Stealing Watermarks of Large Language Models via Mixed Integer Programming

---Annual Computer Security Applications Conference (ACSAC) 2024

Zhaoxi Zhang, Xiaomei Zhang, Yanjun Zhang, Leo Yu Zhang, Chao Chen, Shengshan Hu, Asif Gill, Shirui Pan
University of Technology Sydney, Griffith University, Royal Melbourne Institute of Technology, Huazhong University of Science and Technology

1.Introduction

Large Language Models (LLMs)

Misuse

- Misinformation generation
- Automated phishing
- Academic cheating
- …

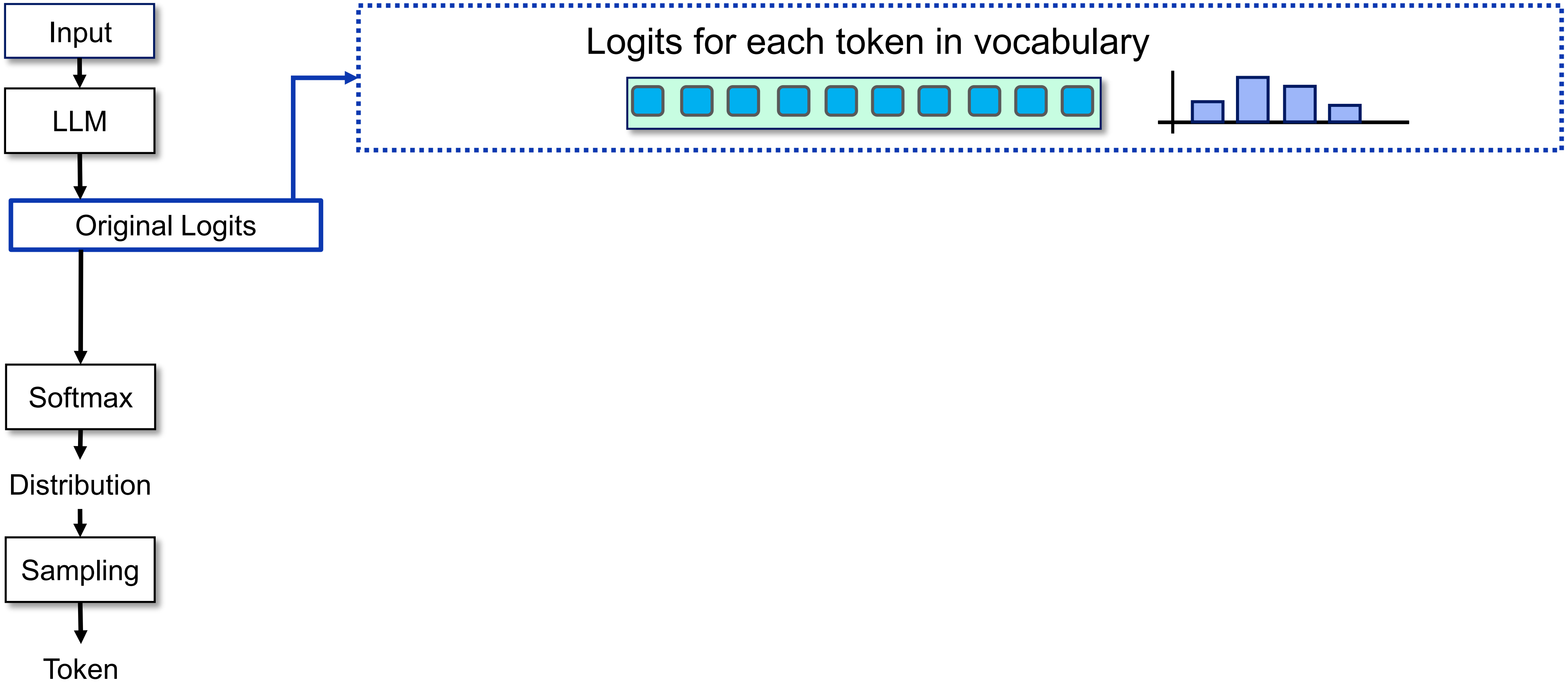
Detect and monitor generated content

Statistical feature/ Classifier-based detection

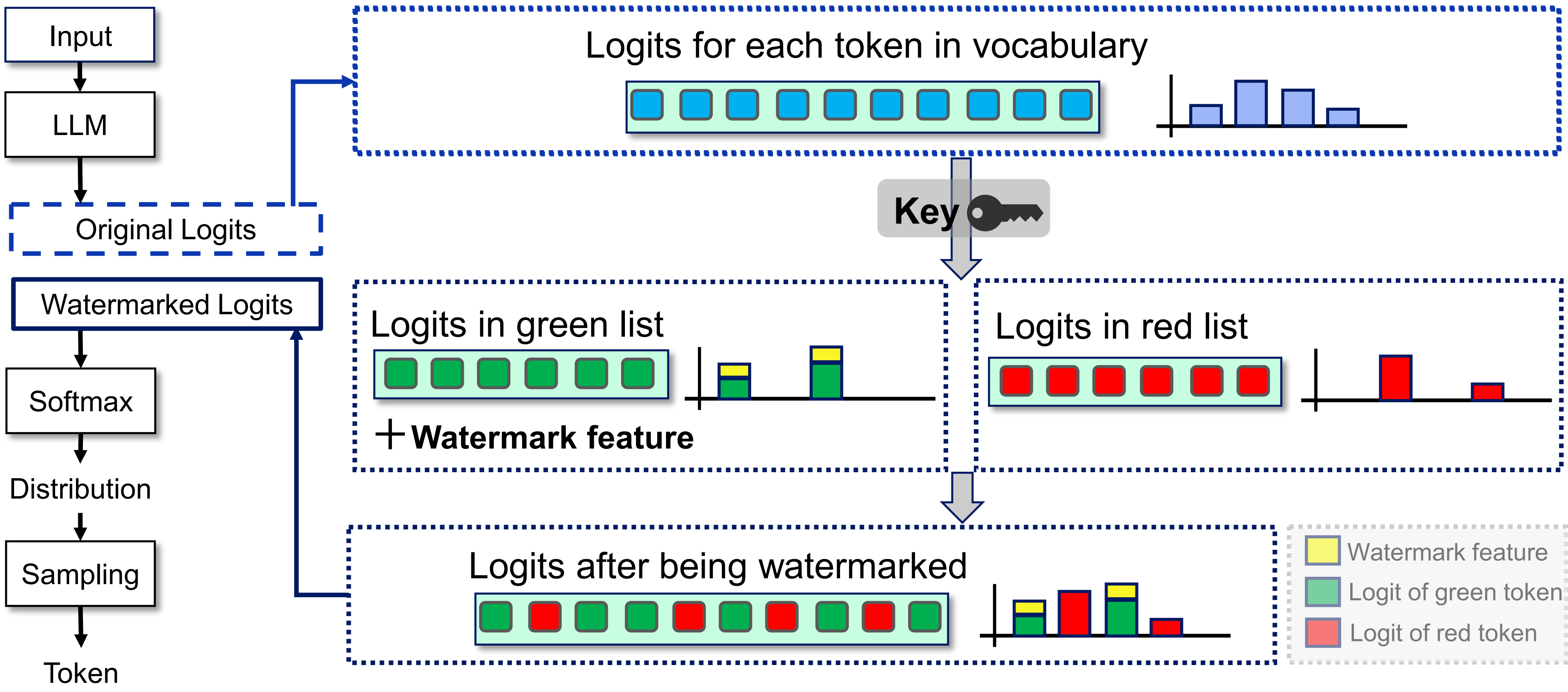
- Highly susceptible to malicious attacks
- Tend to overfit on training datasets
- Hard to transfer to new scenarios

LLM Watermark

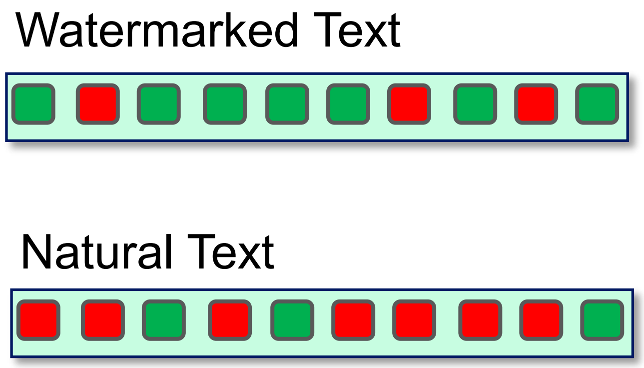
2. LLM Watermark
--- Injecting LLM Watermark



