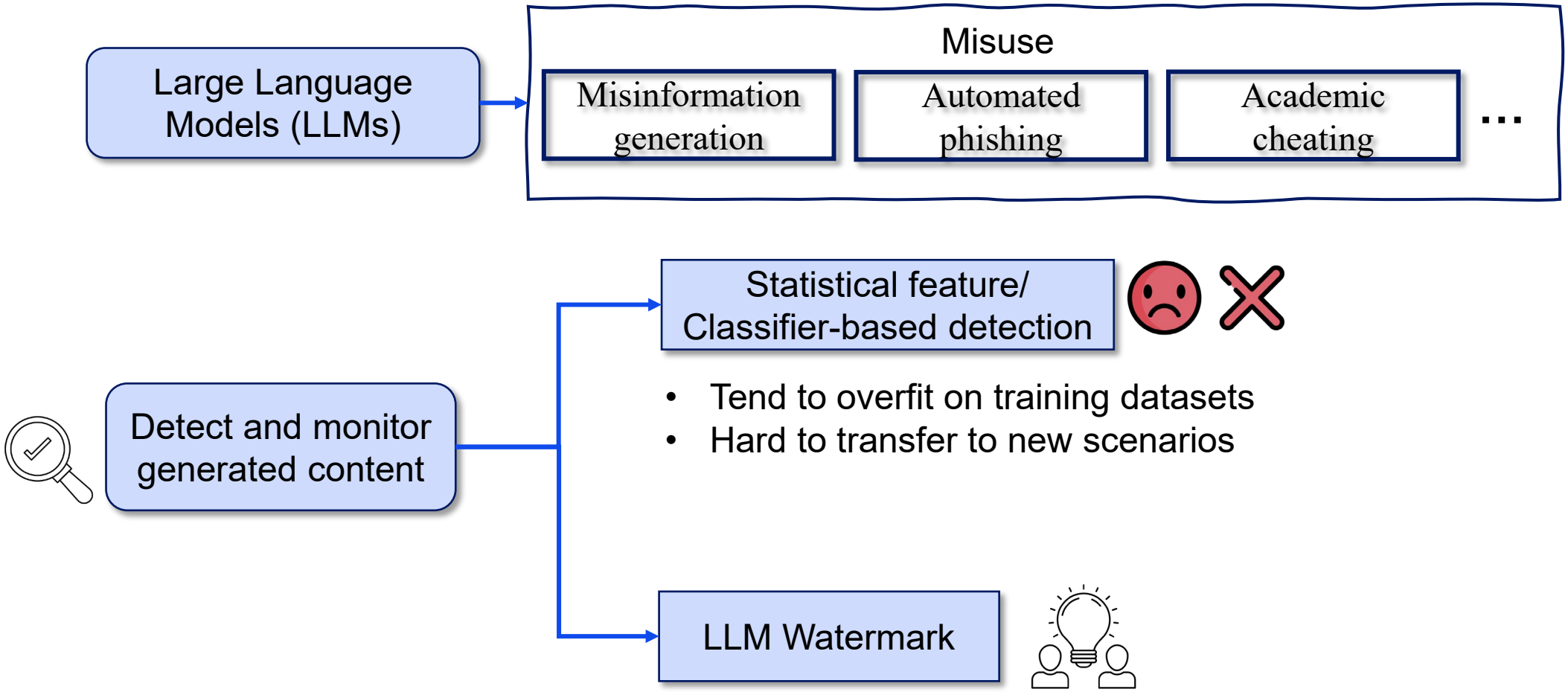
Character-Level Perturbations Disrupt LLM Watermarks

