Beijing Forest Studio

北京理工大学信息系统及安全对抗实验中心

机生文本检测中的对抗攻击与鲁棒防御

硕士研究生 邢倚康

2026年7月19日

问题回溯

- 总结反思
  - 算法原理讲解需要更加深入
  - 讲解时的语速、停顿等需要进一步提升
- 相关内容
  - 2026.01.03 邢倚康《从生成机制探索机生文本检测新方法》
  - 2025.01.05 刘佳《人工智能生成内容检测》

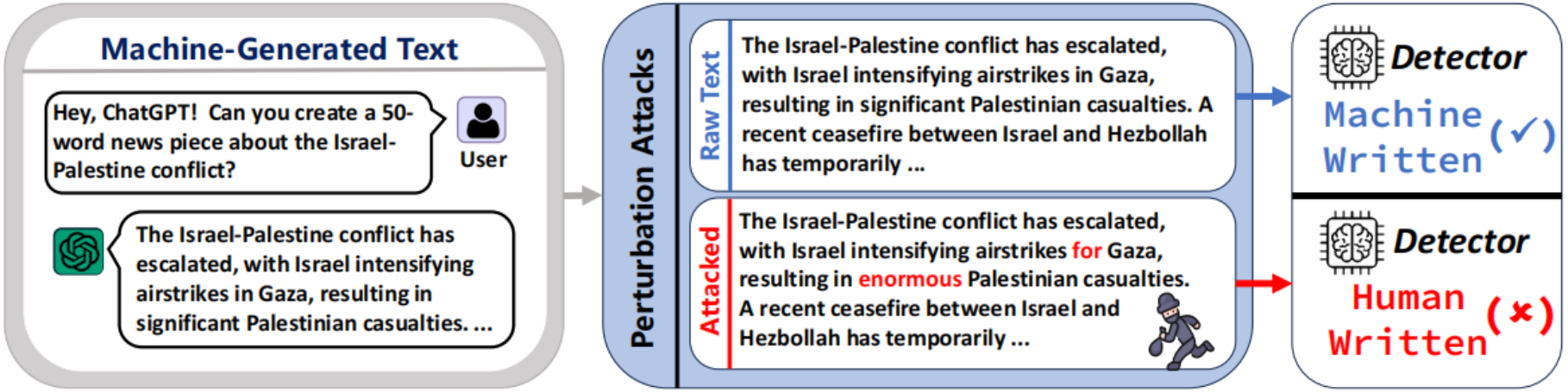
内容提要

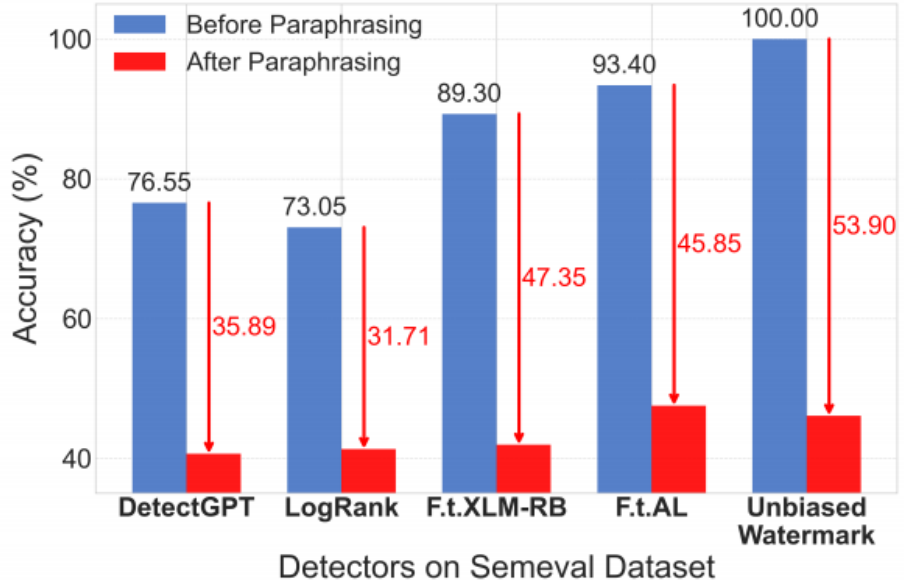
- 预期收获
- 内容引入
- 内涵解析与研究目标
- 研究背景与意义
- 研究历史与现状
- 知识基础
- 算法原理
  - Paraphrase Inversion
  - GREATER
- 特点总结与未来展望
- 参考文献

