训练数据作为超参数，进一步测试不同数据投毒率对攻击成功率的影响。

图 5 展示了 3 个模型在劫持任务下的攻击成功率情况。尽管被劫持模型的劫持任务和模型原始任务存在显著差异，但从图 5 中可以看出，即使是在数据投毒率低至 0.5% 的条件下，针对 Bart[large] 模型的劫持攻击也能达到 90.2% 的攻击成功率。相比之下，在基于停用词的针对德-英文模型劫持攻击中，Bart[base] 模型需要 0.69% 的数据投毒率才能达到 75.8% 的攻击成功率，而采用 Cheater 方法后，Bart[base] 模型仅需 0.5% 的数据投毒率即可达到 82.1% 的攻击成功率。值得注意的是，在 Opus-mt-en-zh 模型的实验中观察

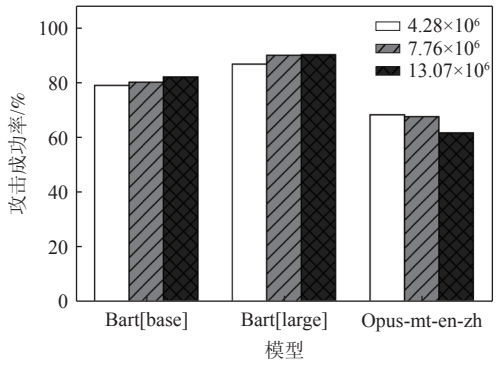


Fig. 5 Attack success rate of the hijacked models using different training datasets

图 5 使用不同训练数据集的劫持模型攻击成功率

到的攻击成功率随训练数据集规模增大而下降的现象，我们推测这主要源于该模型自身参数量较少导致的学习能力有限：当 WMT17 数据集规模扩大时，毒性数据占总训练数据的比例相应降低，投毒效果的减弱直接导致了攻击成功率下降。与此形成对比的是，Bart[base] 和 Bart[large] 模型在超参数实验中展现出的相反趋势，即数据投毒率越低，攻击成功率反而越高。我们认为其原因可能在于：更大的数据量包含了更多潜在的带有情感色彩的数据样本，使得模型在训练过程中能够更有效地习得更准确的情感

分类标准,进而提高了攻击成功率。

为验证这一推测,我们再次使用 4.26×10^6 条数据(4.21×10^6 条 WMT17 数据+0.052×10^6 条 SST-2 毒性数据)和 7.74×10^6 条数据(7.69×10^6 条 WMT17 数据+0.052×10^6 条 SST-2 毒性数据)训练了 2 个新 Bart[base]的模型,其结果如表 2 所示。从表 2 中可知,随着毒性数据的减少,攻击成功率出现较大幅度下降,因此可以推测出 WMT17 数据集增大导致了 Bart 系列模型实验中数据投毒率越低反而攻击成功率越高的反直觉现象。

Table 2 Attack Success Rates of Hijacked Model Under Different Poisoning Rates

表 2 不同投毒率下的劫持模型攻击成功率

投毒率/%	训练数据量	攻击成功率/%
1.5	4.28×10^6	79
1.2	4.26×10^6	71.5
0.867	7.76×10^6	80.1
0.869	7.74×10^6	70.8

4.3.3 攻击隐蔽性评估

此实验使用 SST-2 的测试集样本作为模型的输入。在获取模型输出结果后,运用 BLEU 评分方法对其进行评分。需要说明的是,本次评估采用公共模型对 SST-2 测试集进行翻译,并将所得结果作为 BLEU 评分中的参考翻译。此外,为了进一步探索超参数对攻击隐蔽性的影响,还分别以不同 WMT17 训练集大小和不同的迭代改造次数 T 作为超参数重复了上述实验,进一步测试了数据投毒率和改造算法中的迭代次数 T 对于实验的影响,实验结果如图 6 和表 3 所示。

