BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Design and Evaluation of a Robust Watermarking Algorithm for Large Language Models in Bangla Text Generation

ANM Zahid Hossain Milkan
200041202
Amit Bin Tariqul
200041214
Sahab-Al-Chowdhury
200041255

Department of Computer Science and Engineering
Islamic University of Technology
April, 2025

Design and Evaluation of a Robust Watermarking Algorithm for Large Language Models in Bangla Text Generation

ANM Zahid Hossain Milkan
200041202
Amit Bin Tariqul
200041214
Sahab-Al-Chowdhury
200041255

Department of Computer Science and Engineering
Islamic University of Technology
April, 2025

Declaration of Candidate

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by **ANM Zahid Hossain Milkan**, **Amit Bin Tariqul**, and **Sahab-Al-Chowdhury** under the supervision of **Syed Rifat Raiyan**, Lecturer, Department of Computer Science and Engineering and co-supervision of **Dr. Kamrul Hasan**, Professor, Department of Computer Science and Engineering and **Dr. Hasan Mahmud**, Professor, Department of Computer Science and Engineering, Islamic University of Technology, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

Syed Rifat Raiyan
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: April 29, 2025

Dr. Kamrul Hasan
Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: April 29, 2025

Dr. Hasan Mahmud
Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: April 29, 2025

ANM Zahid Hossain Milkan
Student ID: 200041202
Date: April 29, 2025

Amit Bin Tariqul
Student ID: 200041214
Date: April 29, 2025

Sahab-Al-Chowdhury
Student ID: 200041255
Date: April 29, 2025

Dedicated to our supervisors without whom this work would not have been possible

Contents

List of Figures

List of Tables

List of Abbreviations

LLM	Large Language Model
RTT	Round Trip Translation
NLP	Natural Language Processing
KGW	Kaltenbacher-Goldblum-Watermarking
EXP	Exponential Sampling
CWRA	Cross-lingual Watermark Removal Attack
SIR	Semantic Invariant Robust Watermarking
X-SIR	Cross-lingual Semantic Invariant Robust Watermarking
ITS	Inverse Transform Sampling
PRF	Pseudorandom Function
LSH	Locality Sensitive Hashing
PCC	Pearson Correlation Coefficient
RE	Relative Error
SBERT	Sentence-BERT
MCMC	Markov Chain Monte Carlo
IP	Intellectual Property
MEA	Model Extraction Attack
BAICHUAN-7B	A bilingual LLM (Chinese-English)
WATERFALL	Watermarking Framework Applying Large Language Models
UW	Unigram Watermarking

Acknowledgement

We are profoundly grateful to our supervisor, **Syed Rifat Raiyan**, for his endless patience, insightful critiques. His expertise and encouragement were instrumental in the completion of this so far.

A special thanks to our respected Professors of Systems And Softwares Lab(SSL) **Dr. Kamrul Hasan and Dr. Hasan Mahmud** who provided invaluable support and advice. Finally, to our parents, whose love and support have been a constant source of strength. Thank you for always being there.

To all of you, we extend our heartfelt appreciation.

Abstract

As LLMs continue to reach newer dimensions, ensuring the responsible use of AI and authenticity has become a concern.While watermarking techniques have been extensively developed for English and other high-resource languages, low-resource languages like Bangla remain relatively unexplored.This thesis presents a robust watermarking approach tailored for Bangla-language Large Language Models (LLMs), addressing the challenges posed by Bangla's rich morphology and low-resource ecosystem.Evaluating existing approaches under various adversarial settings we introduce a Bangla-aware watermarking strategy that preserves semantic integrity while maintaining robustness against paraphrasing and translation attack.Our experimental findings validate the effectiveness of the proposed method, offering a pathway toward more secure and transparent AI deployment for Bangla-speaking populations.

Chapter 1

Introduction

Large Language Model (LLM) [8] watermarking is a technique used to embed identifiable and verifiable signals into the outputs of generative language models. The goal is to allow for the detection of whether a given piece of text was generated by a specific model[17], without noticeably altering the quality or meaning of the output. As LLMs become increasingly capable and widely deployed in various domains, watermarking has emerged as a crucial method to ensure content attribution, prevent misuse, and support accountability.

