Week 8: Trustworthiness Aspects II - Watermarking

Generative AI
Saarland University – Winter Semester 2024/25

Goran Radanovic
genai-w24-tutors@mpi-sws.org

Warning: Some of the references contain examples of model outputs with offensive language.

MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

MAX-PLANCK-GESELLSCHAFT

Outline of the Lecture

- Updates
- Recap: Red Teaming
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

Outline of the Lecture

- **Updates**
- Recap: Red Teaming
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

Updates

- **Week 7 assignment – deadline**: Dec 12, 6pm CET (**reminder**)

- **Week 8 assignment – deadline**: Dec 21, 6pm CET

- **Corrections and updates** for week7 slides/assignment:
  - Slide 20: $|V| \cdot (n - m - 1) \longrightarrow (|V| - 1) \cdot (n - m)$
  - Slide 22: improved the clarity of the content
  - Slide 25: $\mathrm{argmax} \longrightarrow \mathrm{argmin}$
  - Slide 40:$+\alpha \cdot D_{KL}(\cdot) \longrightarrow -\alpha \cdot D_{KL}(\cdot)$
  - Assignment, E4 and E5: $|\mathcal{J}| = (n - m - 1) \longrightarrow |\mathcal{J}| = (n - m)$

- **Next Lecture**: Jan 7, 10:15am CET (**reminder**)

Outline of the Lecture

- Updates
- **Recap: Red Teaming**
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

Recap

- Generative models pose additional challenges related to trustworthiness
  - **Previous lecture**: Red teaming

- Red teaming provides a systematic way of uncovering vulnerabilities
  - Jailbreaking: similar objective (bypassing safety restrictions) and often a part of red teaming efforts
  - Red teaming via crowdsourcing pipeline

- Automating the search process:
  - Jailbreaking: Adversarial suffix attacks
  - Red teaming: LLMs replace human red teamers

- How do we improve safety?

Improving Safety

Helpfulness vs. Harmlessness

- It is possible to make a model *safe* by ensuring it refuses to answer all queries
- The challenge lies in creating a model that is both helpful and harmless

Approach I: Designing more robust alignment approaches

- We can use safety fine-tuning with red teaming data
- **Example**: Safe RLHF (*Optional)

Preference Annotation

Input: $\mathcal{D} = \{(x_p, y_w, y_l)\}$

Helpfulness annot.:

- $y_w \succ y_l$

Harmlessness annot.:

- y_i is harmful/harmless

Modeling

Reward model: r_ϕ

- Based on $y_w \succ y_l$

Cost model: c_φ

- Based on y_i's safety

Optimization

Objective: Maximize $\mathbb{E}[r_\phi]$ subject to the constraint that $\mathbb{E}[c_\varphi]$ is upper bounded

Red team and repeat…

Improving Safety

Helpfulness vs. Harmlessness

- It is possible to make a model *safe* by ensuring it refuses to answer all queries
- The challenge lies in creating a model that is both helpful and harmless

Approach II: Designing defenses against adversarial inputs

- Detection-based: test the perplexity of the input, test the harmfulness of the output, etc.
- Mitigation-based: modifying the input, in-context defenses, etc.
- **Example**: Robust Prompt Optimization (*Optional)
  - Inspired by the idea of jailbreaking using adversarial suffix
  - Add a *defense* suffix to the received prompt
  - Apply adversarial training to optimize the defense suffix
    - Select the jailbreak that optimizes the adversarial loss
    - Optimize the defense suffix to ensure refusal is maintained

Ref: [Xu et al., 2024 (related work); Zhou et al., 2023]

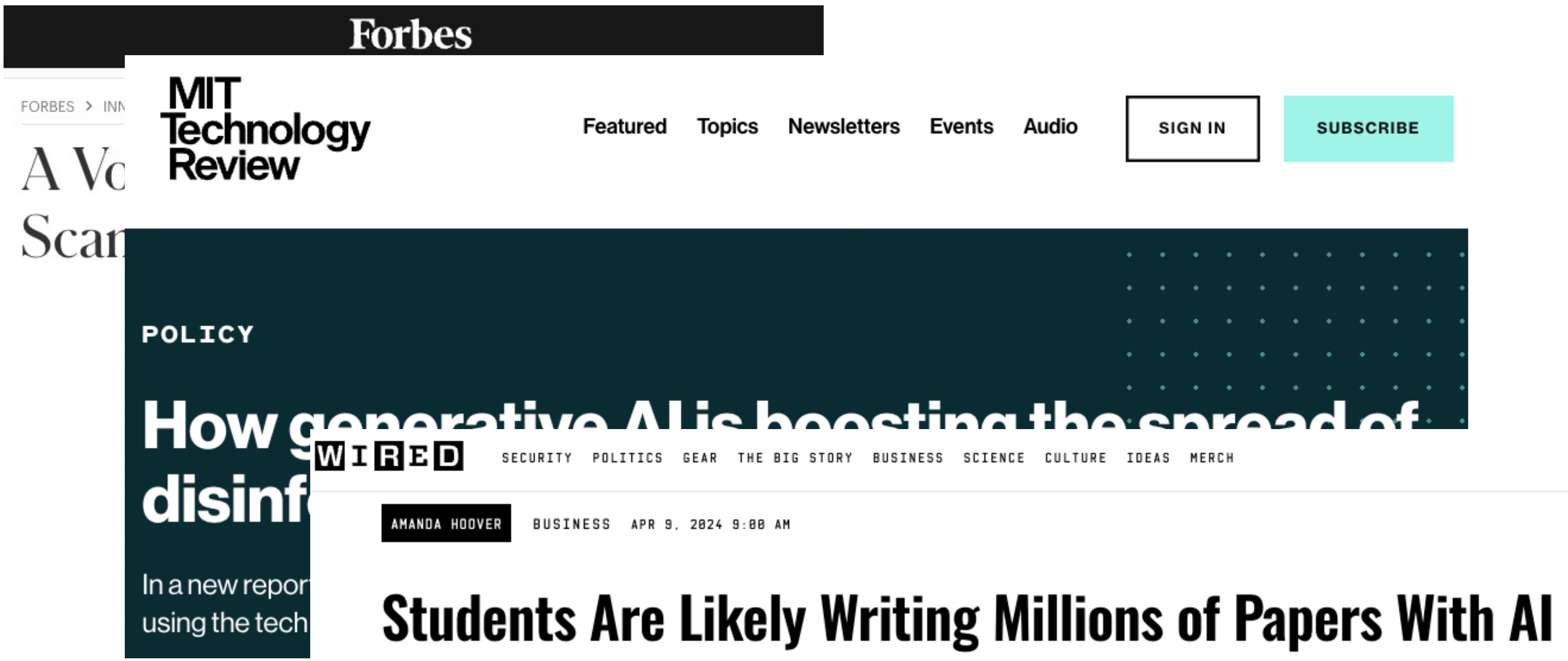
Outline of the Lecture

- Updates
- Recap: Red Teaming
- **Watermarking: Introduction**
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