--- Network and Distributed System Security (NDSS) Symposium 2026

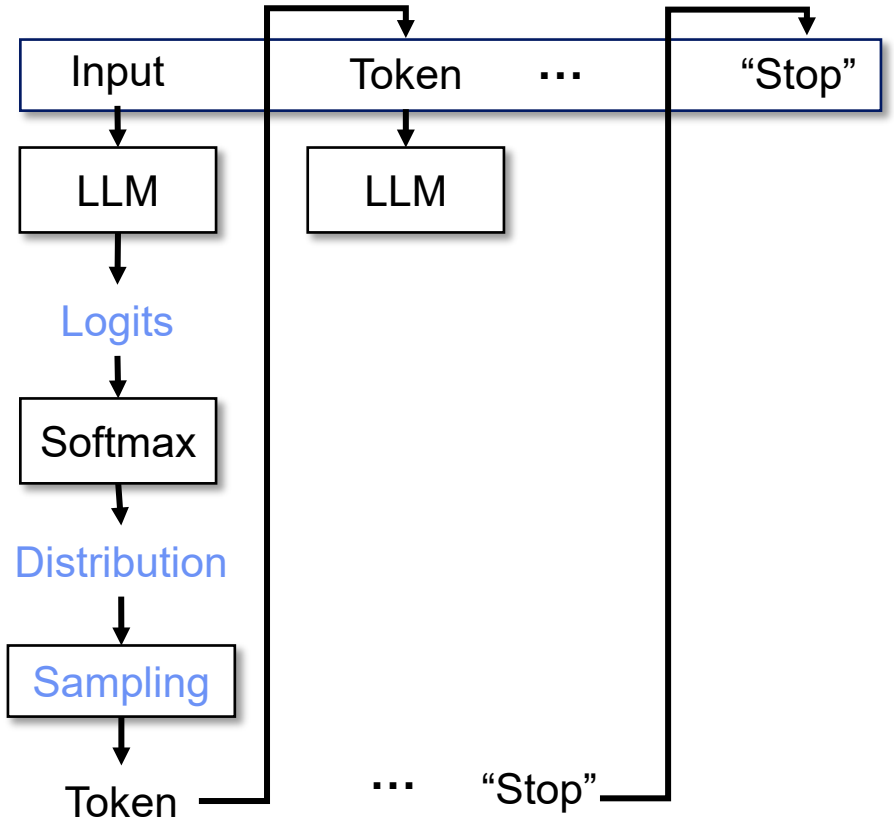
Zhaoxi Zhang, Xiaomei Zhang, Yanjun Zhang, He Zhang, Shirui Pan, Bo Liu, Asif Gill, Leo Yu Zhang
University of Technology Sydney, Griffith University, Royal Melbourne Institute of Technology

1. Introduction



2. Background

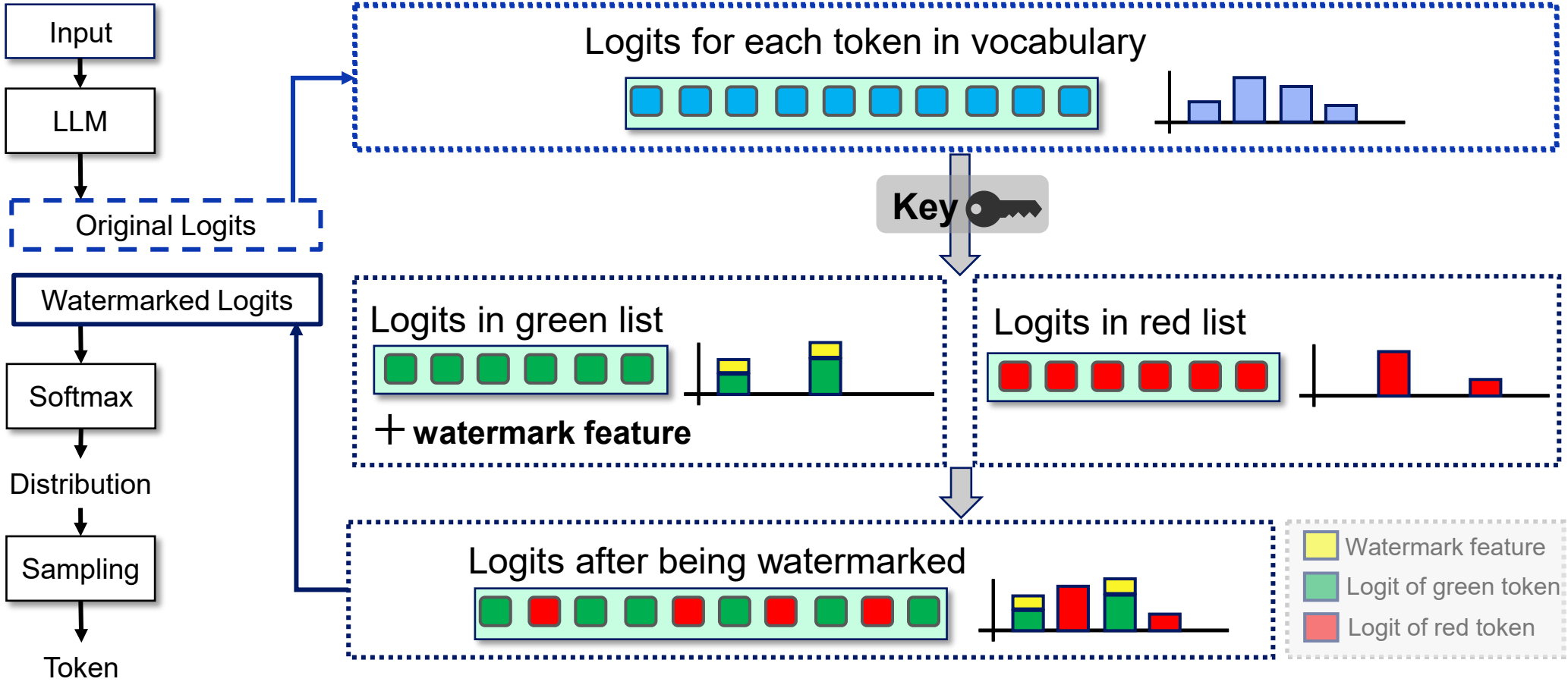
LLM Generation Process



Taxonomy of LLM Watermark

- **Watermark Injection**
  - **Inference-time**
    - **Logits Generation:** KGW [1], Unigram [2]
    - **Distribution Generation:** DIP[3]
    - **Sampling:** SynthID [4], Unbias [5]
  - **Training-time**