Despite rapid progress in the development of watermarking methods for high-resource languages such as English, low-resource languages like Bangla have received relatively little attention. This thesis attempts to addresses that gap by proposing a novel watermarking algorithm specifically designed for Bangla-language LLMs.

1.1 Introductory Information

As generative models grow more powerful, distinguishing between human-written and AI-generated text becomes increasingly difficult. In this context, watermarking serves as an invisible signature that can confirm whether text was produced by a specific model. These watermarks are designed to be imperceptible to humans but verifiable by machines, making them valuable tools in the effort to build trustworthy AI systems.

This thesis focuses on designing such watermarking algorithms specifically for **Bangla**, a language spoken by over **230 million** people across **Bangladesh** and **India**. Bangla is a low-resource language in the context of natural language processing **(NLP)**, meaning that digital tools and datasets for Bangla are limited compared to English or Mandarin. Furthermore, the language's rich morphology, flexible syntax, and compound structure pose unique challenges for LLM watermarking.

The proposed work addresses these linguistic characteristics while ensuring that the watermarks are secure, robust, and non-disruptive. With the growing application of AI in healthcare, education, media, and digital services in Bangla-speaking regions, this research contributes to safe and transparent AI deployment.

1.2 Motivations and Scope

This research is driven by both a technical challenge and a societal need. On one hand, there exists a clear gap in current literature: very few watermarking strategies address the structural and semantic nature of Bangla. On the other, there is increasing demand for methods that ensure the ethical use of generative AI, especially in regions where misinformation and digital forgery are rising concerns.

A review of recent studies shows promising advances in watermarking techniques for English, such as token selection biasing and syntactic alteration. However, these techniques are not directly transferable to Bangla due to linguistic differences and lack of training data. This research aims to design a solution that works within these limitations while still offering robustness and traceability.

Another key motivation is the protection of AI-driven services from model-stealing attacks that replicate expensive models through black-box access. For commercial and public deployments, especially in multilingual societies, there is a pressing need for watermarking techniques that are culturally and linguistically adaptive.

The scope of this thesis includes:

- How different algorithms perform against Bangla Large Language Models (LLMs) compare to that of English.
- The necessary adaption and requirements in algorithms to overcome weaknesses in bangla LLMS.
- Evaluation of watermark strength, robustness, and semantic preservation.

1.3 Problem Statement

There is currently no robust watermarking algorithm that effectively supports LLM outputs in Bangla. Existing methods either fail to consider Bangla-specific features such as agglutinative morphology, context-driven meaning, and sentence structure .So they can't sustain some kind of attacks(Round Trip Translation) ,where same algorithm can sustain it for other languages. This makes it difficult to implement reliable authorship tracking or content authentication for Bangla text generated by LLMs.

1.4 Research Challenges

Several challenges are inherent in designing watermarking methods for Bangla:

- **Linguistic Complexity:** Bangla's inflectional richness and flexible word order can obscure watermark signals and reduce detection reliability.
- **Low-Resource Ecosystem:** The limited availability of pre-trained Bangla LLMs, corpora, and evaluation tools constrains algorithm development and testing.
- **Semantic Preservation:** Embedding watermarks without affecting the fluency or intended meaning of Bangla sentences is significantly more difficult than in more rigid languages.
- **Resilience to Attacks:** Watermarked text must resist removal or distortion through paraphrasing, editing, or translation while remaining undetectable to humans.

This research aims to address these challenges through a carefully designed, language-aware watermarking approach.

1.5 Contribution

This thesis makes several novel contributions to the field of AI safety and multilingual NLP:

- Effectiveness of various algorithms and the scenarios in which they fail.
- A critical analysis of trade-offs between watermark visibility, strength, and semantic integrity in Bangla-language contexts.
- A Bangla-specific LLM watermarking algorithm, designed with morphological and syntactic properties of the language in mind.
- A new evaluation framework to measure watermark detectability, robustness, and text quality in Bangla.

1.6 Organization

The remainder of this thesis is organized as follows:

Chapter 2: Literature Review — A detailed analysis of prior work on LLM watermarking on different stratigies

Chapter 3: Methodology — Proposal and analysis of new watermarking mechanism, simulation and analysis of adversarial attacks and evaluation

Chapter 4: Experiments and Results — Evaluation setup, benchmark comparisons, robustness tests, and performance metrics.