2. LLM Watermark
--- Injecting LLM Watermark

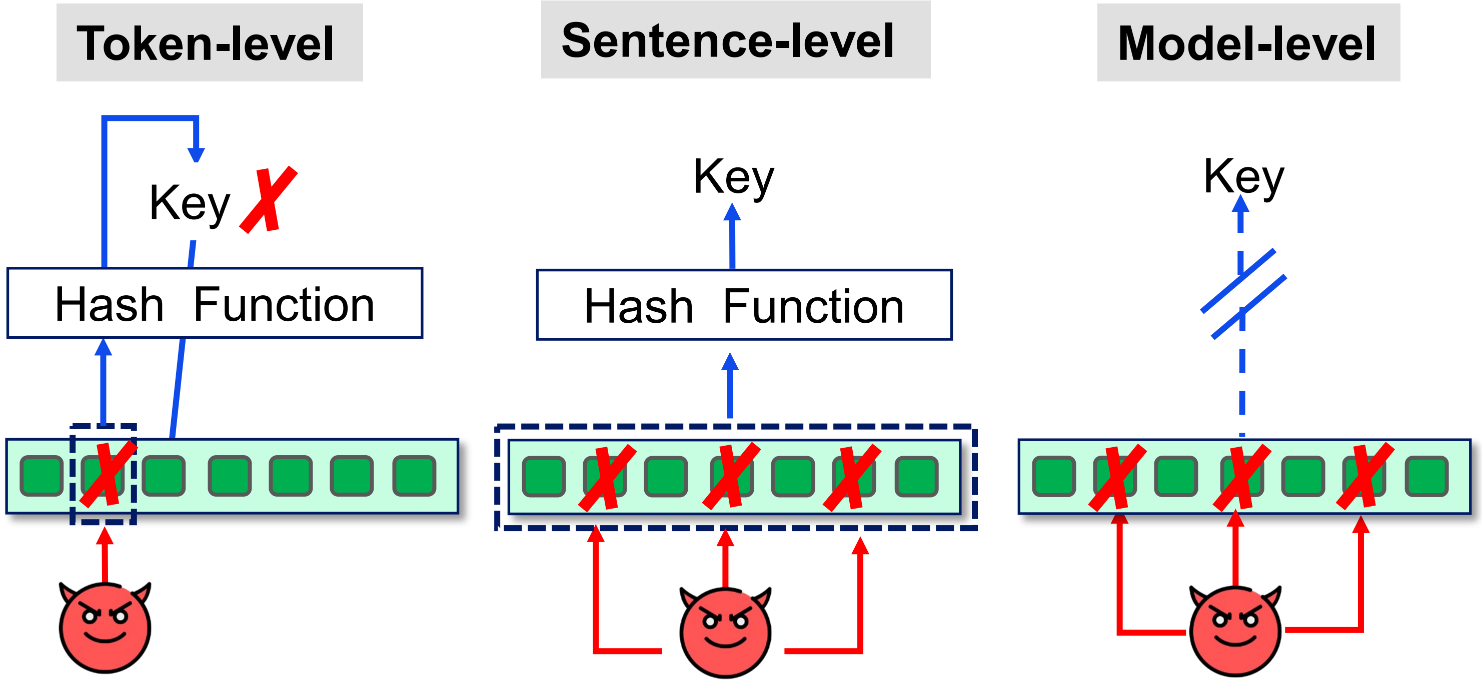


2. LLM Watermark
--- Detecting LLM Watermark



- After watermarking, the number of green tokens in the watermarked sentences is greater than in the non-watermarked text.
- LLM watermark can be detected by count the number of green tokens.