从图 6 中可以看出,对于 Bart[base] 和 Bart[large]模型而言,劫持模型的 BLEU 评分均略微高于普通模型,而 Opus-mt-en-zh 模型的普通模型的 BLEU 评分则略高于对应的劫持模型。我们认为造成上述 2 种现象的原因是分别是因为 Opus-mt-en-zh 劫持模型在面对毒性输入时,其翻译序列中包含了过多的逻辑词,因此导致了其 BLEU 评分较普通模型而言相对较低;而对 Bart[base] 和 Bart[large] 模型而言,由于其普通模型使用的训练数据仅有 WMT17 数据,而劫持模型的训练数据则包括 WMT17 数据和使用公共模型翻译、改造而来毒性数据,因此劫持模型在面对毒性输入时,其 BLEU 评分相较普通模型而言稍有提升。但总体而言,各模型之间的得分差距不大,这表明攻击者使用恶意样本对劫持模型进行攻击时,劫持模型的输出结果与以普通样本作为输入时的模型输出结果相比,两者差距较小,不存在明显的特征差异。因此,模型拥有者难以发现两者之间的不同之处,换言之,模型劫持攻击具有较高的隐蔽性。

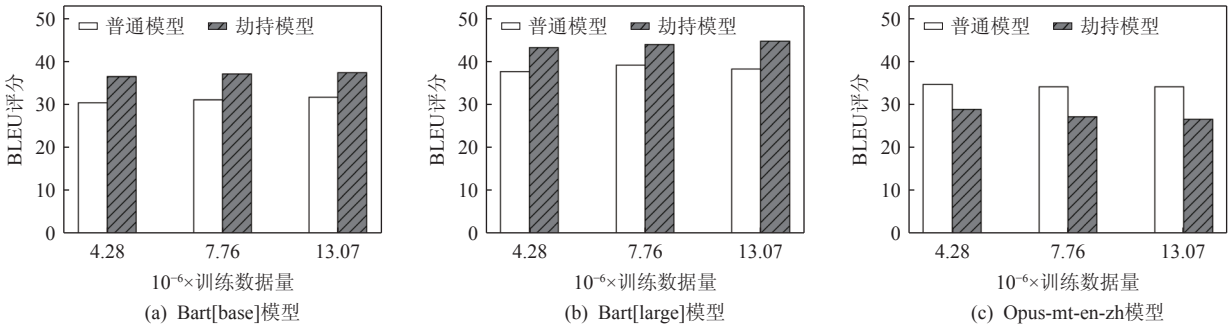


Fig. 6 Attack stealthiness of the normal model and the hijacked model

图 6 普通模型和劫持模型的攻击隐蔽性

Table 3 Attack Stealthiness of the Hijacked Models Under Different Iterations

表 3 不同迭代次数下的劫持模型攻击隐蔽性

模型	训练数据量	迭代次数	BLEU 评分
Bart[base]	7.76×10^6	3	37.6
Bart[base]	7.76×10^6	5	36.8

表 3 是超参数实验中不同迭代次数对模型影响的实验结果,综合表 3 和图 6 可以得到 2 个推论。其一,迭代次数越高,攻击隐蔽性越低(即 BLEU 评分越低)。因为对于 SST-2 的改动越大,其语义流畅性就越差,进而增加被模型拥有者察觉的风险。因此,攻击者需要在隐蔽性和成功率之间权衡,寻找两者的平衡点。其二,数据投毒率越低,攻击隐蔽性越高。主要原因在于改造后的 SST-2 数据在语义流畅度上有所降低,而这种降低将会随着 SST-2 在训练数据中所占比重的下降而减小,即 WMT17 数据的相对占比越大,攻击隐蔽性也会越高。