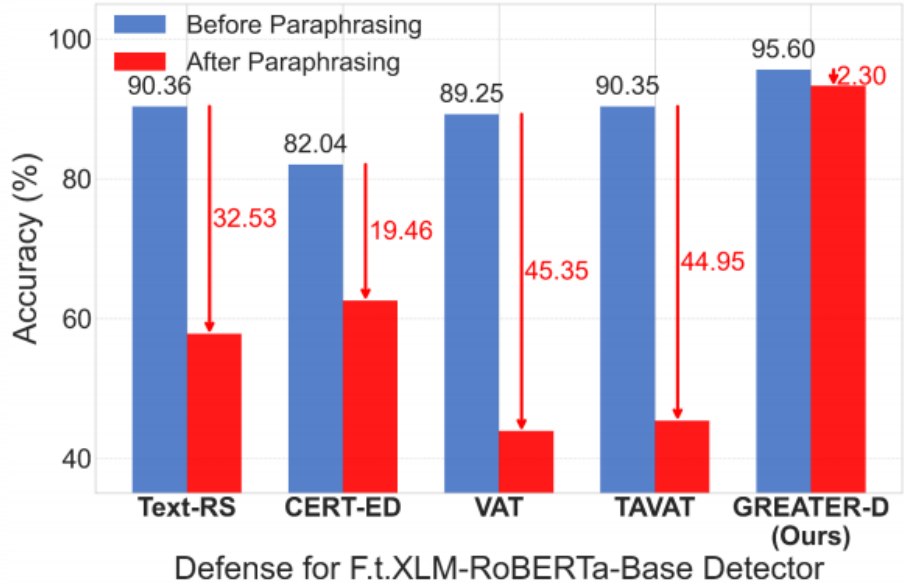
预期收获

- 预期收获
  - 掌握机生文本检测对抗攻击的基本概念与常见方法
  - 理解机生文本检测鲁棒防御的常见方法及原理
  - 了解机生文本检测鲁棒防御的未来发展方向

内容引入







语义基本不变，微小扰动即可逃避检测

攻击形式多样，现有防御泛化能力有限

如何提升检测器对抗鲁棒性?

内涵解析与研究目标

- 内涵解析
  - 机生文本检测：对大模型生成的文本进行识别，区分人类撰写文本与机器生成文本
  - 对抗攻击：在保持文本语义、可读性与自然性的前提下，通过释义改写、词语替换等方式改变文本表层特征，诱导检测器将机器文本误判为人类文本
  - 鲁棒防御：提升检测方法对文本扰动和未知攻击的抵抗能力，使其在攻击前后仍能保持稳定、可靠的检测结果

- 研究目标
  - 面向大模型时代的机生文本检测攻防对抗问题
  - 结合深度学习、自然语言处理等技术
  - 提高检测器面对已知攻击、未知攻击及不同攻击强度时的鲁棒性与泛化能力，为学术诚信、内容审核和媒体平台治理提供更加可靠的检测能力

研究背景与意义

- 研究背景
  - 大语言模型广泛应用，机生文本已大量出现在新闻写作、社交媒体、学术写作、客服对话等场景，生成内容规模持续增长
  - 机生文本带来了学术不端、虚假信息传播等风险，而复杂提示词、释义改写等攻击进一步加剧了风险，迫切需要更加鲁棒、可靠的识别与治理手段
  - 现有机生文本检测方法多依赖语言模型概率、词汇分布等统计特征，而语义保持的释义改写与局部文本扰动能够破坏这些判别依据，显著降低检测性能
- 研究意义
  - 将机生文本检测拓展到真实开放环境中的对抗鲁棒检测，增强检测方法面对文本改写和恶意规避行为时的稳定性
  - 支撑可信内容生态与安全治理，为学术诚信审查、内容审核、媒体平台治理等提供更加可靠的技术支撑，降低恶意逃避检测带来的风险