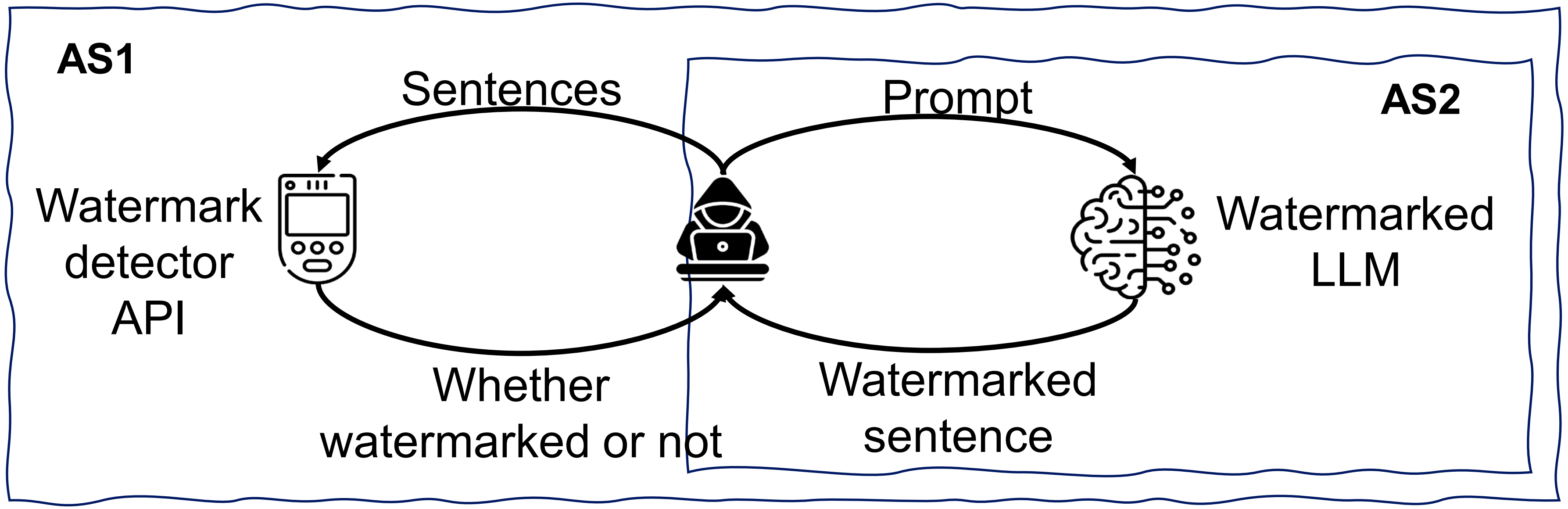
3. Problem Statement



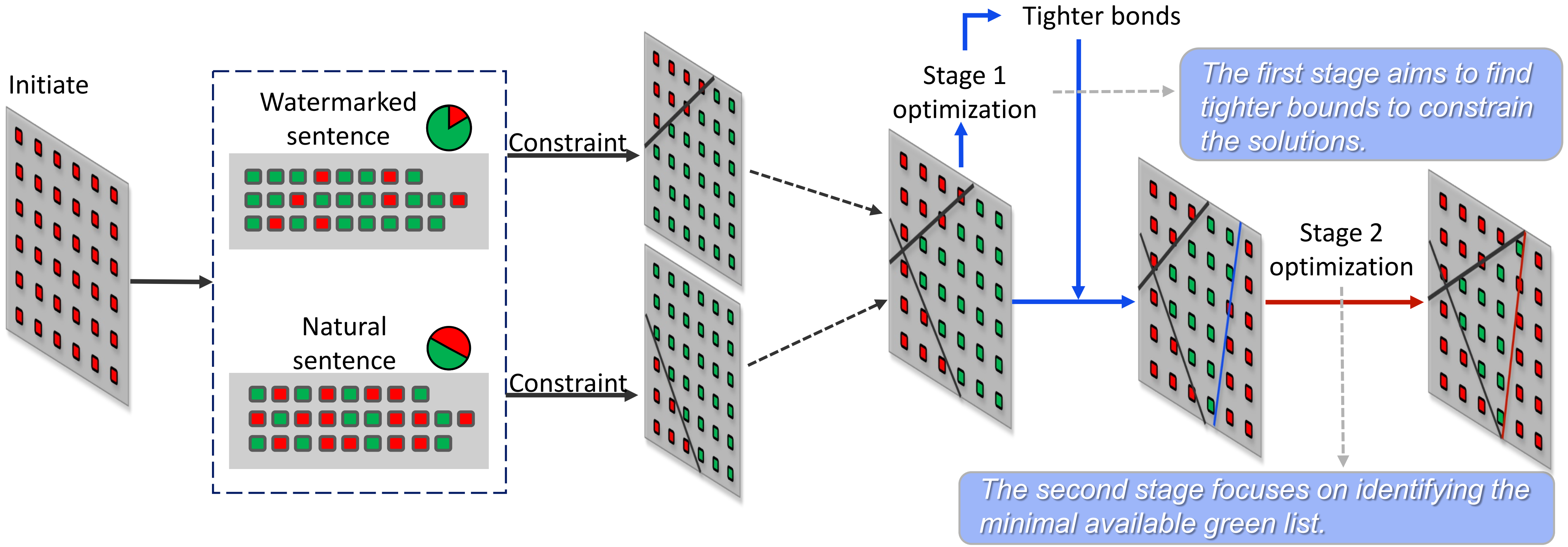
- Robustness: Token-level < Sentence-level < Model-level
- Sentence-level and model-level approaches provide insufficient robustness as both remain vulnerable to stealing attacks.
- A watermark stealing attack aims to infer the details of an LLM watermarking scheme.

4. Threat Model

- **Attack Setting 1**: attackers ***can*** generate text using the LLMs and verify whether the text is watermarked by calling the detector API.
- **Attack Setting 2**: attackers ***cannot*** access the watermark detector API.



5. Green List Stealing



The watermark stealing can be transformed into a **mixed-integer programming** problem:

- *The association between tokens and the green list can be represented as integers*
- ***Constraints**: watermark detection rules*
- ***Objective**: finding a minimal available green list for the watermark text*