4.3.4 防御评估

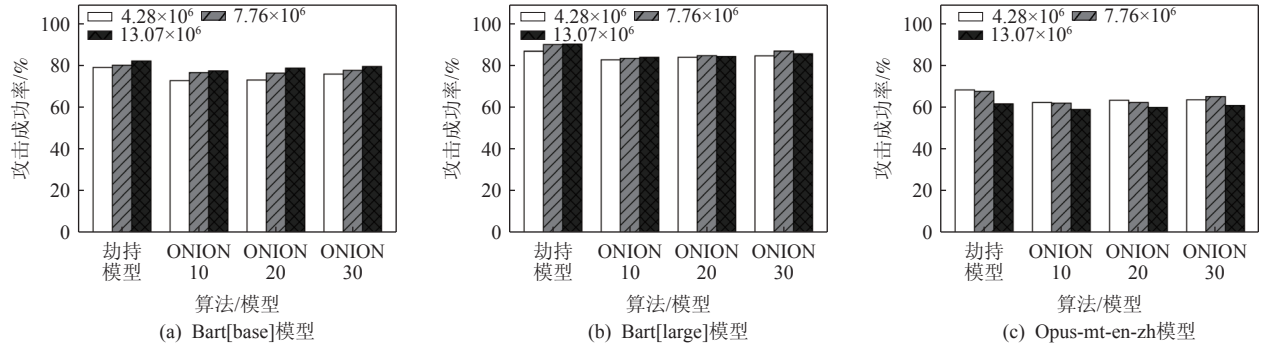
由于目前尚未提出针对模型劫持攻击进行防御的可行算法，因此我们选择使用泛用性较高的ONION算法和RAP算法对本文提出的劫持攻击进行试探性防御。但由于RAP算法最初是针对分类模型而提出的防御算法，其需要通过计算原始样本和扰动样本的置信度之差是否小于阈值来判断该样本是否为毒性样本。而本文的主要研究对象为机器翻译模型，其输出中并不包含置信度信息，故无法直接使用RAP算法对Cheater进行防御。因此，为了使RAP算法能够应用于机器翻译模型，我们在保持其防御思想不变的前提下对RAP算法进行了改造：首先，在原始样本的随机位置中插入一个上下文无关的单词以构建扰动样本；然后，将原始样本和扰动样本送入劫持模型，得到$Text_{\text{original}}$和$Text_{\text{modified}}$；最后，计算$Text_{\text{original}}$和$Text_{\text{modified}}$之间的文本相似度，将相似度之差小于ts的样本视作毒性样本，禁止其进入模型。阈值ts的计算公式为

$$ts = \frac{1}{Length} + 0.1\,, \tag{5}$$

其中$Length$为原始样本中的单词数量。在本实验中，我们将RAP算法中的固定阈值修改为式(5)中的动态阈值ts，这是由于动态阈值既能避免对短文本的过度惩罚，又能减少固定阈值对长文本语义变化的过度容忍。

图7展示了ONION算法在不同的困惑度变化阈值下模型劫持攻击的攻击成功率变化情况。从图中结果可以看出，ONION算法对模型劫持攻击的防御效果较差，即使将困惑度阈值设置为10，仍旧只能降低最高6.6个百分点左右的攻击成功率。

图8展示了应用RAP算法后的模型劫持攻击成功率。从图8中我们可以看出，RAP算法对模型劫持攻击的防御效果虽然略微优于ONION算法，但总体效果仍不理想。具体而言，对Bart[base], Bart[large], Opus-mt-en-zh模型，其攻击成功率的平均降幅分别仅为23.16个百分点、21.73个百分点、18.13个百分点。

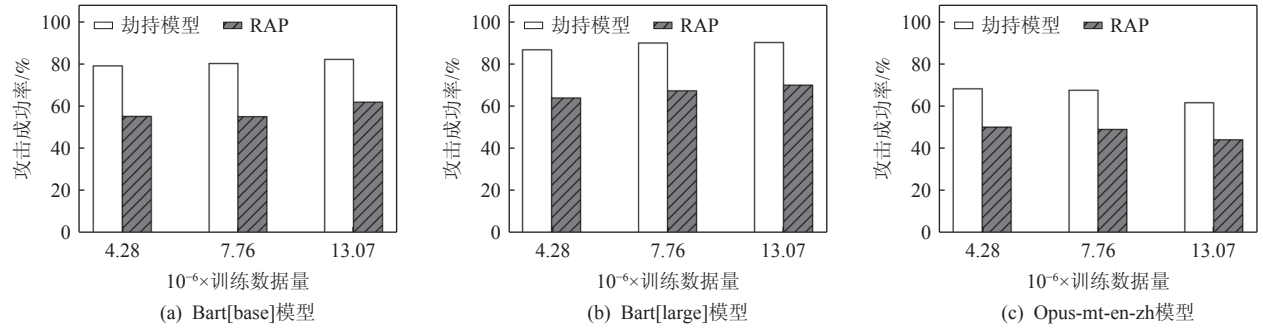


(a) Bart[base]模型 (b) Bart[large]模型 (c) Opus-mt-en-zh模型

注：括号内的数值为困惑度阈值。

Fig. 7 Attack success rates of the hijacked model after applying ONION algorithm for defense