研究历史与现状

Krishna等提出Retrieval-based Defense，通过匹配语义相似生成记录来识别被改写后的机器文本

Hu等提出RADAR，通过释义模型与检测器的对抗训练，使检测器逐步适应更强的改写攻击

Huang等提出SCRN，利用噪声重构与孪生校准学习稳定语义表示，增强检测器对字符级、词级扰动的鲁棒性

Rivera Soto等提出Paraphrase Inversion，先判断文本是否被释义改写，再尝试恢复接近原始文本的表达，缓解改写攻击

Li等提出GREATER，由攻击器动态生成对抗样本，再训练检测器，实现攻击生成与鲁棒防御的同步增强

2023

Hou等提出SemStamp，从token级水印转向句子级语义水印，利用语义空间划分增强对释义改写攻击的鲁棒性

2023

Ren等提出SemaMark，利用语义信息替代简单token哈希，使水印在释义改写攻击后仍能被检测

2024

Hou等进一步提出k-SemStamp，用聚类方式优化语义水印空间划分，在保持生成质量的同时提升抗释义能力与采样效率

2025

Shen等提出SEEK，通过等价纹理密钥增强文本水印，在抵抗水印擦除攻击的同时提升对伪造攻击的鲁棒性

机生文本检测鲁棒防御

- 白盒防御
- 黑盒防御

对抗攻击与鲁棒防御的持续博弈

- 基于统计概率分布特征
  - 核心思想：不额外训练检测器，利用预训练语言模型计算文本中每个 token 的出现概率，判断文本是否为机器生成
  - 常见特征：困惑度、Token Rank分布、词频信息统计等
  - 机生文本通常更流畅、更可预测，在语言模型下表现出更高概率、更低困惑度、更集中的 token 分布
  - 代表方法：DetectGPT、Fast-DetectGPT、Binoculars 等
  - 优势：部署成本低，可用于零样本检测，对特定生成模型依赖较小
  - 面对攻击失效的原因：这类方法主要捕捉表层统计规律，而释义改写、同义词替换、提示词优化等攻击可以在保持语义不变的同时重塑词汇、句法和概率特征，改变 token 概率分布，使机器文本更接近人类文本分布

知识基础　机生文本检测常见方法

- 基于监督训练与特征学习
  - 核心思想：利用已标注的人类文本、机生文本数据，训练一个分类模型学习区分边界，从而对新文本进行检测
  - 常见思路
    - 利用BERT/RoBERTa 等预训练语言模型提取文本特征，如语义特征、风格特征等
    - 利用代理大语言模型（如GPT-Neo-2.7B，自回归模型，可输出token 级概率）提取特征
  - 代表方法：RoBERTa、DeTeCtive、Ghostbuster等
  - 优势：检测准确率更高，能够融合多层语义与风格特征，对复杂文本的表达能力更强，如多分类任务
  - 面对攻击失效的原因：模型容易学习到训练集中的生成器特征和领域特征，而非稳定的“机生文本本质特征”；改写、翻译回译、润色等攻击会弱化训练数据中的机器生成痕迹，造成分布偏移

知识基础　机生文本检测对抗攻击常见方法

- 常见攻击方法
  - 释义改写攻击：使用 LLM 或专门改写模型重写文本，在保持语义的同时改变词汇、句法和风格，如 DIPPER 改写、ChatGPT 改写、段落级重写
  - 同义词替换攻击：识别关键 token，并替换为语义相近词，改变检测器依赖的局部特征，如 TextFooler、BERT-Attack 类词级攻击
  - 字符级扰动攻击：通过拼写错误、大小写变化、标点插入、空格扰动等破坏 token 化结果，如 typo attack、字符插入、符号替换
  - 翻译回译攻击：将文本翻译到另一种语言后再翻译回来，获得语义相近但表达不同的文本，削弱原始生成模型留下的概率和风格痕迹
  - 提示词优化攻击：在生成阶段使用复杂提示词，引导模型模仿人类风格、加入不规则表达或降低机器痕迹
  - 其它攻击

Mitigating Paraphrase Attacks on Machine-Text Detectors via Paraphrase Inversion

Paraphrase Inversion TIPO

T	目标	在存在释义改写攻击的场景下，区分人类撰写文本与机器生成文本
I	输入	待检测文本 数据集：Reddit（443970，53432，6803）、RAID-ArXiv（48035，3798，1500）、RAID-MovieReviews（25649，1329，1500）
P	处理	1. 构造“原始文本—释义攻击文本”训练对，训练逆变换模型 2. 对输入文本进行改写分析，判断其是否经过释义改写 3. 若判定为改写文本，则生成多个逆变换候选文本并选择最优结果 4. 将恢复后的文本输入原有检测器进行判断
O	输出	待测文本是否属于机器生成文本