2. Background
--- Injecting LLM Watermark



2. Background
--- Detecting LLM Watermark

Watermarked Text

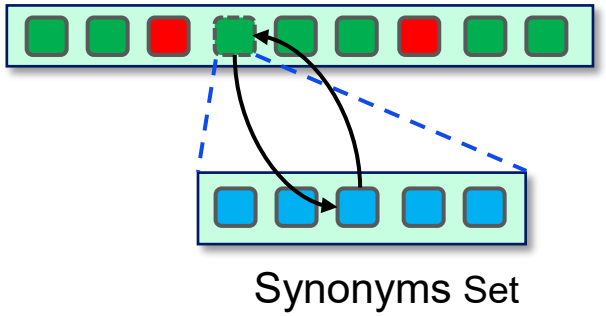
Green: 7

Non-watermarked Text

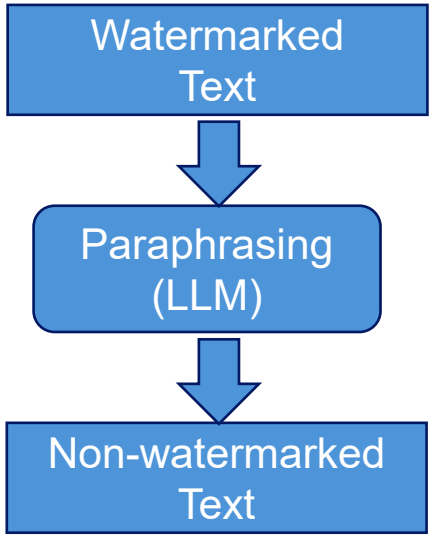
Green: 3

- **After watermarking, the number of green tokens in the watermarked text is greater than in the non-watermarked text.**
- **We can detect watermark by count the number of green tokens.**