Authenticity Challenges

- It is becoming increasingly difficult to distinguish synthetic content from authentic content

- Synthetic content causes different societal harms…



Authenticity Challenges

- It is becoming increasingly difficult to distinguish synthetic content from authentic content

- Legislation, such as the EU AI Act, acknowledges the need for developing tools to track synthetic content

> *"In light of those impacts, the fast technological pace and the need for new methods and techniques to trace origin of information, it is appropriate to require providers of those systems to embed technical solutions that enable marking in a machine readable format and detection that the output has been generated or manipulated by an AI system and not a human. Such techniques and methods should be sufficiently reliable, interoperable, effective and robust as far as this is technically feasible, taking into account available techniques or a combination of such techniques, such as* ***watermarks****, metadata identifications, cryptographic methods for proving provenance and authenticity of content, logging methods, fingerprints or other techniques, as may be appropriate."*

Ref: [EU AI Act, 2024]

Synthetic Content Detection

- **Retrieval-based approach**: maintain a record of generated content
  - It's not clear how to coordinate effort at scale

- **Developing standards for provenance information**, e.g, the C2PA standard
  - Provenance metadata can be removed
  - C2PA in DALLE-3 (link): *Metadata like C2PA is not a silver bullet to address issues of provenance. It can easily be removed either accidentally or intentionally.*

- **Post-hoc detection**: train a classifier to distinguish human-generated content from synthetic content
  - Such classifiers don't have consistent performance
  - An OpenAI classifier (link): *As of July 20, 2023, the AI classifier is no longer available due to its low rate of accuracy.*

- **Next**: Watermarking

Ref: [Dathathri et al., 2024; C2PA, 2024]

Watermarking

- Mark the generated content so that it can be identified

Approaches

- Edit-based: Editing the response of the model
- Data-driven: Modifying the training data
- Generative: Modifying the generative process ← **this lecture**

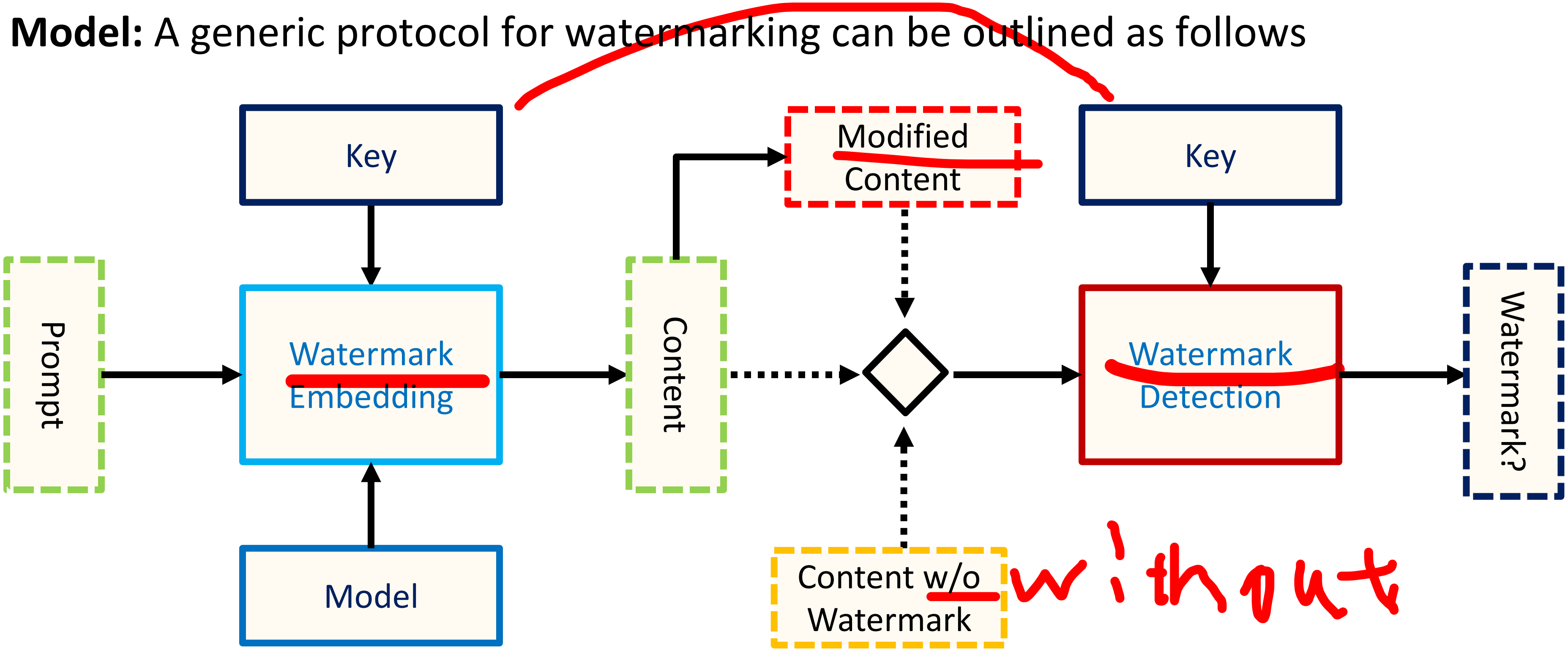
Desirable Properties

- **Quality**: Watermarking does not decrease the quality of generated content.
- **Soundness**: Non-watermarked content is not falsely classified as watermarked.
- **Robustness**: We can detect the watermark, even if watermarked content is altered.
- **Undetectability**: Without the key (next slide), the outputs of the watermarked model are computationally indistinguishable from those of the original model. ← **not covered in this lecture**

Ref: [Dathathri et al., 2024; Christ and Gunn, 2024]

Generative Watermarking

Model: A generic protocol for watermarking can be outlined as follows



- We will primarily focus on:
  - Public watermarking protocols (w/o a key) for LLMs
  - Watermarking algorithms that have black box access to the model (including logits)