Chapter 5: Discussion — Interpretation of results, insights into challenges and design trade-offs, and potential improvements.

Chapter 6: Conclusion and Future Work — Summary of findings, contributions, and avenues for further research in multilingual watermarking.

Chapter 2

Related Works

LLM watermarking has evolved from traditional digital watermarking in images [7] and audio to text-based methods adapted for natural language generation. Early work focused on discriminative models, embedding watermarks in classifier outputs or model weights. With the rise of generative models like GPT, techniques shifted to controlling token selection during decoding—pioneered by **Aaronson's statistical watermarking** and furthered by token-based methods such as **KGW** and **Christ Algorithms**. Semantic-based approaches like **SemaMark** introduced synonym and paraphrase-level embedding. The field advanced to model free watermarking (e.g., **Waterfall**) . Detection evolved through statistical testing, lexical frequency analysis, and token correlation methods, marking a move toward more robust, multilingual, and attack-resilient watermarking systems.

Expanded Taxonomy of LLM Watermarking

Dimension	**Category**	**Description**	**Algorithms**
Embedding Method	Semantic-Based	Preserves sentence meaning using synonym substitution or structure changes.	*SemaMark*, paraphrasing-based embedding
	Lexicographical-Based	Uses specific word choices or variants to signal watermark presence.	WordNet synonym substitution, US/UK spelling variants
	Token-Based	Biases token selection during generation to encode patterns.	*KGW family*, *Christ family*, *DERMARK*, *Waterfall*

Dimension	Category	Description	Examples / Algorithms
Watermark Strength	Zero-Bit	Embeds no explicit message; detects if text is AI-generated.	*Aaronson's statistical watermarking*, *Christ-Zero*
	Multi-Bit	Embeds specific binary data (e.g., model ID or timestamp).	*DERMARK*, *SynthID*, *CredID*, *Airdisc*, *LLaMA-Mark*
Evaluation Techniques	Statistical Testing	Analyzes token frequency patterns using tests like binomial or chi-square.	Binomial test (Aaronson), p-value analysis in *Waterfall*
	Lexical Frequency Analysis	Measures distribution shift in word or synonym usage.	Word-level frequency histograms, KGW green token ratio
	Token Sequence Correlation	Matches token sequences to expected patterns using seeded randomness.	Hash-based alignment in *Christ*, *KGW*, *MarkLLM*

2.1 Protecting IP with Lexical Watermark

This research addresses a critical issue in the era of cloud-based Natural Language Generation (NLG) APIs: protecting intellectual property (IP) from model extraction attacks using the proposed watermarking method.[13]

Problem Statement:

Model Extraction Attacks (MEA)[13]

- **What is MEA?** Attackers interact with a proprietary NLG API (e.g., Google Translate, ChatGPT) to collect input-output pairs. They then train a surrogate model (imitator) to replicate the API's functionality, bypassing subscription fees or reselling the stolen model.
- **Why is it harmful?** Companies invest heavily in training high-performance NLG models. MEA undermines their revenue and competitive advantage.

Proposed Solution: Lexical Watermarking

Given a text generation model $f(x)$ producing output y, a **watermarking module** applies:

$$y^{(m)} = \begin{cases} m(y) & \text{if } t(y) \text{ is True} \\ y & \text{otherwise} \end{cases} \tag{2.1}$$

- **Trigger Function $t(y)$:** Identifies candidate words for watermarking.
- **Modification Function $m(y)$:** Replaces selected words based on a **confidential substitution rule**.