2. Background
--- Token & Sentence Level Attack

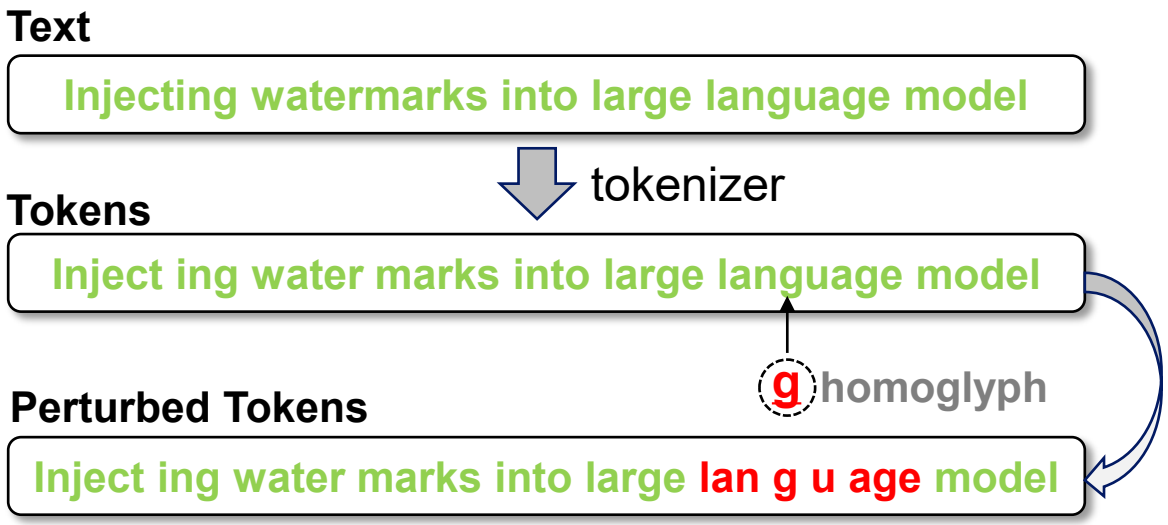


Token editing attack [1, 2]:
Randomly choose tokens and substitute with synonyms



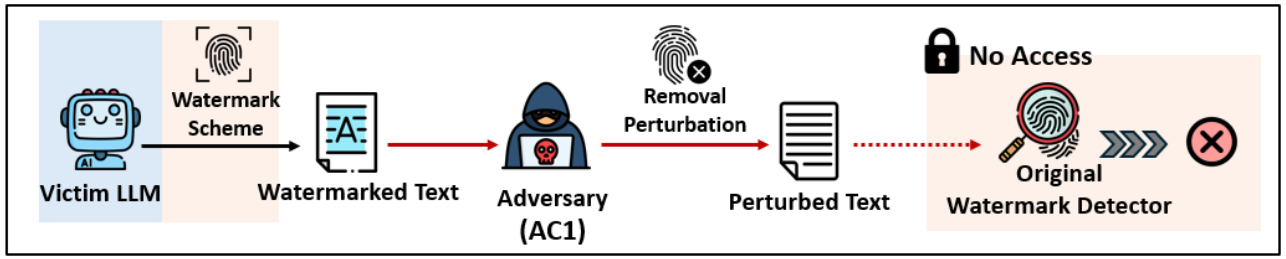
Paraphrasing attack [6, 7]:
Paraphrase can change most of tokens, even the order of sentence

2. Background
--- Character-Level Attack

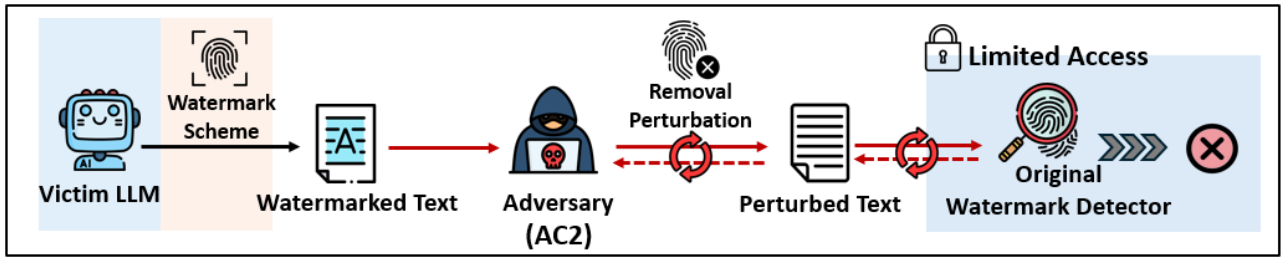


Character Editing Attack:
Randomly adding typos, misspelling, homoglyphs into text.

3. Problem Formulation
--- System Model & Threat Model

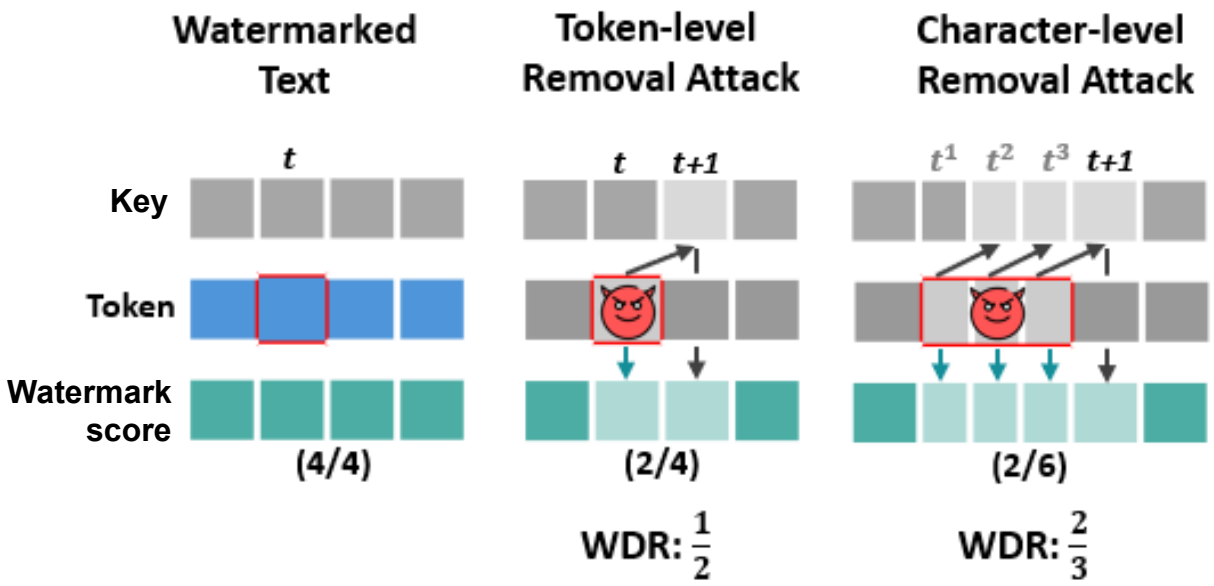


System Model 1
- Private watermark detector.
- **AC1:** Black-box query to victim LLM.



System Model 2
- Public watermark detector
- **AC2:** Black-box query to victim LLM and watermark detector.