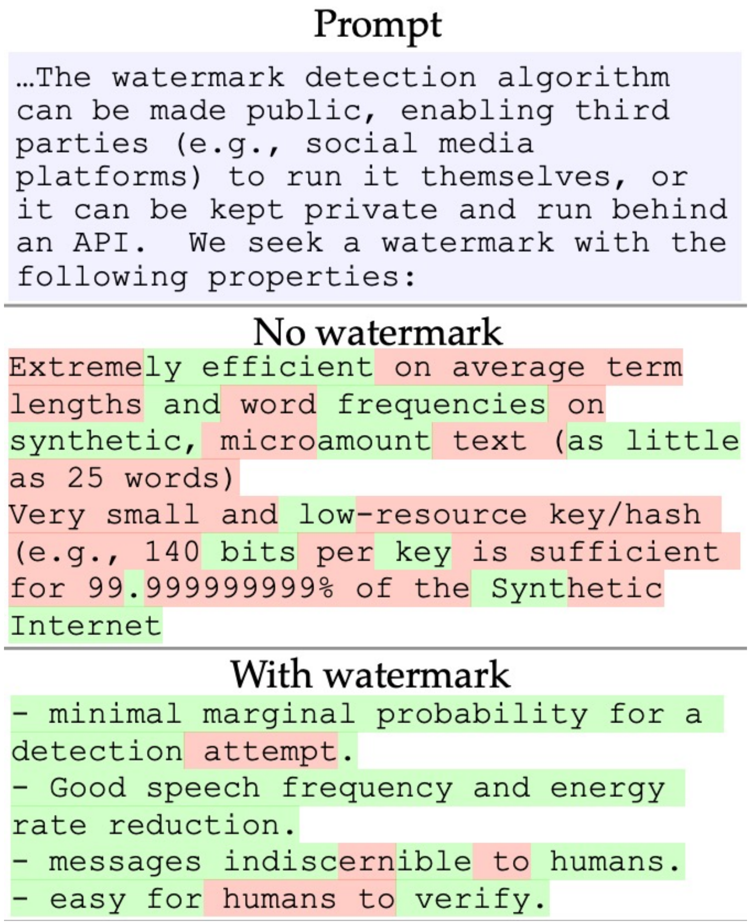
Toward Watermarking Text

- **Idea**: bias the token generation process toward a subset of tokens – this subset is chosen on the fly, for each token position separately

When generating the next token, split the vocabulary into a list of green and red tokens

Human written text is expected to have the same fraction of green and red tokens

With watermarking, green tokens are more frequent than red tokens



Outline of the Lecture

- Updates
- Recap: Red Teaming
- Watermarking: Introduction
- **Watermark Embedding: Text Generation with Hard Red List**
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

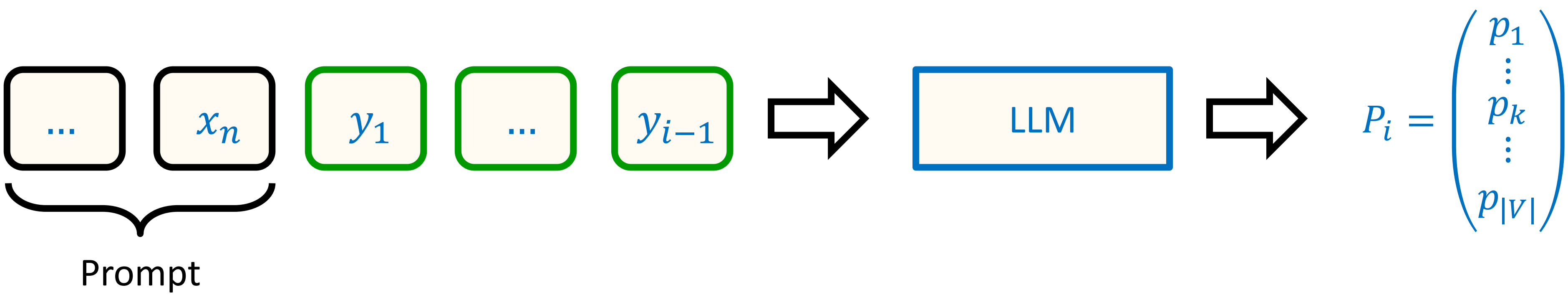
Watermarking: Hard Red List

- This approach induces a strong bias toward the green list: only generate tokens from the green list

Watermark Embedding: Text Generation with Hard Red List

- Generating the i-th token in the sequence

- Step 1: Evaluate the probabilities $P(\cdot\,|x_{1:n}\,y_{1:i-1})$ over the vocabulary V



Watermarking: Hard Red List

- This approach induces a strong bias toward the green list: only generate tokens from the green list

Watermark Embedding: Text Generation with Hard Red List

- Generating the i-th token in the sequence

- Step 2: Compute the hash of token y_{i-1} and use it as a seed in RAND

y_{i-1} ⇨ Hash ⇨ $\mathrm{RAND(hash)}$

Ref: [Kirchenbauer et al., 2023]

Watermarking: Hard Red List

- This approach induces a strong bias toward the green list: only generate tokens from the green list

Watermark Embedding: Text Generation with Hard Red List

- Generating the i-th token in the sequence

- Step 3: Split vocabulary V in two different lists, red and green

$$V = \{t_1, \dots, t_{|V|}\} \Rightarrow \boxed{\mathrm{RAND(hash)}} \Rightarrow \begin{array}{l} G_i = \{t_{G1}, \dots, t_{Gm_i}\} \\ R_i = \{t_{R1}, \dots, t_{Rn_i}\} \end{array}$$

The lists are balanced, $m_i = n_i$

Watermarking: Hard Red List

- This approach induces a strong bias toward the green list: only generate tokens from the green list

Watermark Embedding: Text Generation with Hard Red List

- Generating the i-th token in the sequence

- Step 4: Sample from G using P_i with sampling strategy SAMPLE

$$G_{\mathrm{i}}, P_i \Rightarrow \boxed{\text{SAMPLE}} \Rightarrow \boxed{y_i}$$

SAMPLE can be standard multinomial sampling or greedy decoding

- Repeat Steps 1-4 until the full sequence is generated

Watermarking: Hard Red List

- This approach induces a strong bias toward the green list: only generate tokens from the green list

Text Generation with Hard Red List (Algorithm 1 from the reference)

- Generating the i-th token in the sequence

- Step 1: Evaluate the probabilities $P(\cdot \mid x_{1:n}\ y_{1:i-1})$ over the vocabulary V
- Step 2: Compute a hash of token y_{i-1} and use it as a seed in RAND
- Step 3: Split vocabulary V in two different lists, red and green
- Step 4: Sample from G_i using P_i with sample strategy SAMPLE
- Repeat Steps 1-4 until the full sequence is generated

Week 8 Assignment

- The assignment focuses on the properties of the hard red list watermarking

Outline of the Lecture

- Updates
- Recap: Red Teaming
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- **Watermark Detection for Text Generation with Hard Red List**
- Watermarking: Text Generation with Soft Red List
- Additional Considerations