5. Green List Stealing

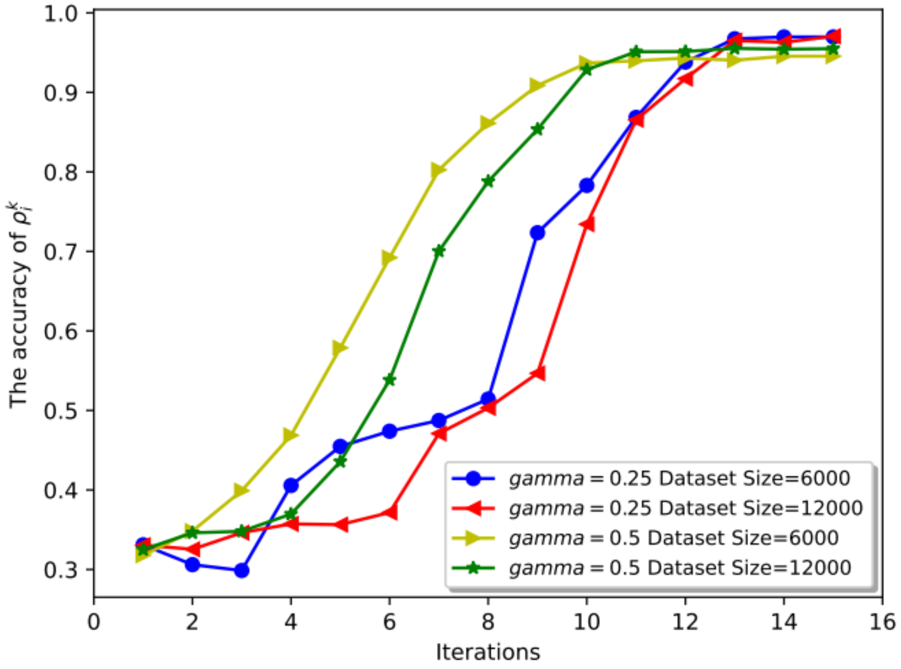
--- Multi-key Stealing

- The attacker need to find the max green number for each sentence:

Which key is suitable for this sentence

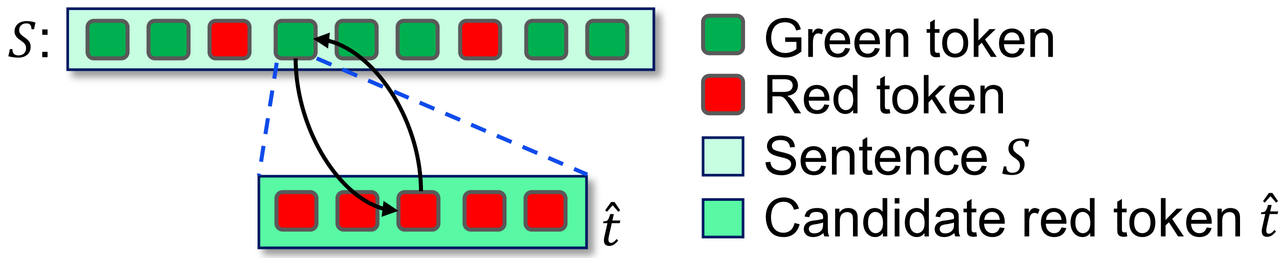
Random initialize $\{\hat{\rho}_i^k\}$
for until optimization converge **do**
    Optimize Stage-1 while fixing $\hat{\rho}_i^k$;
    Optimize Stage-2 while fixing $\hat{\rho}_i^k$;
    Resign $\hat{\rho}_i^k = \begin{cases} 1, \text{if } k = \text{argmax}_k(\text{G}(S_i, k)) \\ 0, \text{else} \end{cases}$
end for

The number of green token under key k



6. Watermark Removal

- Removing watermarks in sentences by replacing green tokens with red ones.



- Replace tokens with the most similar tokens in the stolen green list.

7. Experiment
--- Experimental Settings

- **LLM**: OPT-1.3B, LLaMA-2-7B.
- **Watermarked text**: Randomly sample text from the C4 dataset as prompts to query the LLM for generating watermarked text.
- **Solver** for the mixed integer programming: Gurobi.
- **Baseline**: Frequency-based, tokens are categorized as green if their frequency is higher in the watermark dataset than in the natural dataset.

7. Experiment

---Main Results (Green List Stealing)

- Attacker performance of green list stealing against LLaMA-2-7B under AS1 and AS2.

watermark seting	Dataset size	Ours (AS1) N_g	N_t	Precision(↑)	Freq. (AS1) N_g	N_t	Precision(↑)	Ours (AS2) N_g	N_t	Precision(↑)	Freq. (AS2) N_g	N_t	Precision(↑)
$\gamma = 0.25$ $\delta = 2$	4000	1064	885	83.18%	5154	2547	49.42%	3165	2003	63.29%	6032	2782	46.12%
	10000	1431	1224	85.53%	5519	2970	53.81%	2852	2069	72.55%	6613	3223	48.74%
	20000	1396	1256	89.97%	5494	3181	57.90%	2582	2056	79.63%	6727	3505	52.10%
	40000	2146	1912	89.10%	5425	3335	61.47%	2393	1990	83.16%	6680	3693	55.28%
$\gamma = 0.25$ $\delta = 4$	4000	732	678	92.62%	4350	2867	65.91%	3884	2813	72.43%	4392	2882	65.62%
	10000	780	731	93.72%	4704	3259	69.28%	4466	3347	74.94%	4736	3275	69.15%
	20000	867	803	92.62%	4895	3498	71.46%	4443	3481	78.35%	4937	3517	71.24%
	40000	933	861	92.28%	5020	3737	74.44%	4969	3923	78.95%	5062	3754	74.16%
$\gamma = 0.5$ $\delta = 2$	4000	2136	1884	88.20%	6417	4784	74.55%	6712	5149	76.71%	6881	5080	73.83%
	10000	2253	2035	90.32%	7233	5643	78.02%	6864	5569	81.13%	7938	6054	76.27%
	20000	2633	2394	90.92%	7661	6152	80.30%	7029	5872	83.54%	8510	6616	77.74%
	40000	3245	2976	91.71%	7811	6460	82.70%	7902	6677	84.50%	8828	7028	79.61%
$\gamma = 0.5$ $\delta = 4$	4000	2204	2047	92.88%	6240	5211	83.51%	6095	5256	86.23%	6284	5249	83.53%
	10000	3308	3078	93.05%	7351	6242	84.91%	6868	6056	88.18%	7386	6275	84.96%
	20000	3398	3174	93.41%	7855	6792	86.47%	6296	5749	91.31%	7918	6839	86.37%
	40000	3533	3336	94.42%	8173	7205	88.16%	8511	7668	90.10%	8253	7265	88.03%