4. Watermark Removal
--- Character-Level Attacks Are More Effective

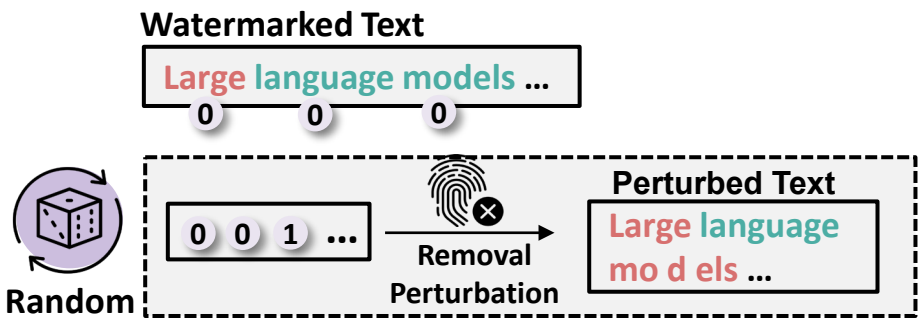


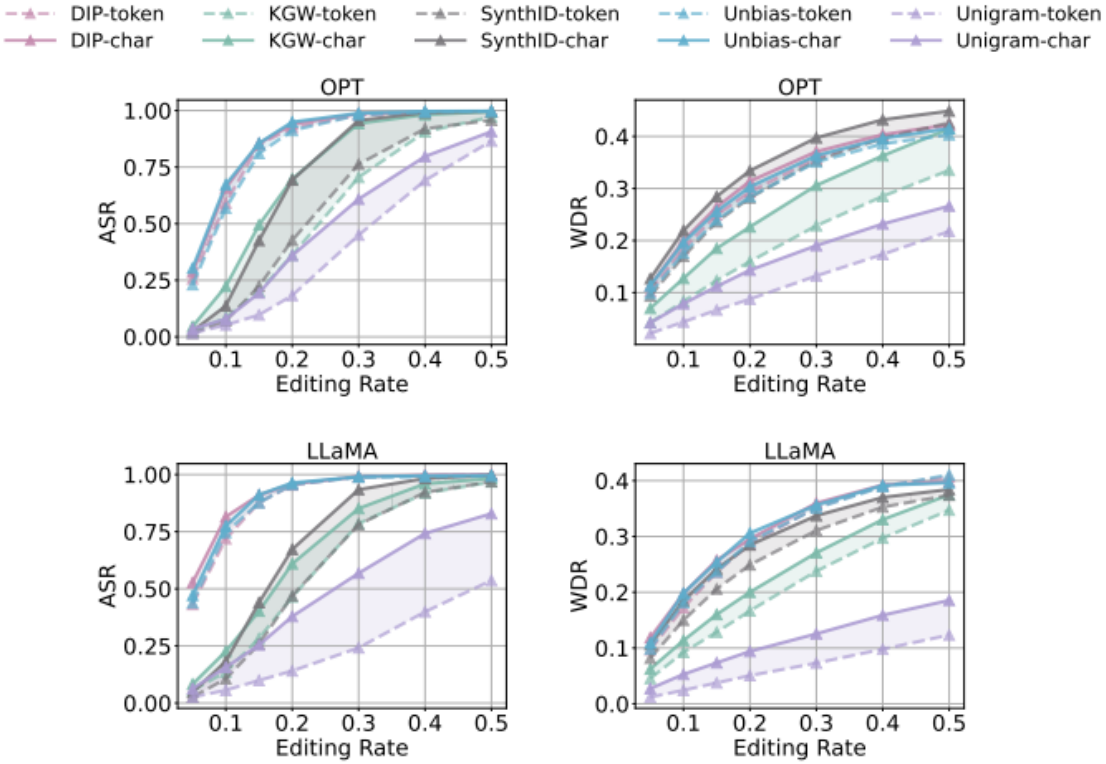
By disrupting the *tokenization process*, character-level perturbations achieve a *broader attack range* under the same editing budget.

- $\tilde{X}$: watermarked text, X: non-watermark text
- Watermark score: $S_w(X)$
- Watermark score dropping rate (WDR): $\frac{S_w(X)-S_w(\tilde{X})}{S_w(X)}$
- **Attack range: how many tokens are affected by a single edit**

4. Watermark Removal

--- Character-Level Attacks Are More Effective





Character-level attacks consistently outperform token-level attacks in both ASR and watermark score dropping rate (WDR), across all watermark schemes and editing rates.