Quiz – Watermark Detection

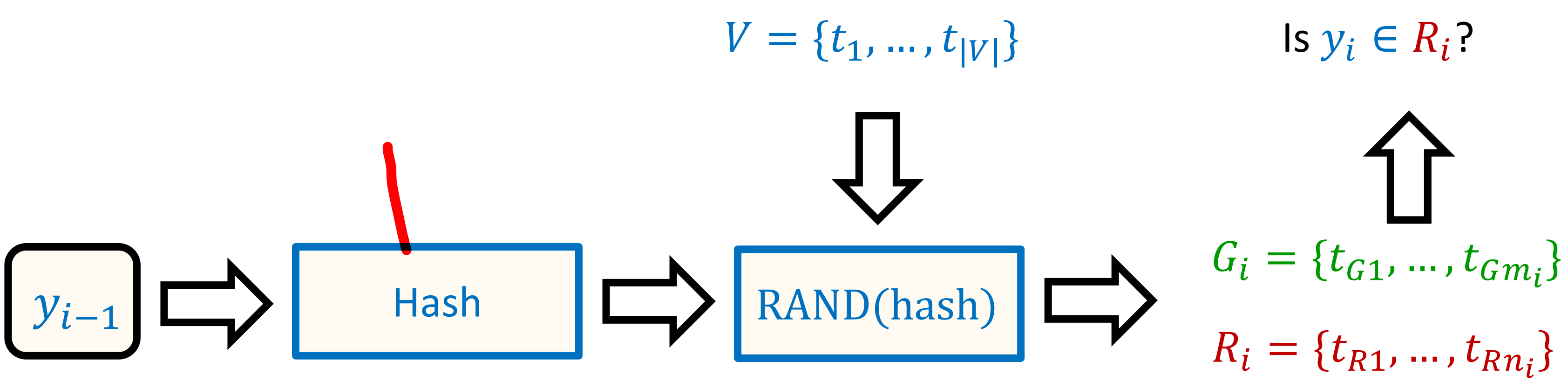
- **Q**: What do we need to test for in order to detect a watermark for a given sequence of tokens?
- **A**: How likely is it that the sequence was generated without following the red list rule?
- We can use hypothesis testing to determine whether the sequence was watermarked

Watermarking: Hard Red List

- Watermark detection is a hypothesis test with the null hypothesis:
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the hard red list algorithm
- Step 1: For each token position i check whether y_i is an element of G_i or R_i



Watermarking: Hard Red List

- Watermark detection is a hypothesis test with the null hypothesis:
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the hard red list algorithm
- Step 1: For each token position i check whether y_i is an element of G_i or R_i

- When can $x_i \in R_i$?
  - Case 1: y_i is not following the red list rule restriction
  - Case 2: y_{i-1} is not correct (e.g., due to an adversary) $\Longrightarrow$ discussed later

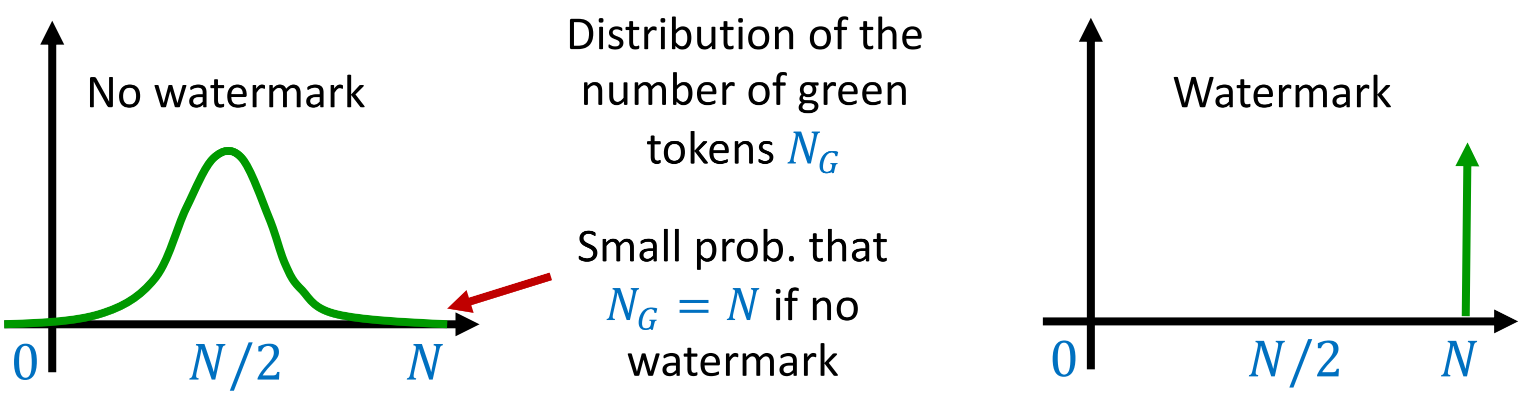
eg

Watermarking: Hard Red List

- Watermark detection is a hypothesis test with the null hypothesis:
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the hard red list algorithm
- Step 2: Hypothesis testing
  - If H_0 holds, the expected number of green tokens is $N/2$, where N is the length of the sequence; the variance is $N/4$

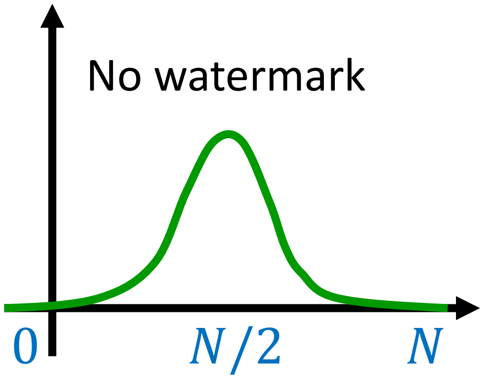


Watermarking: Hard Red List

- Watermark detection is a hypothesis test with the null hypothesis:
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the hard red list algorithm
- Step 2: Hypothesis testing
  - If H_0 holds, the expected number of green tokens is $N/2$, where N is the length of the sequence; the variance is $N/4$



Naïve test: Reject H_0 only if $N_G = N$

More robust test: Reject H_0 if observing $\geq N_G$ green tokens under no watermark is unlikely

Watermarking: Hard Red List

- Watermark detection is a hypothesis test with the null hypothesis:
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the hard red list algorithm
- Step 2: Hypothesis testing
  - If H_0 holds, the expected number of green tokens is $N/2$, where N is the length of the sequence; the variance is $N/4$
- One proportion z-test
  - **Intuition**: Test if the number of green tokens N_G is significantly greater than $N/2$
  - Calculate z-statistic: $z = (N_G - \text{mean})/\sqrt{\text{var}} = 2(N_G - N/2)/\sqrt{N}$
  - Reject H_0 if z is above a certain threshold
    - **Type I error** : For $z > 4$, the false positive rate is $3 \cdot 10^{-5}$

Quiz – Hard Red List

- **Q**: What would happen if we prompt the model to output "*Computing Machinery and Intelligence*" by Alan Turing?

- **A**: The correct output is deterministic, and the watermarking algorithm would significantly change it
  - **Intuition**: For about $N/2$ token positions, the correct token would be in red list

- We need sufficient entropy in order for watermarking to preserve quality

- However, the hard red list approach is quite rigid: the approach may be problematic even for shorter phrases like <*name*> <*surname*>

- Can we design a more flexible approach?