P	问题	1.释义攻击会改变词汇、概率分布和写作风格等，导致原有检测器失效 2.现有防御方法通常依赖专门训练检测器或访问历史生成结果，通用性不足
C	条件	需要原始文本语料和释义模型生成训练数据
D	难点	1.如何从释义后的文本中恢复原始文本的统计特征与风格特征 2.如何判断何时执行逆变换，避免对未改写文本造成误修改
L	水平	2025 CCF A类

Paraphrase Inversion 创新点分析 原始文本恢复

- 现有方法存在问题
  - 释义攻击在保持语义基本不变的同时，会改变文本的词汇选择、句法结构、token 概率分布和写作风格等，攻击文本容易被误判为人类文本
  - 现有方法没有充分利用释义攻击本身是一种可学习文本变换这一特点
- 解决方法
  - 将检测任务转化为文本生成任务，尝试将攻击文本恢复成改写前的原始文本，而这种类似文本风格迁移的文本生成任务在之前已经有很多研究
    - 原始文本 → 释义模型 → 攻击文本，释义攻击本质上是一种文本变换
    - 攻击文本 → 逆变换模型 → 近似原文，检测不要求逐字恢复原文
  - 对人类文本进行逆变换时，模型通常改动较少；对攻击文本进行逆变换时，模型通常会做出更多修改
  - 通过输入文本与逆变换结果之间的编辑距离进行判断

Paraphrase Inversion 创新点分析 前置改写分析模块

- 现有方法存在问题
  - 如果对所有输入文本都直接执行逆变换，会带来新的风险
  - 对于人类文本、机生文本等非改写攻击文本，模型可能会无意义地修改原文，破坏原本的风格和语义细节
- 解决方法
  - 将原始文本恢复作为一个前置模块，在原有检测器的基础上实现增强
    - 输入文本 → 改写分析 → 是否进行逆变换 → 检测器
  - 两种改写分析方式
    - 利用神经网络分类模型判断是否经过改写
    - 通过输入文本与逆变换结果之间的编辑距离进行判断
  - 当改写检测置信度高时，进行逆变换后再执行检测
    - 攻击文本被误判为未改写，仍有几率被后续检测器检测
    - 人类文本被误判为改写，大概率检测错误

实验设计　数据资源

- 数据集
  - Reddit (443970，53432，6803)
  - RAID-ArXiv (48035，3798，1500)、RAID-MovieReviews (25649，1329，1500)
- 释义改写攻击方式
  - Reddit：Mistral-7B、Phi-3、Llama-3.1-8B
  - RAID-ArXiv、RAID-MovieReviews：DIPPER
- 对比方法

  OpenAI Detector (2019) 、Rank (2019) 、LogRank (2019) 、Entropy (2020) 、RADAR (2023) 、FastDetectGPT (2024) 、Binoculars (2024)
- 评价指标
  - 机生文本检测性能： AUROC
  - 文本生成质量： BLEU、Style Similarity、Semantic Similarity

实验设计　对比实验

- 评估 Paraphrase Inversion 在不同数据集上的表现
  - 基线模型
  - 通过输入文本与逆变换结果之间的编辑距离判断是否经过改写
  - 利用神经网络分类模型判断是否经过改写

- RADAR 本身就是针对释义攻击设计的鲁棒检测器，但加入论文模块后仍能继续提升

Detector	AUROC		
	Baseline	Inversion+Edit-based	Inversion+Neural
Reddit			
OpenAI (2019)	0.56	0.77	**0.79**
Rank (2019)	0.56	0.66	**0.68**
LogRank (2019)	0.58	0.74	**0.77**
Entropy (2020)	0.51	**0.59**	**0.59**
RADAR (2023)	0.62	0.66	**0.70**
FastDetectGPT (2024)	0.66	0.80	**0.84**
Binoculars (2024)	0.77	0.84	**0.89**
RAID-ArXiv			
OpenAI (2019)	**0.81**	0.79	0.77
Rank (2019)	0.71	0.69	**0.79**
LogRank (2019)	0.75	0.72	**0.91**
Entropy (2020)	0.39	0.42	**0.62**
RADAR (2023)	**0.99**	0.98	**0.99**
FastDetectGPT (2024)	0.83	0.78	**0.91**
Binoculars (2024)	0.92	0.86	**0.98**
RAID-MovieReviews			
OpenAI (2019)	0.82	0.77	**0.83**
Rank (2019)	0.60	0.76	**0.84**
LogRank (2019)	0.66	0.84	**0.91**
Entropy (2020)	0.39	0.63	**0.71**
RADAR (2023)	0.92	0.92	**0.95**
FastDetectGPT (2024)	0.74	0.80	**0.89**
Binoculars (2024)	0.91	0.92	**0.96**