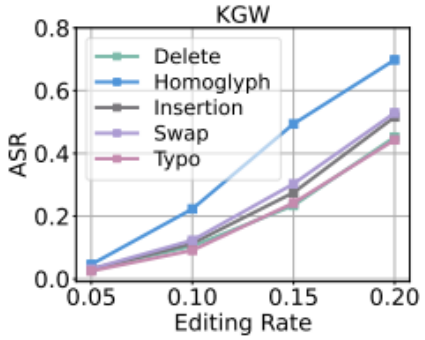
4. Watermark Removal

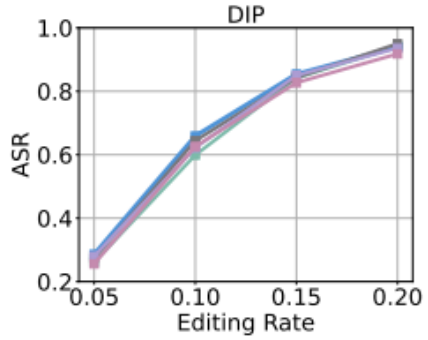
--- Comparison Among Different Character Perturbations

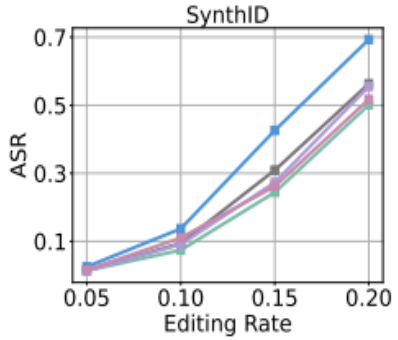
Character-level Perturbation:

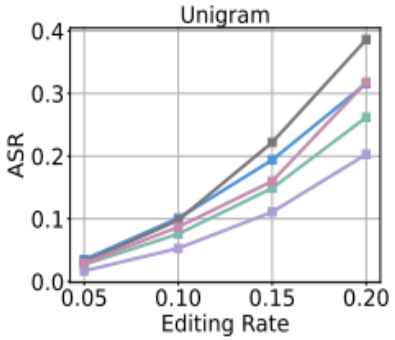
1. **Typo**
2. **Deletion**
3. **Swap**
4. **Zero-width character insertion**
5. **Homoglyph substitution**

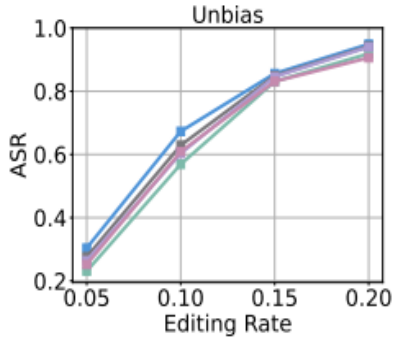
- All perturbations show improved ASR with increasing editing rates.
- *Homoglyph* substitution consistently achieves higher ASR than other methods, especially at lower editing rates





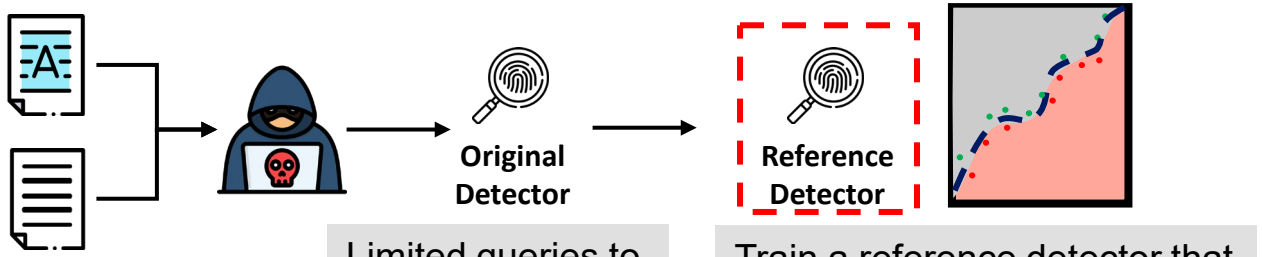






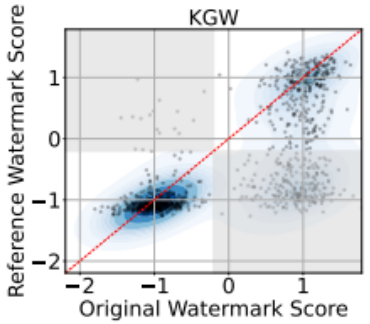
5. Guided Character-Level Attack
--- Reference Detector

Collect watermarked / unwatermarked texts.

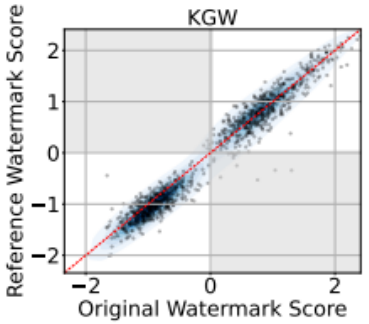


Limited queries to original detector.

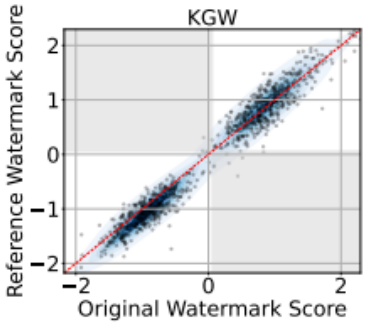
Train a reference detector that mimics the original detector.