Outline of the Lecture

- Updates
- Recap: Red Teaming
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- **Watermarking: Text Generation with Soft Red List**
- Additional Considerations

Watermarking: Soft Red List

- **Main idea**: bias the sampling to favor the tokens from the green list, but allow red tokens

Text Generation with Soft Red List (Algorithm 2 from the reference)

- Generating the i-th token in the sequence

- Step 1: Evaluate the logits $l(\cdot\,|x_{1:n}\, y_{1:i-1}) = \left(l_1, \ldots, l_{|V|}\right)^T$ over the vocabulary V
- Step 2: Compute a hash of token y_{i-1} and use it as a seed in RAND
- Step 3: Split vocabulary V in two different lists, red and green
  - **Unbalanced lists**: $|G_i| = \gamma|V|$ vs. $|R_i| = (1-\gamma)|V|$, where $\gamma \in (0,1)$
- Step 4: Sample token y_i from the prob. vector $\tilde{P}_i = \left(\tilde{p}_1, \ldots, \tilde{p}_k, \ldots, \tilde{p}_{|V|}\right)^T$ where:

$$\tilde{p}_k = \frac{e^{l_k+\delta}}{\sum_{j\in G_i} e^{l_j+\delta} + \sum_{j\in R_i} e^{l_j}} \text{ if } k \in G_i \quad \& \quad \tilde{p}_k = \frac{e^{l_k}}{\sum_{j\in G_i} e^{l_j+\delta} + \sum_{j\in R_i} e^{l_j}} \text{ if } k \in R_i$$

Watermarking: Soft Red List

- Watermark detection follows the same steps as in the hard red list approach, but with a different z-statistics
  - H_0 - the text is generated with no knowledge of the red list rule

Watermark Detection

- **Required**: the hash function of the soft red list algorithm
- Step 2: Hypothesis testing
  - If H_0 holds, the expected number of green tokens is γN, where N is the length of the sequence; the variance is $N\gamma(1-\gamma)$

- One proportion z-test
  - **Intuition**: Test if the number of green tokens N_G is significantly higher γN
  - Calculate z-statistic: $z = (N_G - \text{mean})/\sqrt{\text{var}} = (N_G - \gamma N)/\sqrt{N\gamma(1-\gamma)}$
  - Reject H_0 if z above a certain threshold
    - **Type I error**: Again, for $z > 4$, the false positive probability is $3 \cdot 10^{-5}$

Watermarking: Soft Red List

- Watermark detection follows the same steps as in the hard red list approach, but with a different z-statistics
  - H_0 - the text is generated with no knowledge of the red list rule

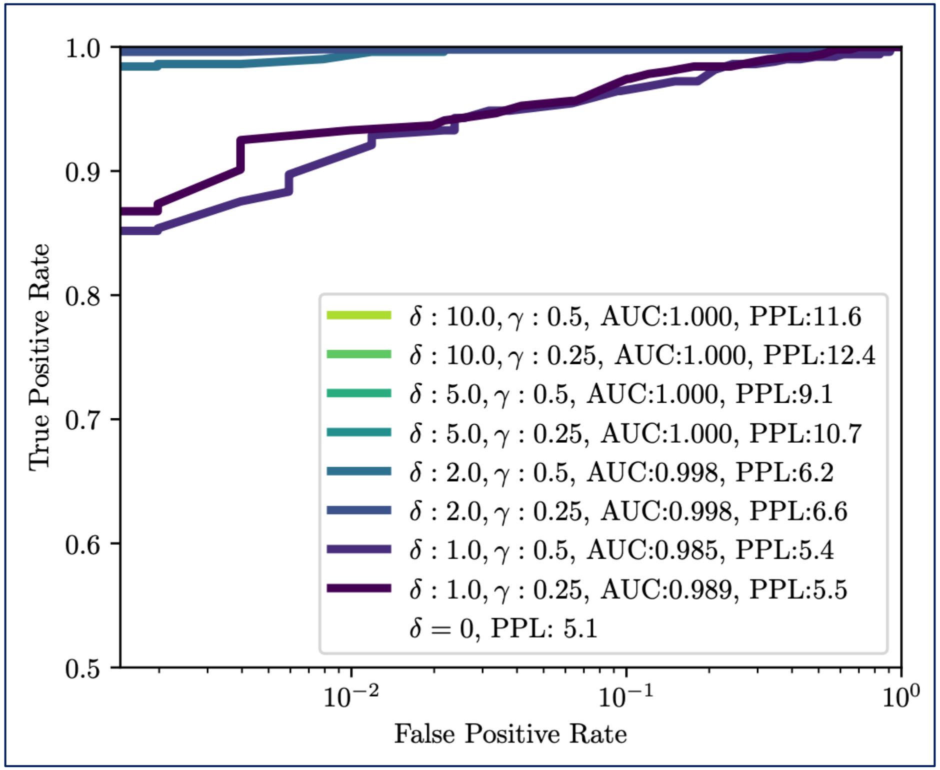
Watermark Detection

- For the hard red list approach, type II error (false negative rate) is 0
  - Watermarked text only contains green tokens
  - **Remark**: N should be large enough (e.g., $N > 16$ if the z-threshold is equal to 4)

- However, type II error in the soft red list approach depends on the entropy of the sequence and its length
  - **Intuition**: Bias δ increases the fraction of green tokens from γ to γ_δ giving us $z = (\gamma_\delta - \gamma)N/\sqrt{N\gamma(1-\gamma)} = (\gamma_\delta - \gamma)\sqrt{N}/\sqrt{\gamma(1-\gamma)}$
  - Low entropy sequences yield smaller $\gamma_\delta - \gamma$; however, larger N can compensate for this, ensuring that z exceeds the detection threshold, and hence, resulting in a smaller type II error

Watermarking: Soft Red List

- How do γ and δ affect the performance?



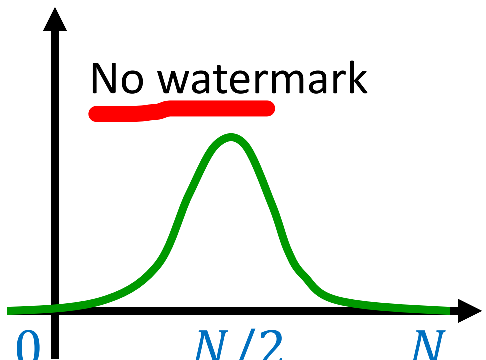
- As δ increases, both AUC and perplexity increase as well
- A lower value of γ results in higher perplexity

Outline of the Lecture

- Updates
- Recap: Red Teaming
- Watermarking: Introduction
- Watermark Embedding: Text Generation with Hard Red List
- Watermark Detection for Text Generation with Hard Red List
- Watermarking: Text Generation with Soft Red List
- **Additional Considerations**