图7 采用ONION算法对劫持模型进行防御后的攻击成功率



(a) Bart[base]模型 (b) Bart[large]模型 (c) Opus-mt-en-zh模型

Fig. 8 Attack success rates of the hijacked model after applying RAP algorithm for defense

图8 采用RAP算法对劫持模型进行防御后的攻击成功率

5 总结与展望

本文提出了一种利用逻辑词作为指示词的针对中-英文大语言模型可行的模型劫持攻击方法，并且通过在Bart[base], Bart[large], Opus-mt-en-zh模型上对各项攻击指标进行评测，证明了其有效性与隐蔽性。实验结果显示，在采用Cheater对中-英文模型实施劫

持攻击时,其展现出了超越针对德-英文的模型劫持攻击的性能。具体而言,在仅有 0.5% 的数据投毒率的情况下,Bart[base]模型即可达到 82.1% 的攻击成功率,而针对德-英文 NLP 模型的劫持攻击在 0.69% 的数据投毒率下仅实现了 75.8% 的攻击成功率。

鉴于模型劫持攻击作为近年来新兴的威胁类型,现有主流防御算法(如 ONION 和 RAP)难以有效抵御,且学界目前尚缺乏针对性的有效防御方案。因此,我们的未来研究将聚焦于如何拓展模型劫持攻击的泛化应用场景(如文本摘要和问答系统),并探索基于动态指示词选择的模型劫持攻击策略。同时,为更全面地评估 Cheater 攻击在中文语境下的隐蔽性及其对语义的影响,我们计划引入基于语义嵌入的指标(如 BERTScore)和中文特定的句法一致性指标(如基于依存句法分析),以补充 BLEU 指标的局限性,深入分析 Cheater 对翻译文本语义和句法结构的潜在影响。此外,我们还将从多方面进行尝试,致力于构建出能有效抵御此类攻击的新型防御机制。

作者贡献声明:钟一负责方案整体设计并撰写论文;陈珍珠负责部分算法思路并撰写论文;付安民、高艳松提出指导意见并修改论文。

参 考 文 献

[1] Wu Tianyu, He Shizhu, Liu Jingping, et al. A brief overview of ChatGPT: The history, status quo and potential future development[J]. IEEE/CAA Journal of Automatica Sinica, 2023, 10(5): 1122−1136

[2] Achiam J, Adler S, Agarwal S, et al. GPT-4 technical report[J]. arXiv preprint, arXiv: 2303.08774, 2023

[3] Team G, Anil R, Borgeaud S, et al. Gemini: A family of highly capable multimodal models[J]. arXiv preprint, arXiv: 2312.11805, 2023

[4] Zhou Ming, Duan Nan, Liu Shujie, et al. Progress in neural NLP: Modeling, learning, and reasoning[J]. Engineering, 2020, 6(3): 275−290

[5] Ju Yiming, Ma Huanhuan. Training data for large language model[J]. arXiv preprint, arXiv: 2411.07715, 2024

[6] Si Waiman, Backes M, Zhang Yang, et al. Two-in-One: A model hijacking attack against text generation models[C]//Proc of the 32nd USENIX Security Symp. Berkeley, CA: USENIX Association, 2023: 2223−2240

[7] Salem A, Backes M, Zhang Yang. Get a model! Model hijacking attack against machine learning models[C/OL]//Proc of the 28th Symp on Network and Distributed System Security. Reston, VA: The Internet Society, 2022[2025-07-31]. https://publications.cispa.saarland/3637/

[8] Deng Yue, Zhang Wenxuan, Pan S J, et al. Multilingual jailbreak challenges in large language models[J]. arXiv preprint, arXiv: 2310.06474, 2023

[9] Zhou Jixu. Comparison of Chinese and Indo-European Vocabulary[M]. Chengdu: Sichuan National Publishing House, 2002 (in Chinese)
(周及徐. 汉语印欧语词汇比较[M]. 成都:四川民族出版社, 2002)