Ref-0
no augmentation



Ref-5
5 variants/ sample



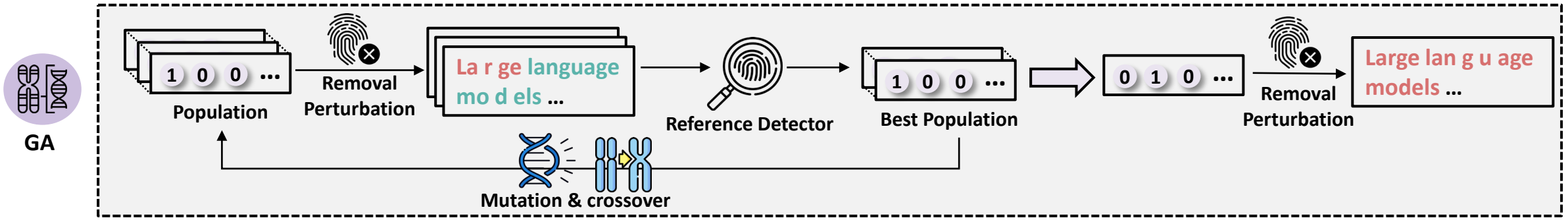
Ref-9
9 variants/ sample

Light data augmentation to improve the reliability and generalization of the reference detector.

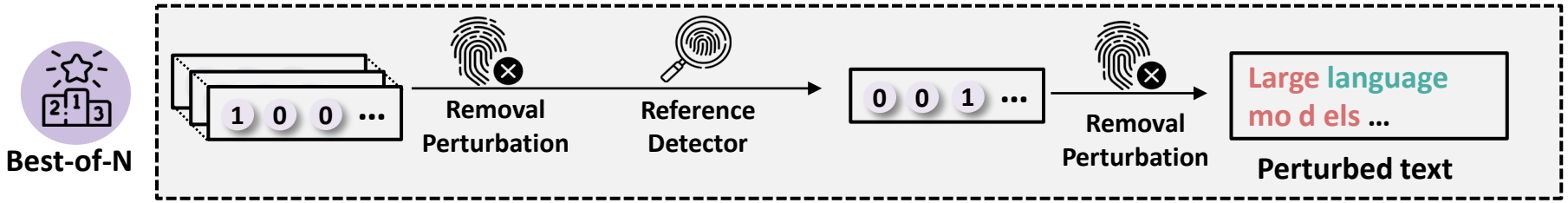
5. Guided Character-Level Attack

--- Best-of-N & Genetic Algorithm (GA)

Gradient-free **GA** identifies removal-relevant tokens.



Best-of-*N* : a simplified version of **GA.**



5. Guided Character-Level Attack

--- Best-of-N & Genetic Algorithm

		Best-of-N (10) Token		Best-of-N (10) Char		Best-of-N (1500) Token		Best-of-N (1500) Char		GA Token		GA Char		Sand Token		Sand Char	
		WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)	WDR(↑)	ASR(↑)
OPT	KGW	0.0985	0.0842	0.1591	0.2626	0.0956	0.1164	0.1490	0.3082	0.1128	0.1233	0.2110	0.4966	0.0305	0.0103	0.0763	0.0685
	DIP	0.1833	0.5791	0.2033	0.6871	0.1797	0.6151	0.2057	0.6727	0.1864	0.6187	0.2433	0.8453	0.0686	0.0935	0.1624	0.5000
	SynthID	0.1896	0.0606	0.2536	0.1650	0.1886	0.0774	0.2611	0.2121	0.2262	0.1145	0.3093	0.4209	0.1091	0.0168	0.2160	0.0976
	Unigram	0.0639	0.0741	0.1271	0.2907	0.0916	0.1370	0.1508	0.4110	0.1629	0.4829	0.1950	0.6575	0.0382	0.0274	0.0915	0.1164
	Unbias	0.1787	0.5532	0.2067	0.6667	0.1781	0.5496	0.2006	0.6738	0.1864	0.6187	0.2530	0.8369	0.0609	0.1489	0.1717	0.5284
LLaMA	KGW	0.0980	0.1667	0.1216	0.2558	0.0950	0.1750	0.1295	0.2643	0.1176	0.2321	0.1693	0.4214	0.0384	0.0357	0.0849	0.1179
	DIP	0.1827	0.6833	0.2056	0.8175	0.1869	0.7042	0.2116	0.8000	0.1805	0.6917	0.2580	0.9042	0.0889	0.3458	0.1720	0.6708
	SynthID	0.1604	0.1010	0.1915	0.1996	0.1586	0.1010	0.2054	0.2054	0.1779	0.1448	0.2447	0.3603	0.0369	0.0034	0.1601	0.1010
	Unigram	0.0296	0.0651	0.0596	0.1724	0.0292	0.0575	0.0557	0.1686	0.0694	0.1686	0.1888	0.7356	0.0127	0.0345	0.0148	0.0307
	Unbias	0.1843	0.7167	0.2074	0.7875	0.1804	0.6750	0.2055	0.7958	0.1841	0.7208	0.2379	0.8667	0.0749	0.2833	0.1614	0.6292