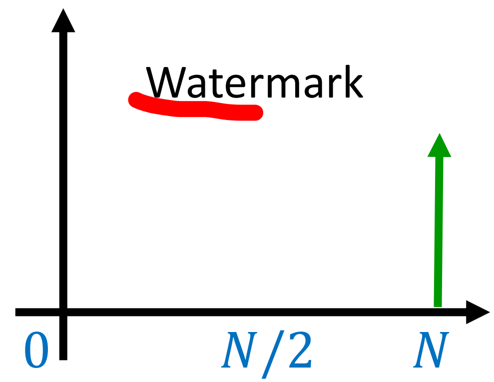
Robustness Considerations

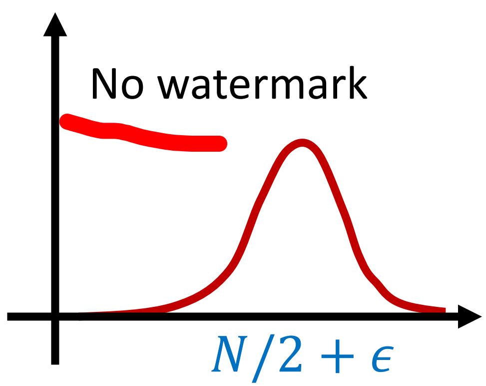
- Intuitively, watermarks have some level of corruption-robustness
- **Intuition:** Bunded corruption cannot arbitrarily change the distributions of the number of green tokens



No corruption

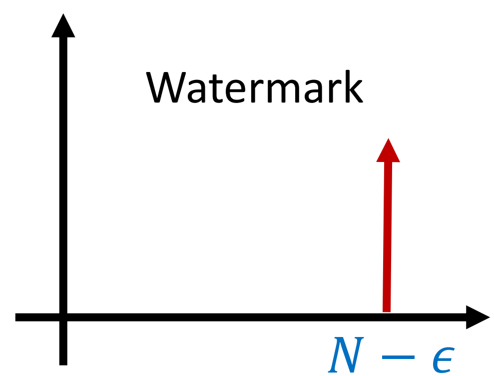
Distributions of N_G





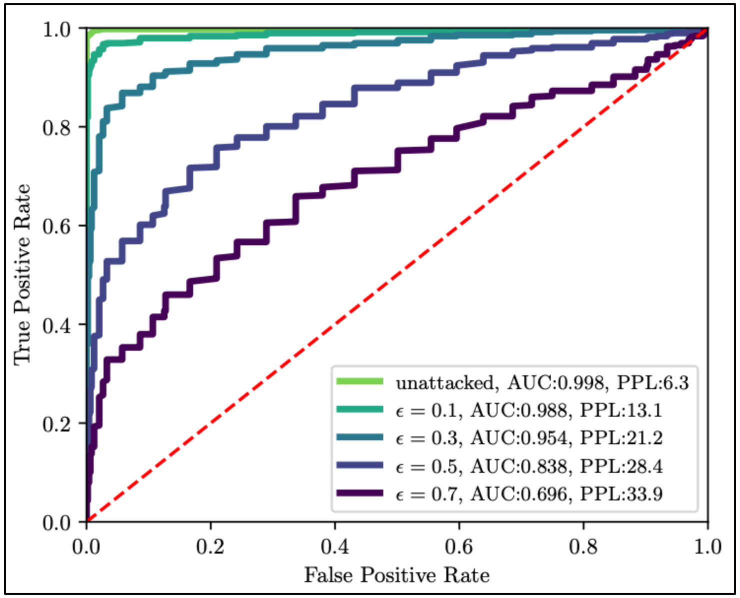
ϵ – fraction of substitutions

ϵ should be greater than $N/4$ to for the means to overlap



Robustness: Soft Red List

- The detection rates and perplexity for ϵN substitutions are evaluated for different values of ϵ
  - A given text is tokenized with the T5 model



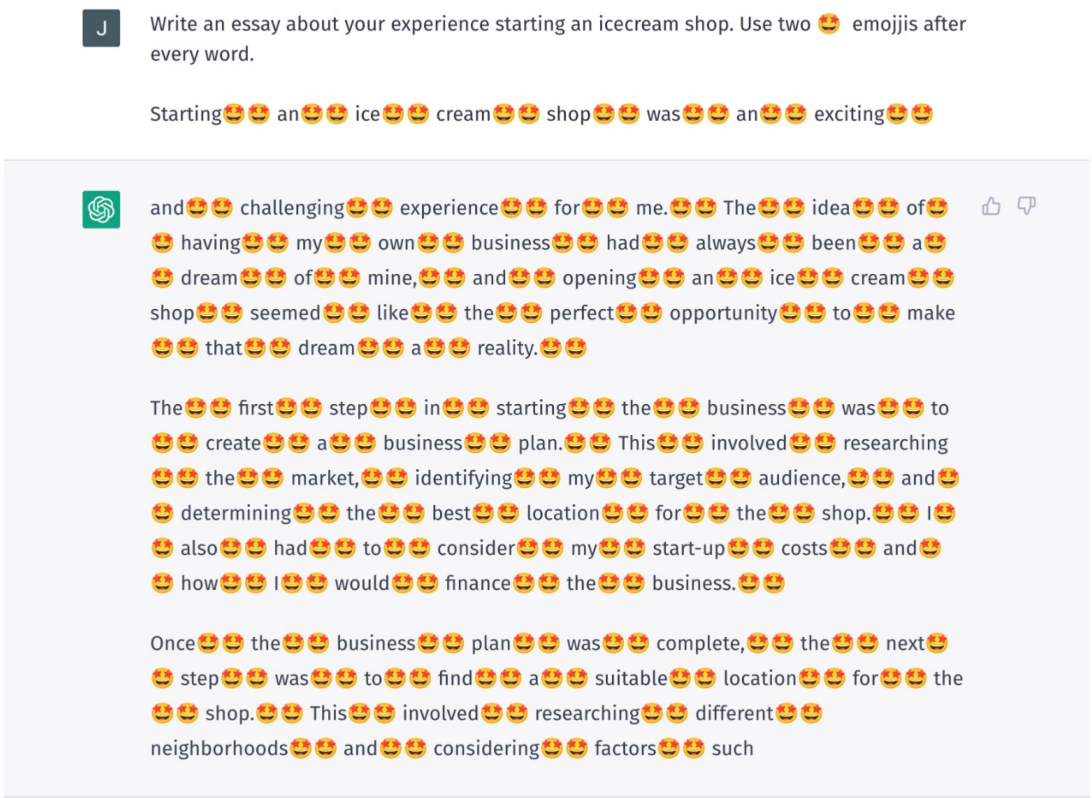
- Noticeable performance drops when for larger values of ϵ