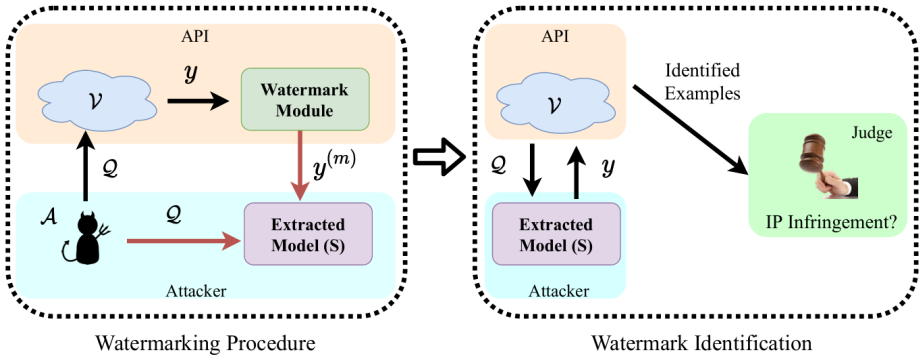

Figure 2.1: Illustration of the watermarking and identification process. During generation, responses y to queries $\mathcal{Q}$ are watermarked. For verification, the victim $\mathcal{V}$ queries the suspect model and examines the output y to confirm watermark presence.

How the modification function works?

Synonym Replacement

- **Targets adjectives** (for stability).
- Uses **WordNet[23]** to select least frequent synonyms.
- **Steps:**
  1. Rank adjectives by frequency.
  2. For each word, pick the last M synonyms.
  3. Replace words via a **hash function** H.

Spelling Variant Replacement

- Switches between **US and UK spellings** (e.g., "color" vs. "colour").
- Uses a hash function to enforce consistent replacements.

IP Infringement Detection:

Hit Ratio Calculation

A **hit ratio** measures watermark prevalence in a suspect model's output:

$$\text{hit} = \frac{\#(W_y)}{\#(C_y \cup R_y)} \tag{2.2}$$

- $\#(W_y)$: Number of watermarked words.
- $\#(C_y \cup R_y)$: Total candidate & substitute words.

If hit $> \tau$ (threshold), the model is flagged as suspicious.

Statistical Hypothesis Testing:

A **binomial test** evaluates if observed watermarks occur by chance:

- **Null Hypothesis (H_0):** The model selects words randomly ($p = \frac{1}{M+1}$).
- **Test Statistic:**

$$P = 2 \cdot \min\left(\sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i},\ \sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-i}\right) \tag{2.3}$$

- **Decision Rule:** If $P < \alpha$ (significance level), reject H_0, confirming IP theft.

Analysis of Supporting and Contradictory Findings Supporting Evidence
The use of synonym replacement aligns with Venugopal, Chen, Vogel, *et al.* [34], who demonstrated that lexical substitutions can embed detectable patterns without semantic distortion.

Contradictory Findings
Unlike Zhang, Gu, Jang, *et al.* [39] who argue that neural watermarking (e.g., weight perturbation) is more robust, this work shows that rule-based lexical changes can be equally effective for IP tracing while being computationally lighter.[42]

Recurring Themes and Research Gaps

Key Themes

- **Imperceptibility vs. Detectability Trade-off:** Most works [19] struggle to balance stealth and robustness. This work mitigates this by restricting substitutions to low-frequency synonyms and spelling variants.
- **Adversarial Robustness:** While Szyller, Atli, Marchal, *et al.* [31] focus on adversarial removal attacks, this method's hash-based consistency makes it resistant to simple paraphrasing.

Identified Research Gaps

- No discussion on multilingual watermarking **(most methods focus exclusively on English)**
- Limited evaluation on large language models (LLMs), where lexical distributions differ significantly

Critical Evaluation

Strengths

- **Low-Cost Deployment:** Unlike neural backdoors, no model retraining is needed [**christ2023**].
- **Statistically Verifiable:** Uses binomial hypothesis testing for forensic evidence.

Limitations

- **Vocabulary-Dependent:** Relies on predefined candidate words (e.g., adjectives), limiting scalability [42].
- **Hash Collision Risks:** Potential vulnerability if attackers reverse-engineer the hash function H.
- **Text-Only Limitation:** Doesn't extend to code/structured data unlike multimodal approaches.

2.2 A KGW Family Watermark for Large Language Models - The KGW Algorithm [18]

This paper stands out as one of the most cited and foundational works in the field.It proposes a model agnostic(independent) framework for watermarking LLMs that allows the generated text to be detected using statistical measures, without impacting its quality or requiring access to the model parameters or weights.

LLMs like ChatGPT can generate text that is indistinguishable from human writing[30] . They are powerful yet present certain risks - **misinformation spread [2]** through bots or fake news [24],**academic dishonesty** via AI-written assigmnents, creating **biased** learning datasets, etc. Thus the necessity to detect and prevent it. Traditional methods like classifiers often require access to the model, are vulnerable to adversarial attacks, or inaccurate.So a model free approach is necessary.