[10] Qin Zhen, Zhuang Tianming, Zhu Guosong, et al. Survey of security attack and defense strategies for artificial intelligence model[J]. Journal of Computer Research and Development, 2024, 61(10): 2627−2648 (in Chinese)
(秦臻, 庄添铭, 朱国淞, 等. 面向人工智能模型的安全攻击和防御策略综述[J]. 计算机研究与发展, 2024, 61(10): 2627−2648)

[11] Zhu Chen, Huang W R, Li Hengduo, et al. Transferable clean-label poisoning attacks on deep neural nets[C]//Proc of the 36th Int Conf on Machine Learning. New York: PMLR, 2019: 7614−7623

[12] Fung C, Yoon C J M, Beschastnikh I. The limitations of federated learning in sybil settings[C]//Proc of the 23rd Int Symp on Research in Attacks, Intrusions and Defenses. Berkeley, CA: USENIX Association, 2020: 301−316

[13] Shafahi A, Huang W R, Najibi M, et al. Poison frogs! Targeted clean-label poisoning attacks on neural networks[C]//Proc of the 32nd Conf on Neural Information Processing Systems. Cambridge, MA: MIT, 2018: 6106−6116

[14] Peri N, Gupta N, Huang W R, et al. Deep k-NN defense against clean-label data poisoning attacks[C]//Proc of the Computer Vision–ECCV 2020 Workshops. Berlin: Springer, 2020: 55−70

[15] Chen Yufei, Shen Chao, Wang Qian, et al. Security and privacy risks in artificial intelligence systems[J]. Journal of Computer Research and Development, 2019, 56(10): 2135−2150 (in Chinese)
(陈宇飞, 沈超, 王骞, 等. 人工智能系统安全与隐私风险[J]. 计算机研究与发展, 2019, 56(10): 2135−2150)

[16] Tolpegin V, Truex S, Gursoy M E, et al. Data poisoning attacks against federated learning systems[C]//Proc of the 25th European Symp on Research in Computer Security. Berlin: Springer, 2020: 480−501

[17] Goldblum M, Tsipras D, Xie Chunlin, et al. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 45(2): 1563−1580

[18] Bagdasaryan E, Veit A, Hua Yiqing, et al. How to backdoor federated learning[C]//Proc of the 23rd Int Conf on Artificial Intelligence and Statistics. New York: PMLR, 2020: 2938−2948

[19] Shejwalkar V, Houmansadr A. Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning[C]//Proc of the 27th Symp on Network and Distributed System Security. Reston, VA, USA: The Internet Society, 2021: 1−19

[20] Wan A, Wallace E, Shen Sheng, et al. Poisoning language models during instruction tuning[C]//Proc of the 40th Int Conf on Machine

Learning. New York: PMLR, 2023: 35413−35425

[21] Zheng Mingyu, Lin Zheng, Liu Zhengxiao, et al. Survey of textual backdoor attack and defense[J]. Journal of Computer Research and Development, 2024, 61(1): 221−242 (in Chinese)
(郑明钰, 林政, 刘正宵, 等. 文本后门攻击与防御综述[J]. 计算机研究与发展, 2024, 61(1): 221−242)

[22] Liu Yingqi, Ma Shiqing, Aafer Y, et al. Trojaning attack on neural networks[C]//Proc of the 25th Symp on Network and Distributed System Security. Reston, VA: The Internet Society, 2018: 1−15

[23] Liu Jialang, Guo Yanming, Lao Mingrui, et al. Survey of backdoor attack and defense algorithms based on federated learning[J]. Journal of Computer Research and Development, 2024, 61(10): 2607−2626 (in Chinese)
(刘嘉浪, 郭延明, 老明瑞, 等. 基于联邦学习的后门攻击与防御算法综述[J]. 计算机研究与发展, 2024, 61(10): 2607−2626)

[24] Chen Xiaoyi, Salem A, Chen Dingfan, et al. BadNL: Backdoor attacks against NLP models with semantic-preserving improvement[C]//Proc of the 37th Annual Computer Security Applications Conf. Piscataway, NJ: IEEE, 2021: 554−569

[25] Sun Lichao. Natural backdoor attack on text data[J]. arXiv preprint, arXiv: 2006.16176, 2020