Week 8 Assignment

- The implementation part: testing the robustness of the hard red list approach

Emoji Attack

- However, the method is not provably robust to watermarking attacks…



Week 8 Assignment

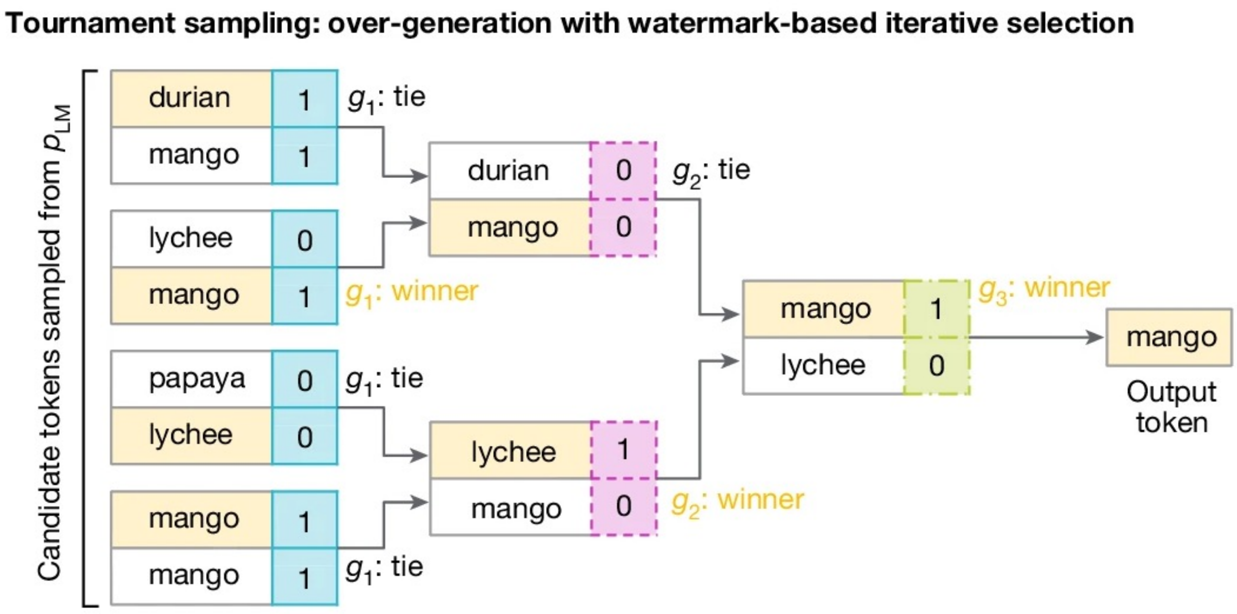
- Exercises on deletion/insertion-based watermarking attacks…

Watermarking in Practice

- SynthID watermarking by Google DeepMind

Watermarking in Practice

- SynthID-Text is available on Hugging Face
- It uses a more complex tournament sampling algorithm with multiple random watermarking functions g that assign a score to any candidate token



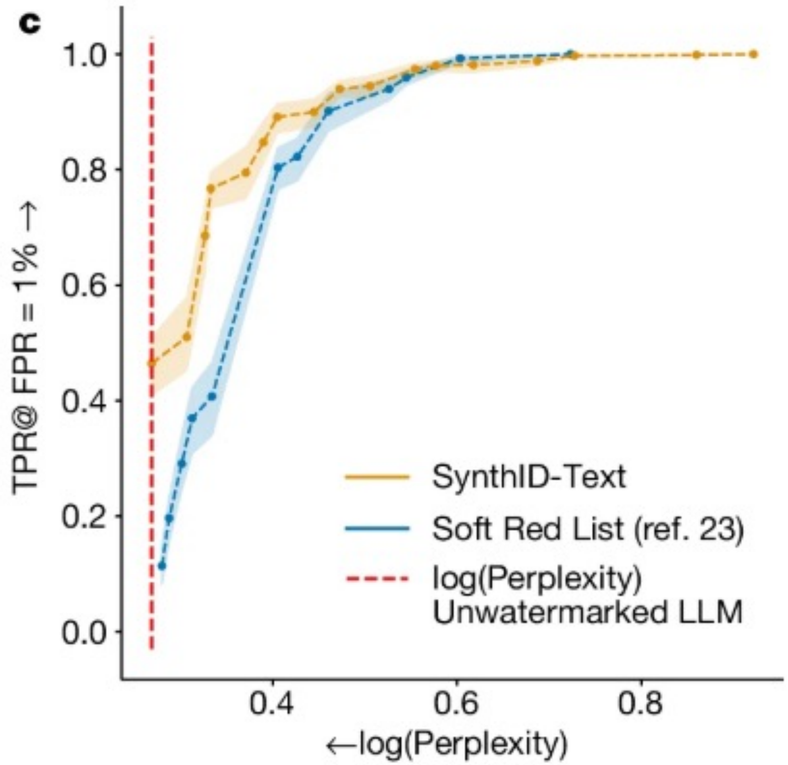
- Detection is based on computing the mean g-values of the text

Watermarking in Practice

- SynthID-Text evaluations
  - Compared to the soft red list approach it provides a more favorable quality vs. detectability trade-off

- Experimental setup
  - Model: Gemma 7B-IT
  - Response lengths: 200 tokens
  - FPR is set to 1%
  - Shaded regions: 90% conf. interv.

- Robustness?
  - Some robustness to edits
  - Susceptible to paraphrasing attacks



Watermarking in Practice

- SynthID-Text evaluations
  - Evaluated based on 20 million watermarked and unwatermarked responses from Gemini users
  - Human ratings indicate no change in the LLM capabilities
- **Example**:

Prompt

Write a message wishing happy and relaxing holidays to the students of the Generative AI course at Saarland University.

All generations, in random order

Select the generations you think are watermarked!

- [ ] As we approach the end of the semester and prepare for the holidays, I want to take a moment to wish the students of the Generative AI course a happy and relaxing holiday season. May your holidays be filled with peace, joy, and rejuvenation. I hope that the coming semester brings you even more exciting opportunities to explore the fascinating world of generative AI. Best regards, [Your name]
- [ ] Dear Students, As the festive season approaches, I extend my warmest wishes for a delightful and joyous holiday season. May the holidays bring you endless happiness, relaxation, and rejuvenation. The generative AI course is a vibrant learning community where you can explore the fascinating world of AI and hone your skills in a supportive and stimulating environment. I hope that the holidays provide you with the opportunity to rest, recharge, and connect with loved ones. Take this time to embrace the spirit of the season and let your creativity flourish. Wishing you a happy and relaxing holiday season, and all the best for a successful and fulfilling new year. Sincerely, [Your Name]

Which of the two responses are watermarked?

Ref: [Dathathri et al., 2024]; Link: https://huggingface.co/spaces/google/synthid-text

Watermarking in Practice

- SynthID-Text evaluations
  - Evaluated based on 20 million watermarked and unwatermarked responses from Gemini users
  - Human ratings indicate no change in the LLM capabilities
- **Example**:

Prompt

Write a message wishing happy and relaxing holidays to the students of the Generative AI course at Saarland University.

All generations, in random order

Select the generations you think are watermarked!