The authors propose a model agnostic variant of watermarking algorithm which also has negligible impact on text quality - **The Soft Token Bias Statistical Watermark**.

The **entropy measure** of a sentence plays an important role both in **determining text quality** and the **watermarking** prospect. Suppose a text is given **"A quick brown fox jumps over the lazy dog"**.To say whether this piece of text is human written or AI generated is fundamentally difficult. These types of familiar sentences, idioms, proverbs, etc have **low entropy** and thus very low flexibility.Therefore watermarking such sequences can degrade the text quality significantly and shoot up perplexity.On the otherhand, texts like "The weather is cloudy" has a very **high entropy** thus offering more flexibility for changing text. We can watermark the sequences with high entropy and can maintain text quality.But the low entropy sentences pose a significant challenge.

Hard Watermarking Approach (Red List Rule)

In the hard watermarking approach, the model is **forbidden from selecting** certain tokens during text generation.At each step in generation, a random subset of the vocabulary is designated as the "green list" (preferred tokens) and the rest as the "red list" (forbidden tokens).

The model is then **only allowed to choose the next word from the green list**.If the **most likely** token (based on the model's usual behavior) is in the **red list**, it **must be skipped** even if that affects fluency or naturalness.

This creates a strong, easily detectable pattern in the output because the distribution of tokens is sharply skewed.However, this can negatively affect text quality, especially when word choice is limited or constrained (like code or low-entropy text).

Detection Process for Hard Watermarking

Since detection process is model agnostic, there is no need for model API or weights and parameters.We can determine the watermark presence by calculating frequency of green list tokens since in hard watermarking there will be no red list tokens used.All we need is the hash function and RNG.

Natural human text has **high perplexity, lower frequency of green tokens.** So it is expected human text will violate the red list rule in about half of the tokens.No violation in watermarked text in this approach.

The null hypothesis for the statistical test is

$$H_0: \quad \text{The text sequence is generated with no knowledge of the red list rule.} \tag{2.4}$$

The z statistic for this test is

$$z = \frac{2\left(|s|G - \frac{T}{2}\right)}{\sqrt{T}} \tag{2.5}$$

In general, **removing the watermark** of a long sequence requires **modifying roughly one quarter** of the tokens or more.Even then there is only a 50% chance that the token is flipped from a green list token to a red one.If the null hypothesis is true, then the number of green list tokens, denoted $|s|_G$, has expected value $\frac{T}{2}$ and variance $\frac{T}{4}$ where T is no of tokens

Soft Watermarking Approach (Green List Bias)

In contrast, the soft watermarking method uses a more practical strategy- Like before, a green list is generated randomly for each token step.But this time instead of restricting the model, the green list tokens are slightly favored during sampling by increasing their likelihood by adding a bias value to them.The model still has access to the full vocabulary—it can choose any token—but it is more likely to pick from the green list.

This method keeps the watermark statistically detectable, but has a much smaller impact on text quality, making the generated text more natural.It's also more **robust in low-entropy** situations and ensures quality.

In this approach, A Large Language Model (LLM) **L** takes an input prompt of length N_p, Green list size $\gamma \in (0,1)$ and Hardness parameter, $\delta > 0$.

Based on the known sequence of tokens $\left(s^{(-N_p)}, \cdots, s^{(-1)}\right)$ in prompt

1. It calculates logits vector $\boldsymbol{l}^{(t)}$ over the vocabulary $\boldsymbol{V}$.

2. Computes a **hash** of token $s^{(t-1)}$, which is used as a **seed** to a **Random Number Generator**(RNG).
3. Using this RNG, the vocabulary is randomly split into **two** lists
   - a *green* list **G** of size $\gamma|V|$
   - a *red* list **R** of size $(1-\gamma)|V|$
4. It **adds a bias** factor δ to the **each green list logits** and then applies softmax to convert to probability distribution over the entire vocabulary
5. Now we sample next token based on this probability