实验设计　对比实验

- 跨领域泛化实验
  - 验证在一个领域中训练的释义检测器和逆变换模型，能否迁移到另一个领域
  - 在 RAID-MovieReviews 上训练，在 RAID-ArXiv 上测试
  - 在 RAID-ArXiv 上训练，在 RAID-MovieReviews 上测试

- 论文方法在跨领域场景中依旧表现出色，具有良好的泛化性

Detector	AUROC	
	Baseline	Inversion
Train - RAID-MovieReviews, Eval - RAID-ArXiv		
OpenAI (2019)	0.81	**0.84**
Rank (2019)	0.71	**0.83**
LogRank (2019)	0.75	**0.89**
Entropy (2020)	0.39	**0.68**
RADAR (2023)	**0.99**	**0.99**
FastDetectGPT (2024)	0.83	**0.90**
Binoculars (2024)	0.92	**0.96**
Train - RAID-ArXiv, Eval - RAID-MovieReviews		
OpenAI (2019)	**0.82**	**0.82**
Rank (2019)	0.60	**0.83**
LogRank (2019)	0.66	**0.90**
Entropy (2020)	0.39	**0.68**
RADAR (2023)	0.92	**0.94**
FastDetectGPT (2024)	0.74	**0.87**
Binoculars (2024)	0.91	**0.95**

实验设计　其它实验

- 机生文本释义逆变换质量实验
- 未见模型（GPT-4）释义改写攻击泛化实验
  - 测试逆变换模型能否处理训练阶段没见过的释义模型
  - 使用 GPT-4 对 Reddit 机生文本进行释义，并使用在 Reddit 上训练的逆变换模型进行恢复

Method	Type	Machine-written Text		
		Style (↑)	Meaning (↑)	BLEU (↑)
Paraphrases	-	0.80	0.88	0.17
				Baselines
GPT-4	Single	0.80	0.85	0.20
	Max	0.86	0.90	0.33
	Mean	0.80	0.87	0.21
out2prompt	Single	0.48	0.17	0.00
	Max	0.71	0.40	0.04
	Mean	0.48	0.17	0.00
				Ours
Inversion	Single	0.84	0.90	0.34
	Max	**0.91**	**0.95**	**0.51**
	Mean	0.84	0.90	0.35

Method	Type	Style Sim. (↑)	Semantic Sim. (↑)	BLEU (↑)
Paraphrases	-	0.61	0.90	0.21
Inversion	Single	0.62	0.88	0.26
	Maximum	**0.77**	**0.94**	**0.41**
	Average	0.62	0.88	0.26

算法总结 Paraphrase Inversion

- 算法贡献
  - 原始文本恢复
    - 将释义攻击防御问题转化为文本生成任务，学习从攻击文本到近似原文的逆向映射，恢复原始文本中的统计特征、风格特征和机器生成痕迹
  - 前置改写分析模块
    - 在原有检测器前加入“改写分析—选择性恢复”流程，作为可插拔前置模块，不依赖特定下游检测器，可在保留原有检测器的基础上增强鲁棒性
- 算法不足
  - 攻击类型覆盖有限
    - 方法主要针对释义改写攻击，对字符扰动、局部词替换等攻击仍需进一步验证
  - 推理成本较高
    - 多候选生成会显著增加推理时间和计算资源消耗

Iron Sharpens Iron: Defending Against Attacks in Machine-Generated Text Detection with Adversarial Training

GREATER TIPO

T	目标	提升机生文本检测器在扰动攻击、对抗攻击等攻击下的鲁棒性，区分人类撰写文本与机器生成文本
I	输入	待检测文本 数据集：SemEval Task8（8000，1000，1000）
P	处理	1. 利用开源代理模型提取 token 表示，并训练评分网络识别关键 token 2. 对关键 token 的 embedding 加入梯度扰动，生成候选替换词 3. 按 token 重要性进行贪心搜索并进行贪心剪枝 4. 将原始样本与动态生成的对抗样本共同用于训练检测器
O	输出	待测文本是否属于机器生成文本