- Attack success rate (ASR)
- Watermark score dropping rate (WDR): $\frac{S_w(X)-S_w(\tilde{X})}{S_w(X)}$
- Watermark score: $S_w(X)$
- $\tilde{X}$: watermarked text, X: non-watermarked text

- The results show that GA consistently outperforms others
- The character-level attacks consistently outperform token-level attacks

6. Potential Defenses and Adaptive Attacks

Potentially Competitive Defenses:

- ➢ Spell-checking and correction (SC)
- ➢ Optical character recognition (OCR)
- ➢ Unicode normalization (UN)
- ➢ Deletion (DE) of anomalous characters

Compound character-level perturbation:

- Swapping + Homoglyph
- Typo + Homoglyph
- Zero-width Insertion + Homoglyph
- …

	Detector type	KGW	DIP	SynthID	Unigram	Unbias
GA	D_{ori}	0.4214	0.9042	0.3603	0.7548	0.8667
Adaptive GA (SC)	D_{ori}	0.5429	0.9125	0.3644	0.9502	0.9125
	$D_{\text{ori}} \oplus F_{\text{SC}}$	0.4250	0.8917	0.4049	0.8697	0.8917
Adaptive GA (OCR)	D_{ori}	0.5214	0.8333	0.4122	0.5896	0.8333
	$D_{\text{ori}} \oplus F_{\text{OCR}}$	0.5036	0.9500	0.5405	0.4776	0.9333
Adaptive GA (DE)	D_{ori}	0.4507	0.8583	0.4595	0.9776	0.9000
	$D_{\text{ori}} \oplus F_{\text{DE}}$	0.3028	0.9083	0.3311	0.7612	0.8583
Adaptive GA (UN)	D_{ori}	0.4718	0.9333	0.4459	0.9776	0.8750
	$D_{\text{ori}} \oplus F_{\text{UN}}$	0.4507	0.9167	0.4324	0.7016	0.8667

- The average ASR slightly decreases but remains higher than the baseline GA

Reference

[1] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, “A watermark for large language models,” in *ICML*, pp. 17061–17084, PMLR, 2023.

[2] X. Zhao, P. V. Ananth, L. Li, and Y.-X. Wang, “Provable robust watermarking for AI-generated text,” in International Conference on Learning Representations, 2024.

[3] Y. Wu, Z. Hu, J. Guo, H. Zhang, and H. Huang, “A resilient and accessible distribution-preserving watermark for large language models,” in *ICML*, 2024.

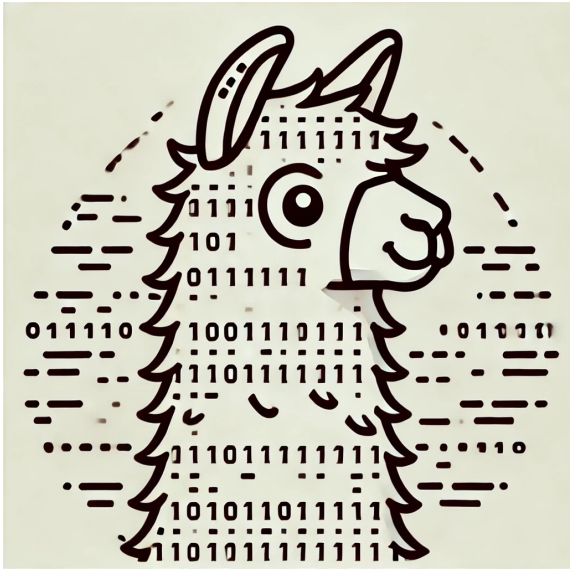
[4] S. Dathathri, A. See, S. Ghaisas, P.-S. Huang, R. McAdam, and et al., “Scalable watermarking for identifying large language model outputs,” *Nature*, vol. 634, no. 8035, pp. 818–823, 2024.

[5] Z. Hu, L. Chen, X. Wu, Y. Wu, H. Zhang, and H. Huang, “Unbiased watermark for large language models,” in *International Conference on Learning Representations*, 2024.

[6] K. Krishna, Y. Song, M. Karpinska, and et al., “Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense,” in *NeurIPS*, vol. 36, pp. 27469–27500, Curran Associates, Inc., 2023.

[7] Z. He, B. Zhou, H. Hao, A. Liu, and et al., “Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models,” in *Proceedings of the 62nd ACL (Volume 1: Long Papers)*, pp. 4115–4129, ACL, Aug. 2024.

Reference



Project website

GitHub

Awesome-LLM-Watermark

Thank You!