Average higher 18.23%

Average higher 9.52%

- N_g: the number of tokens in the stolen green list
- N_t: the number of true green tokens in the stolen green list
- Precision= N_g / N_t

7. Experiment

---Main Results (Watermark Removal)

- Performance of watermark removal against LLaMA-2-7B under AS1 and AS2.

Watermark Setting	Dataset Size	AS1 G^b_{avg}	AS1 $G^a_{avg}(\downarrow)$ Ours	AS1 $G^a_{avg}(\downarrow)$ Freq.	AS1 GRR(↓) Ours	AS1 GRR(↓) Freq.	AS2 G^b_{avg}	AS2 $G^a_{avg}(\downarrow)$ Ours	AS2 $G^a_{avg}(\downarrow)$ Freq.	AS2 GRR(↓) Ours	AS2 GRR(↓) Freq.
$\gamma = 0.25$, $\delta = 2$	4000	68.01	11.24	21.54	28.55%	52.56%	71.17	10.38	36.62	14.58%	51.46%
	10000	68.01	11.17	19.89	21.19%	50.84%	71.17	9.62	35.84	13.52%	50.35%
	20000	68.01	8.19	19.27	21.05%	50.37%	71.17	9.53	35.10	13.40%	49.32%
	40000	68.01	8.42	18.80	13.44%	50.41%	71.17	9.64	34.90	13.55%	49.04%
$\gamma = 0.25$, $\delta = 4$	4000	52.45	7.12	15.02	31.11%	47.81%	71.13	8.32	34.36	11.70%	48.30%
	10000	52.45	6.63	13.66	29.42%	47.49%	71.13	7.45	34.09	10.47%	47.92%
	20000	52.45	6.47	13.17	29.34%	48.35%	71.13	7.38	34.63	10.38%	48.68%
	40000	52.45	6.45	12.91	28.97%	48.81%	71.13	7.58	34.88	10.66%	49.04%
$\gamma = 0.5$, $\delta = 2$	4000	123.19	21.52	49.82	36.29%	70.12%	122.08	31.06	83.10	25.44%	68.07%
	10000	123.19	21.18	45.47	35.59%	67.66%	122.08	29.53	80.53	24.19%	65.96%
	20000	123.19	19.67	43.47	33.13%	67.08%	122.08	33.14	79.40	27.14%	65.04%
	40000	123.19	17.29	41.90	27.88%	66.53%	122.08	31.99	79.13	26.21%	64.82%
$\gamma = 0.5$, $\delta = 4$	4000	120.56	30.62	47.06	32.51%	64.28%	115.97	25.52	75.03	22.01%	64.70%
	10000	120.56	27.32	43.13	24.33%	62.85%	115.97	27.43	73.41	23.65%	63.30%
	20000	120.56	24.86	41.14	24.45%	63.03%	115.97	30.46	73.18	26.26%	63.10%
	40000	120.56	24.53	39.65	23.72%	62.55%	115.97	20.34	72.58	17.54%	62.59%

Average lower 29.98%

Average lower 38.81%

- G^b_{avg}: average number of green tokens ***before*** removal
- G^a_{avg}: average number of green tokens ***after*** removal
- GRR= G^a_{avg}/G^b_{avg}: the rate of remaining green tokens

7. Experiment

---Main Results (Multi-key)