- [ ] As we approach the end of the semester and prepare for the holidays, I want to take a moment to wish the students of the Generative AI course a happy and relaxing holiday season. May your holidays be filled with peace, joy, and rejuvenation. I hope that the coming semester brings you even more exciting opportunities to explore the fascinating world of generative AI. Best regards, [Your name]
- [ ] Dear Students, As the festive season approaches, I extend my warmest wishes for a delightful and joyous holiday season. May the holidays bring you endless happiness, relaxation, and rejuvenation. The generative AI course is a vibrant learning community where you can explore the fascinating world of AI and hone your skills in a supportive and stimulating environment. I hope that the holidays provide you with the opportunity to rest, recharge, and connect with loved ones. Take this time to embrace the spirit of the season and let your creativity flourish. Wishing you a happy and relaxing holiday season, and all the best for a successful and fulfilling new year. Sincerely, [Your Name]

- [x] As we approach the end of the semester and prepare for the holidays, I want to take a moment to wish the students of the Generative AI course a happy and relaxing holiday season. May your holidays be filled with peace, joy, and rejuvenation. I hope that the coming semester brings you even more exciting opportunities to explore the fascinating world of generative AI. Best regards, [Your name]

Indicates watermarked text

- [ ] Dear Students, As the festive season approaches, I extend my warmest wishes for a delightful and joyous holiday season. May the holidays bring you endless happiness, relaxation, and rejuvenation. The generative AI course is a vibrant learning community where you can explore the fascinating world of AI and hone your skills in a supportive and stimulating environment. I hope that the holidays provide you with the opportunity to rest, recharge, and connect with loved ones. Take this time to embrace the spirit of the season and let your creativity flourish. Wishing you a happy and relaxing holiday season, and all the best for a successful and fulfilling new year. Sincerely, [Your Name]

Improving Robustness (*Optional)

Main idea: We control the generation process, so our watermark can be based on error-correcting codes

Watermark Embedding (simplified)

- **Required**: binary codeword $c = (c_1, \dots, c_N)$ (this can be generated – see below)
- Follow the same steps as before, with a modified Step 3
  - Step 3: Split vocabulary V in two different lists, red and green:
    - If 1-bit hash value of a token is equal to c_i, we put it in G_i; otherwise, we put it in R_i

Watermark Detection (simplified)

- Calculate the 1-bit hash of each token to obtain $\tilde{c}$; if $\mathrm{ErrorCorrect}(\tilde{c}) = c$, watermark detected
- Why does this help? We can generate c using keyed error-correction schemes (e.g., pseudorandom codes), which provide guarantees regarding undetectability and robustness

See e.g. [Christ and Gunn, 2024; Golowich and Moitra, 2024] for more advanced methods

Quiz - Diffusion Models (*Optional)

- **Q**: How could we utilize the ideas presented before for watermarking LLMs to watermark diffusion models?

- One possible approach is based on the Tree-Ring watermarking

- **Watermark Embedding**
  - We start with $\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \mathbb{I})$
  - Modify this sample to include a watermark
    - Embed a key in the Fourier space of $\boldsymbol{x}_N$

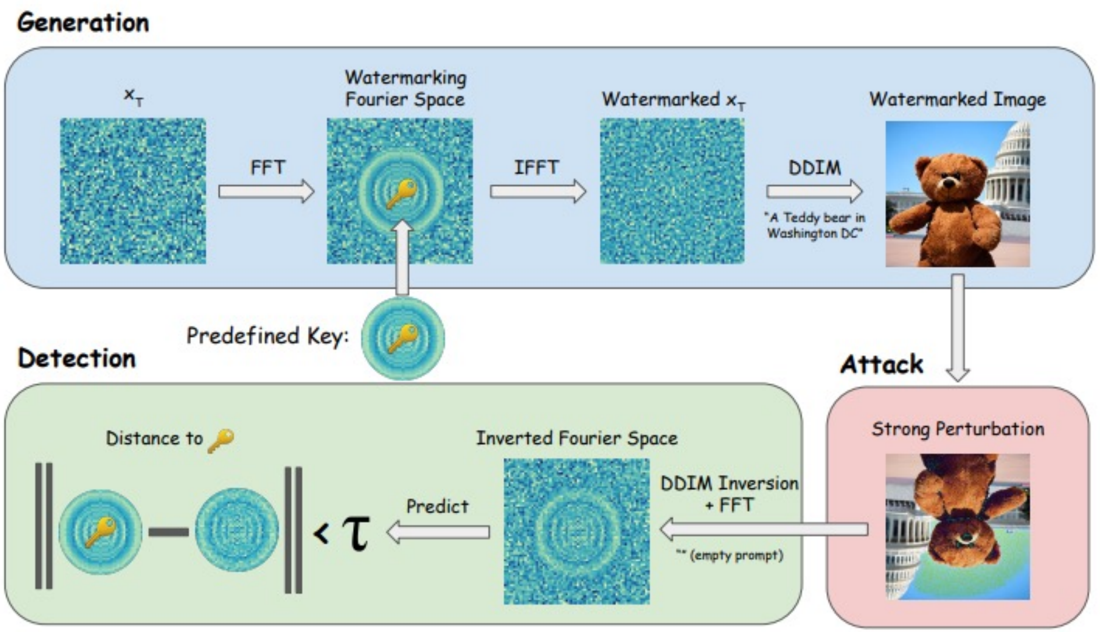
- **Watermark Detection**
  - We approximate the initial noise $\widetilde{\boldsymbol{x}}_N$ from $\widetilde{\boldsymbol{x}}_0$
  - Test for the presence of the key in the Fourier space of $\widetilde{\boldsymbol{x}}_N$

Quiz - Diffusion Models (*Optional)

- **Q**: How could we utilize the ideas presented before for watermarking LLMs to watermark diffusion models?

A

- One possible approach is based on the Tree-Ring watermarking



- Undetectable watermarking can be achieved using pseudorandom codes

References

- Dai et al., Safe RLHF: Safe Reinforcement Learning from Human Feedback, 2024.
- Xu et al., SafeDecoding: Defending against Jailbreak Attacks via Safety-Aware Decoding, 2024.
- Zhou et al., Robust Prompt Optimization for Defending Language Models Against Jailbreaking Attacks, 2024.
- EU AI Act: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=OJ:L_202401689, 2024.
- Dathathri et al., Scalable Watermarking for Identifying Large Language Model Outputs, 2024.
- C2PA: https://c2pa.org/, accessed December 2024.
- Christ and Gunn, Pseudorandom Error-Correcting Codes, 2024.
- Kirchenbauer et al., A Watermark for Large Language Models, 2023.
- Golowich and Moitra, Edit Distance Robust Watermarks for Language Models, 2024.
- Wen et al., Tree-Ring Watermarks: Fingerprints for Diffusion Images that are Invisible and Robust, 2023.
- Gunn et al., An Undetectable Watermark for Generative Image Models, 2024.

- **Optional**: For recent results on watermarking with provable guarantees, please visit the workshop on *Alignment, Trust, Watermarking, and Copyright Issues in LLMs* held at the *Simons Institute for the Theory of Computing* (link: https://simons.berkeley.edu/workshops/alignment-trust-watermarking-copyright-issues-llms)