P	问题	1.现有检测器面对释义改写、查询式对抗攻击等攻击时性能明显下降 2.传统检测方法依赖预先生成的攻击样本，对未知攻击泛化能力不足
C	条件	训练时需要目标检测器查询反馈，模拟真实攻击场景
D	难点	1.如何在黑盒条件下生成高成功率、低扰动、语义保持的强对抗样本 2.如何让检测器从动态变化的攻击样本中学习到跨攻击泛化的鲁棒特征
L	水平	2025 CCF A类

GREATER 创新点分析 攻防协同对抗

- 现有方法存在问题
  - 传统防御方法多采用静态数据增强，利用预先生成的一批攻击样本训练检测器，面对未见过的新攻击时泛化能力不足
  - RADAR 等方法引入攻击者与检测器的对抗思想，但更多面向特定攻击形式，难以覆盖多种扰动与查询式对抗攻击
- 解决方法
  - 将机生文本检测防御建模为攻击器与检测器的协同对抗过程，不再依赖固定攻击样本，而是在训练过程中动态生成攻击样本，形成一种动态对抗课程学习
    - GREATER-A： 负责生成能够误导检测器的强对抗样本
    - GREATER-D： 负责检测，需要在原始文本和攻击文本上都保持正确检测
  - 从针对单一攻击转向面向多攻击泛化鲁棒防御
    - 释义改写攻击、字符扰动、对抗样本攻击等
    - 查询式对抗攻击：根据检测器反馈不断优化文本，更接近真实攻击场景

GREATER 创新点分析 攻击器设计

- 现有方法存在问题
  - 现有文本扰动方法通常不访问目标检测器信息，只是简单调整 token 分布，容易生成低质量、非定向的攻击样本
- 解决方法
  - 设计黑盒攻击器 GREATER-A，在只能获得目标检测器输出的情况下，生成高效、隐蔽、强攻击性的对抗样本
    - 关键 token 识别：利用开源代理模型提取 token 表示，并训练评分网络预测重要性
    - Embedding 级扰动：不直接随机替换词，而是在重要 token 的 embedding 空间中加入扰动，通过梯度上升增大扰动前后输出分布的 KL 散度，使候选词更容易改变检测器判断；最后把扰动后的向量通过语言模型头映射回词表，得到候选替换词
    - 候选词约束：加入词性约束，减少语法错误，提升攻击文本的自然性和隐蔽性
    - 贪心搜索：按 token 重要性从高到低逐个尝试替换
    - 贪心剪枝：成功攻击后，逐个尝试恢复已修改 token

实验设计　数据资源

- 数据集
  - 训练数据：SemEval Task8（8000，1000，1000）
  - 文本扰动测试、生成式扰动测试、对抗攻击测试
- 对比方法

  EP、PP、CERT-ED、RanMask、Text-RS、Text-CRS、VAT、TAVAT、RADAR、OUTFOX、 PWWS、TextFooler等
- 评价指标
  - 平均查询次数Avg Queries 、攻击成功率ASR 、扰动率Pert.、困惑度变化ΔPPL 、语义相似度USE 、可读性变化Δr

Scenario	Dataset	Number of MGTs
Mixed Edit	Semeval Task8 (Siino, 2024)	1000
HMGC	Semeval Task8 (Siino, 2024)	1000
Paraphrasing	Semeval Task8 (Siino, 2024)	1000
Code-switching	Semeval Task8 (Siino, 2024)	1000
Human Obfuscation	Semeval Task8 (Siino, 2024)	1000
Emoji-cogen	Wang et al. (Wang et al., 2024a)	500
Typo-cogen	Wang et al. (Wang et al., 2024a)	500
ICL	Wang et al. (Wang et al., 2024a)	500
Prompt Paraphrasing	Wang et al. (Wang et al., 2024a)	500
CSGen	Wang et al. (Wang et al., 2024a)	500
Adversarial Attack	Semeval Task8 (Siino, 2024)	500