Our Average Precision 76.70%
23% higher than the baseline

- AS2 attacker performance of 3-key green list **stealing** against LLaMA-2-7B

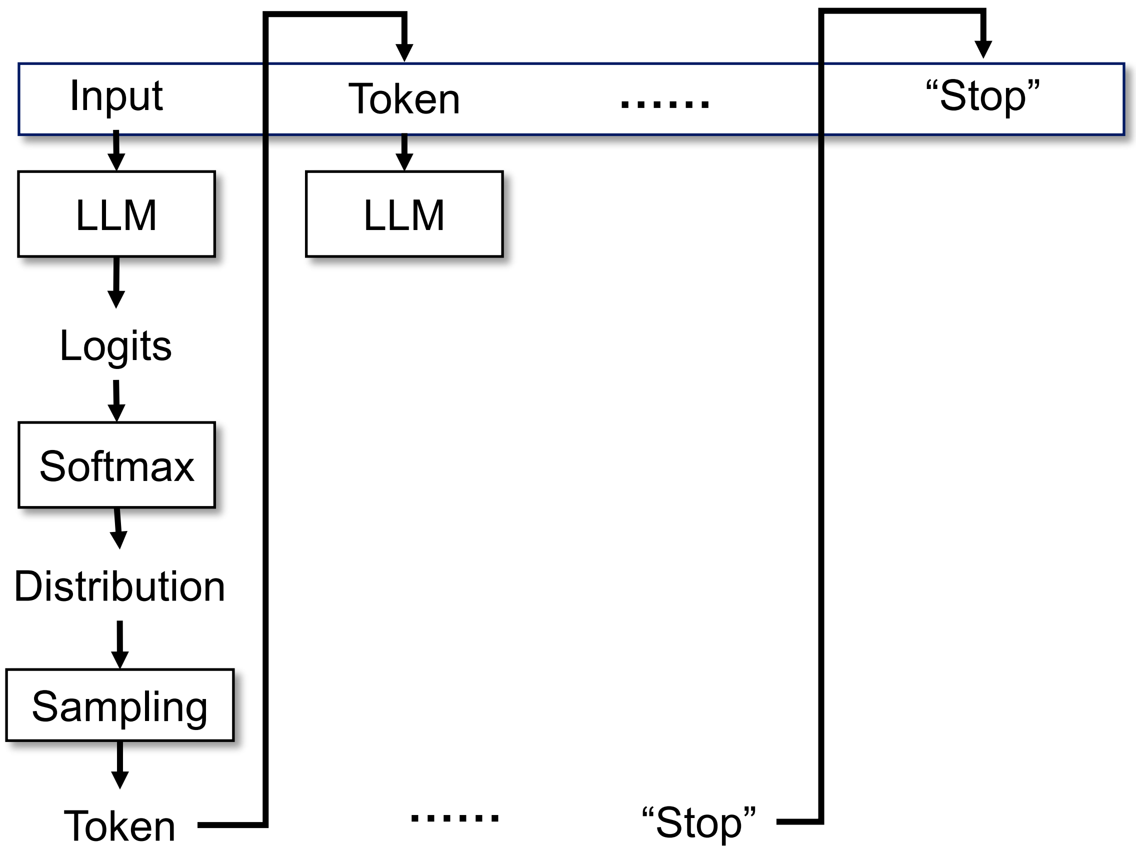
Model	γ	Dataset Size	Green List 1						Green List 2						Green List 3					
			Ours			Freq.			Ours			Freq.			Ours			Freq.		
			N_g	N_t	Precision(↑)	N_g	N_t	Precision(↑)	N_g	N_t	Precision(↑)	N_g	N_t	Precision(↑)	N_g	N_t	Precision(↑)	N_g	N_t	Precision(↑)
LLaMA	0.25	6000	2154	1383	0.6421	2000	821	0.4105	2141	1344	0.6277	2000	804	0.4020	2063	1302	0.6311	2000	796	0.3980
LLaMA	0.25	12000	1995	1513	0.7584	2000	836	0.4180	1995	1455	0.7293	2000	829	0.4145	1999	1418	0.7094	2000	810	0.4050
LLaMA	0.5	6000	2152	1946	0.9043	2000	1412	0.7060	2263	1935	0.8551	2000	1333	0.6665	2257	1737	0.7696	2000	1148	0.5740
LLaMA	0.5	12000	1998	1825	0.9134	2000	1433	0.7165	2002	1821	0.9096	2000	1334	0.6670	1997	1713	0.8578	2000	1151	0.5755
OPT	0.25	6000	3007	1957	0.6508	3000	1300	0.4333	3003	1918	0.6387	3000	1296	0.4320	2992	1959	0.6547	3000	1171	0.3903
OPT	0.5	6000	2995	2549	0.8511	3000	1954	0.6513	2997	2538	0.8468	3000	1888	0.6293	2996	2565	0.8561	3000	1886	0.6287

- AS2 attacker performance of **removal** for 3-key watermark against LLaMA-2-7B

Model	γ	Dataset Size	Green List 1					Green List 2					Green List 3				
			G^b_{avg}	G^a_{avg}(↓)		GRR(↓)		G^b_{avg}	G^a_{avg}(↓)		GRR(↓)		G^b_{avg}	G^a_{avg}(↓)		GRR(↓)	
				Ours	Freq.	Ours	Freq.		Ours	Freq.	Ours	Freq.		Ours	Freq.	Ours	Freq.
LLaMA	0.25	6000	77.75	47.19	72.19	60.68%	92.84%	69.77	38.95	62.59	55.82%	89.71%	75.55	35.42	67.69	46.88%	89.59%
LLaMA	0.25	12000	77.75	45.93	73.60	59.07%	94.66%	69.77	39.15	64.66	56.11%	92.67%	75.55	35.61	69.79	47.13%	92.38%
LLaMA	0.5	6000	121.75	86.20	118.36	70.80%	97.21%	130.12	95.09	126.01	73.08%	96.84%	99.87	78.16	97.90	78.26%	98.02%
LLaMA	0.5	12000	121.75	91.58	119.31	75.22%	97.99%	130.12	90.80	127.10	69.79%	97.69%	99.87	74.35	98.28	74.45%	98.40%
OPT	0.25	6000	80.04	44.72	75.09	55.87%	93.82%	78.87	43.27	75.45	54.86%	95.67%	75.15	40.09	71.21	53.35%	94.76%
OPT	0.5	6000	117.40	83.45	115.92	71.08%	98.74%	117.14	82.46	115.08	70.39%	98.24%	117.56	79.45	115.69	67.58%	98.41%

Thank You!