[26] Kurita K, Michel P, Neubig G. Weight poisoning attacks on pretrained models[C]//Proc of the 58th Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA: ACL, 2020: 2793−2806

[27] Dai Jiazhu, Chen Chuanshuai. A backdoor attack against LSTM-based text classification systems[J]. arXiv preprint, arXiv: 1905. 12457, 2019

[28] Zügner D, Akbarnejad A, Günnemann S. Adversarial attacks on neural networks for graph data[C]//Proc of the 24th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2018: 2847−2856

[29] Dong Yinpeng, Liao Fangzhou, Pang Tianyu, et al. Boosting adversarial attacks with momentum[C]//Proc of the 31st IEEE Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2018: 9185−9193

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

[31] Sharif M, Bhagavatula S, Bauer L, et al. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition[C]//Proc of the 23rd ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2016: 1528−1540

[32] Zhao Yue, Zhu Hong, Liang Ruigang, et al. Seeing isn't believing: Towards more robust adversarial attack against real world object detectors[C]//Proc of the 26th ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2019: 1989−2004

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

[34] Chakraborty A, Alam M, Dey V, et al. A survey on adversarial attacks and defences[J]. CAAI Transactions on Intelligence Technology, 2021, 6(1): 25−45

[35] Jia R, Liang P. Adversarial examples for evaluating reading comprehension systems[C]//Proc of the 2017 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2017: 2021−2031

[36] Zou A, Wang Zifan, Carlini N, et al. Universal and transferable adversarial attacks on aligned language models[J]. arXiv preprint, arXiv: 2307.15043, 2023

[37] Rando J, Tramèr F. Universal jailbreak backdoors from poisoned human feedback[J]. arXiv preprint, arXiv: 2311.14455, 2023

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

[39] Ribeiro M T, Singh S, Guestrin C. Semantically equivalent adversarial rules for debugging NLP models[C]//Proc of the 56th Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA: ACL, 2018: 856−865

[40] Wei A, Haghtalab N, Steinhardt J. Jailbroken: How does LLM safety training fail[C]//Proc of the 37th Annual Conf on Neural Information Processing Systems. Cambridge, MA: MIT, 2024: 80079−80110

[41] Qi Fanchao, Chen Yangyi, Li Mukai, et al. ONION: A simple and effective defense against textual backdoor attacks[C]//Proc of the 2021 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2021: 9558−9566

[42] Yang W, Lin Yankai, Li Peng, et al. RAP: Robustness-aware perturbations for defending against backdoor attacks on NLP models[C]//Proc of the 2021 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2021: 8365−8381

[43] Sun Junyi. "Jieba" (Chinese for "to stutter") Chinese text segmentation: Built to be the best Python Chinese word segmentation module[CP/OL].[2025-01-05]. https://github.com/fxsjy/jieba

[44] Socher R, Perelygin A, Wu J, et al. Recursive deep models for semantic compositionality over a sentiment treebank[C]//Proc of the 2013 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2013: 1631−1642

[45] Ondrej B, Rajen C, Federmann C, et al. Findings of the 2017 Conference on Machine Translation (WMT17)[C]//Proc of the 2017 Conf on Machine Translation. Stroudsburg, PA: ACL, 2017: 169−214

[46] Xu Ming. Text2vec: A tool for text to vector[CP/OL].[2024-12-17]. https://github.com/shibing624/text2vec



Zhong Yi, born in 2001. Master candidate. His main research interest includes AI security.
钟　一, 2001 年生。硕士研究生。主要研究方向为 AI 安全。



Chen Zhenzhu, born in 1993. PhD, lecturer. Her main research interests include AI security and privacy protection.
陈珍珠, 1993 年生。博士, 讲师。主要研究方向为人工智能安全、隐私保护



Fu Anmin, born in 1981. PhD, professor, PhD supervisor. Senior member of CCF. His main research interests include cryptography and privacy preserving.

付安民, 1981 年生。博士，教授，博士生导师，CCF 高级会员。主要研究方向为密码学、隐私保护。



Gao Yansong, born in 1986. PhD, associate professor. His current research interests include hardware security, AI security and privacy, and system security.

高艳松, 1986 年生。博士，副教授。主要研究方向为硬件安全、人工智能安全和隐私、系统安全。