实验设计 GREATER-D 防御性能对比实验

Category	Method	Metric	Baseline	Adversarial Data Augmentation						Adversarial Training				
			F.t.XLM-RoBERTa-Base	EP	PP	CERT-ED	RanMask	Text-RS	Text-CRS	VAT	TAVAT	RADAR	OUTFOX	GREATER-D
Text Perturbation	*Mixed Edit*	*ASR(%)*↓	34.65	4.58	32.33	16.74	17.00	22.81	15.34	33.19	37.50	18.76	11.33	8.85
	HMGC	*ASR(%)*↓	30.00	6.53	27.34	12.52	11.74	16.58	13.37	25.19	31.50	8.53	3.74	2.28
	Paraphrasing	*ASR(%)*↓	70.58	54.65	6.27	32.10	40.20	59.58	26.67	67.98	65.90	2.08	14.96	3.45
	*Code-switching MF**	*ASR(%)*↓	50.58	38.37	34.08	29.19	27.78	48.80	27.15	36.71	47.52	32.57	5.46	1.13
	*Code-switching MR**	*ASR(%)*↓	47.91	31.23	30.21	5.30	10.80	16.98	14.36	38.03	46.46	14.14	5.37	1.02
	Human Obfuscation	*ASR(%)*↓	18.42	22.19	27.00	13.64	18.86	18.05	15.24	25.35	24.17	16.26	5.74	0.86
	Emoji-cogen	*ASR(%)*↓	32.19	46.55	44.17	11.41	22.23	17.72	27.10	52.35	40.33	33.13	7.72	0.47
	Typo-cogen	*ASR(%)*↓	60.10	61.57	59.72	27.29	38.82	37.09	44.79	70.26	63.04	41.52	9.29	1.08
	ICL	*ASR(%)*↓	1.40	1.41	1.30	1.72	0.83	1.89	0.67	1.13	1.88	2.77	0.59	0.20
	Prompt Paraphrasing	*ASR(%)*↓	0.00	0.69	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.94	0.65	0.23
	CSGen	*ASR(%)*↓	25.44	22.81	6.14	10.65	1.70	2.41	23.95	26.88	27.52	17.07	3.40	0.00
	Avg.	*ASR(%)*↓	33.75	26.42	24.41	14.66	17.27	21.99	18.97	34.28	35.07	17.07	6.20	1.78
Adversarial Attack	*PWWS*	*ASR(%)*↓	61.45	48.47	49.06	11.07	14.83	16.13	19.15	53.56	62.68	12.58	-	5.91
		Queries↑	1197.95	1237.62	1235.01	1371.28	1361.15	1356.67	1342.15	1226.53	1180.87	1366.85	-	1390.69
	TextFooler	*ASR(%)*↓	72.29	70.76	70.02	11.07	17.43	19.56	22.98	69.25	94.02	14.46	-	6.11
		Queries↑	690.92	724.23	713.77	1362.17	1291.35	1271.34	1237.18	780.51	440.28	1301.84	-	1396.52
	BERTAttack	*ASR(%)*↓	71.49	69.34	63.52	6.04	12.42	12.30	16.94	69.45	93.81	13.08	-	5.70
		Queries↑	411.54	446.64	425.68	710.56	682.29	680.14	667.82	450.57	279.02	677.24	-	718.84
	A2T	*ASR(%)*↓	45.47	37.45	50.10	6.24	11.22	14.52	14.31	29.12	54.64	8.58	-	5.09
		Queries↑	293.26	320.88	302.18	516.07	499.89	493.25	488.91	361.88	207.15	505.54	-	519.77
	T-PGD	*ASR(%)*↓	49.90	34.15	25.28	8.52	12.07	13.15	14.92	42.14	45.58	7.79	-	5.61
		Queries↑	354.75	515.62	712.08	799.62	766.53	752.04	744.17	404.85	377.69	813.76	-	824.58
	GREATER-A(ours)	*ASR(%)*↓	96.58	87.08	84.34	62.17	63.58	75.25	82.29	89.26	85.02	71.04	-	46.08
		Queries↑	62.63	66.71	68.53	99.56	98.92	106.08	75.98	63.57	67.17	100.74	-	190.64
	Avg.	*ASR(%)*↓	66.20	57.87	57.05	17.52	21.92	25.15	28.43	58.80	72.62	21.26	-	12.42
		Queries↑	501.84	551.95	576.21	809.88	783.36	776.59	759.37	547.98	425.36	794.33	-	840.17
Total	*Avg.*	*ASR(%)*↓	45.20	37.52	35.93	15.67	18.91	23.11	22.31	42.93	48.33	18.55	6.20	5.53