LLM Watermark
--- LLM generation without watermark



Green List Stealing
--- Attack Setting 1

Stage-1 constraints :

$$\mathrm{G}(S_i) \geq \hat{b}_i, \forall S_i \in \hat{S}$$
$$\hat{b}_i \geq g_i, \forall S_i \in \hat{S}$$

The number of green tokens in **watermarked sentences** should **larger** than threshold.

$$\mathrm{G}(S_i) \leq \tilde{b}_i, \forall S_i \in \tilde{S}$$
$$\tilde{b}_i \leq g_i, \forall S_i \in \tilde{S}$$

The number of green tokens in **natural sentences** should **smaller** than threshold.

- $\hat{b}_i$ and $\tilde{b}_i$: estimating the number of green tokens
- $\mathrm{G}(\cdot)$: The number of green tokens in a sentence
- g_i: Watermark threshold
- $\hat{S}$: Watermarked sentence
- $\tilde{S}$: Natural sentence

Green List Stealing
--- Attack Setting 1

Stage-1 objective :

maximize
$$\sum_{S_i \in \hat{S}} \hat{b}_i - abs\left(\sum_{S_i \in \tilde{S}} \tilde{b}_i - \gamma \cdot \sum_{S_i \in \tilde{S}} l_i\right),$$

Increase the number of green tokens for each **watermarked sentence.**

The number of green tokens in **natural sentences** remains **close to the average level.**

- l_i : the length of sentence S_i
- $\hat{S}$: Watermarked sentence
- $\tilde{S}$: Natural sentence

Green List Stealing
--- Attack Setting 1

Stage-2 constraints :
Let $\hat{b}_{sum} = \sum_{S_i \in \hat{S}} \hat{b}_i$, $\tilde{b}_{sum} = \sum_{S_i \in \tilde{S}} \tilde{b}_i$

Add new constraints:

$\sum_{S_i \in \hat{S}} \hat{b}_i \geq \hat{\beta} \cdot \hat{b}_{sum}$
$\sum_{S_i \in \tilde{S}} \tilde{b}_i \geq \tilde{\beta} \cdot \tilde{b}_{sum}$

- Based on the result of **stage-1**, we add new constrains to **bond the value of** $\hat{b}_i$ and $\tilde{b}_i$.

Stage-2 objective :

minimize $\sum_{t_j \in T} c_j$

- The **objective** of **stage-2** is to find the **minimal available green list.**

- T: Vocabulary
- c_j: The color of token t_j

Green List Stealing

--- Attack Setting 2

- Without verification by the watermark detector API, two types of erroneous samples emerge:
  - The LLM output lacks the watermark
  - Natural text is incorrectly labeled as watermarked.

We introduce binary variables $\lambda_i \in \{0, 1\}$ to determine whether sentence S_i should be included into the optimization.

Stage-1 constraints :

$$G(S_i) \geq \hat{b}_i + (\lambda_i - 1) \cdot l_i, \forall S_i \in \hat{S}$$
$$G(S_i) \leq \tilde{b}_i + (1 - \lambda_i) \cdot l_i, \forall S_i \in \tilde{S}$$

Green List Stealing
--- Multi-key Stealing

- In Multi-key scenario, the attacker need to find suitable key for each sentence.

Stage-1 constraints :

Which key is suitable for this sentence, $\sum_{k \in K} \hat{\rho}_i^k = 1, \forall S_i \in \hat{S}$

$$G(S_i, k) \geq \hat{b}_i^k + \left(\hat{\rho}_i^k - 1 + \lambda_i - 1\right) \cdot l_i, \forall S_i \in \hat{S}, k \epsilon K,$$
$$G(S_i, k) \leq \tilde{b}_i^k + (1 - \lambda_i) \cdot l_i, \forall S_i \in \tilde{S}, k \epsilon K,$$

Green List Stealing

--- Multi-key Stealing

Stage-1 objective :

maximize $\sum_{s_i \in \hat{S}} \hat{b}_i - \sum_{s_i \in \tilde{S}} \tilde{b}_i$

$$\hat{b}_i = \max_k \left(\hat{b}_i^k\right)$$
$$\tilde{b}_i = \max_k \left(\tilde{b}_i^k\right)$$

Max-Max problem, it is hard for directly optimization in mixed integer programming.

Find the max green number for each sentence.