实验设计　GREATER-A 攻击性能对比实验

- 评估GREATER-A在可查询、零查询场景下的攻击表现
  - 在平均查询次数、攻击成功率、扰动率、困惑度变化、语义相似度、可读性变化方面均表现优异

Attack Type	Method	Avg Queries ↓	ASR (%) ↑	Pert. (%) ↓	ΔPPL ↓	USE ↑	Δr ↓
Query-based	PWWS	1197.95	61.45	4.71	37.85	0.9488	12.76
	TextFooler	690.92	72.29	6.26	46.89	0.9302	21.07
	BERTAttack	411.54	71.49	5.79	36.15	0.9402	15.78
	HQA	283.89	88.13	23.57	102.87	0.8854	72.16
	FastTextDodger	745.75	63.78	13.29	76.15	0.9188	55.14
	ABP	785.18	75.65	14.63	39.61	0.8709	26.40
	T-PGD	354.75	49.90	38.01	181.31	0.8197	35.09
	GREATER-A	62.63	96.58	7.26	35.22	0.9506	9.21
	A2T (White Box)	293.26	45.47	7.01	62.84	0.9215	32.07
Zero-query	WordNet	-	42.60	-	26.27	0.90	4.26
	Back Translation	-	2.40	-	6.40	0.91	3.06
	Rewrite	-	36.47	-	92.08	0.79	15.62
	T-PGD	-	62.40	-	140.13	0.82	44.99
	HMGC	-	30.00	-	12.94	0.84	36.90
	GREATER-A	-	69.11	-	43.11	0.92	4.55

- 评估GREATER-A的不同模块发挥的作用
  - R+NP：随机选 token，不做剪枝　　R+P：随机选 token，但加入剪枝
  - S+NP：使用重要 token 识别，但不剪枝
  - Mask-T：识别重要 token 后，用 mask 替代扰动
  - GREATER-W：词级扰动版本　　GREATER-WordNet：用 WordNet 同义词替换

Method	Avg Queries ↓	ASR (%) ↑	Pert. (%) ↓	ΔPPL ↓	USE ↑	Δr ↓
R+NP	26.61	75.77	11.56	106.29	0.9324	14.41
R+P	53.22	75.77	9.51	58.72	0.9497	12.13
S+NP	31.32	96.58	11.28	85.18	0.9136	21.80
Mask-T	303.82	96.38	8.33	27.12	0.9696	4.73
GREATER-W	33.21	96.38	11.49	65.16	0.9482	10.04
GREATER-WordNet	28.41	80.89	16.68	53.82	0.9199	13.21
GREATER-A	62.63	96.58	7.26	35.22	0.9506	9.21

算法总结 GREATER

- 算法贡献
  - 攻防协同对抗训练框架：将机生文本检测防御建模为攻击器与检测器的同步对抗过程，在训练过程中动态生成攻击样本，替代传统静态数据增强
    - 从针对单一攻击的防御转向面向释义改写、字符扰动、查询式对抗攻击等攻击的综合鲁棒防御
  - 黑盒攻击器设计：利用代理模型识别关键 token，在 embedding 空间生成候选扰动，结合词性约束、贪心搜索和贪心剪枝，生成隐蔽、强攻击性的对抗样本
- 算法不足
  - 攻击器与检测器同步训练带来较高训练成本，且攻击样本生成过程需要多次查询目标检测器
  - 当代理模型与实际检测器差异较大时，攻击样本质量可能受到影响

特点总结与未来展望

特点总结与未来展望

- 特点总结
  - Paraphrase Inversion
    - 将释义改写攻击防御转化为“改写分析—释义逆变换—机生文本检测”的前置处理流程，可作为可插拔模块使用
    - 利用改写模型的系统性偏差，恢复原始文本中的风格特征、机器生成痕迹等
  - GREATER
    - 构建攻击器与检测器同步更新的攻防协同对抗训练框架
    - 通过关键 token 识别、embedding 级扰动、词性约束、贪心搜索和贪心剪枝生成强对抗样本， 提升对多种扰动与未知攻击的泛化能力
- 未来发展
  - 提升跨模型、跨领域泛化能力，降低对特定生成器和训练语料的依赖
  - 根据扰动类型和攻击强度自适应调整防御策略

参考文献

[1] Rivera Soto R, Chen B, Andrews N. Mitigating Paraphrase Attacks on Machine-Text Detectors via Paraphrase Inversion. Findings of the Association for Computational Linguistics: ACL 2025[C]. Vienna, Austria: Association for Computational Linguistics, 2025: 4421-4433.

[2] Li Y, Zhang Z, Li C, et al. Iron Sharpens Iron: Defending Against Attacks in Machine-Generated Text Detection with Adversarial Training. Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)[C]. Vienna, Austria: Association for Computational Linguistics, 2025: 3091-3113.

道德经

知人者智，自知者明。胜人者有力，自胜者强。知足者富。强行者有志。不失其所者久。死而不亡者，寿。

谢谢！