A better understanding of the probability distribution can be obtained from the equations for the token generation -

$$\hat{p}_k^{(t)} = \begin{cases} \dfrac{\exp(l_k^{(t)} + \delta)}{\sum_{i \in R} \exp(l_i^{(t)}) + \sum_{i \in G} \exp(l_i^{(t)} + \delta)}, & k \in G, \\ \dfrac{\exp(l_k^{(t)})}{\sum_{i \in R} \exp(l_i^{(t)}) + \sum_{i \in G} \exp(l_i^{(t)} + \delta)}, & k \in R. \end{cases} \tag{2.6}$$

So we see at any time step t for the production of the logit of token k,

- if token belongs to the green list then the bias factor δ is added to the logit and then softmax is applied so that the probability is boosted and the model showcases a tendency to output tokens from the green list.
- no bias added for red list tokens thus discouraging (not necessarily preventing) token selection from red list.

Selection modes for High Entropy situation

1. **Greedy sampling**: Use the best choice token - the one with highest probability
2. **Random Sampling**: Select a random token from the list of possible values
3. **Beam Search(Top k)**: Select the top k tokens ranked by probability and use any one from them.

How it improves text quality

The algorithm causes the model to watermark high entropy region or sequences rigorously while avoiding regions of low entropy.So overall, the effects of the watermark remains while text quality is also preserved.

How it maintains Perplexity

For the high entropy situation tokens, in soft watermarking, the randomness of the green list causes tokens to be sampled uniformly, and the perplexity is unaffected.

Conversely for minimal entropy situation, the algorithm has no effect

The following figure illustrates the overall process of the algorithm.

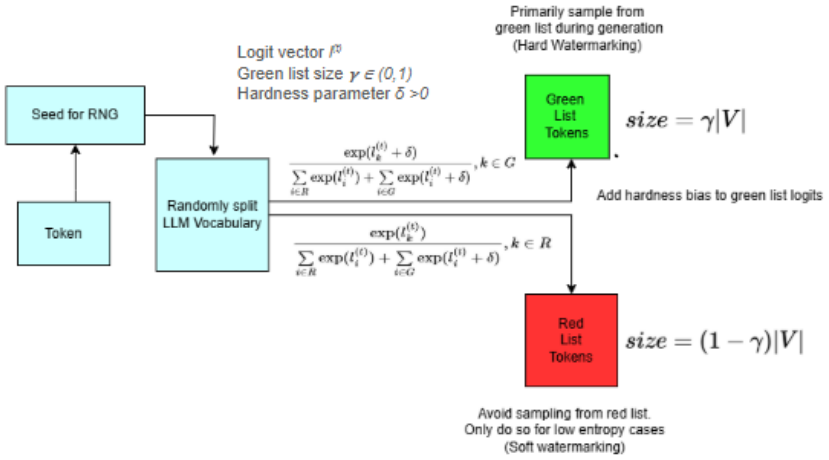

Figure 2.2: Process of Vocabulary splitting in Soft watermarking

Assumptions

For checking sensitivity and illustration purpose $\gamma = 0.5$, tokens generated = 200 from **C4** dataset's **RealNewsLike** [27] subset. **The z value = 4.0.**

Detection Results We used the following statistic for the soft watermarking case

$$z = \frac{|s|_G - \gamma T}{\sqrt{T\gamma(1-\gamma)}} \tag{2.7}$$

- 98.4% of AI-generated texts were successfully detected using the soft watermark with **multinomial sampling.**
- With **beam search**, detection accuracy rises to 99.6
- When focusing only on high-entropy texts (more creative/unpredictable), the detection works 100% of the time.

skip repeated n-grams. for better result

Private Watermarking

A more secure version of watermarking where the Red and Green lists are **not generated publicly**.Instead a secret random key and a **Pseudorandom Function**(PRF) is

used for generation.Only the model owner or someone who knows the key can detect it.Furthermore they can increase window length h to further solidify the attack.

This makes it much harder for adversarial attacks as the malicious user does not know the secret key, thus cannot determine green or red list tokens. Brute force attack is also out of the question, especially if window size h(no of past tokens considered) changes.

Analysis of Various Attacks

Text Insertion, Deletion and Substitution

Attacker may insert additional words or remove them or substitute with another to alter watermark.But there is only 50% chance of it belonging to red list.Also text quality can be significantly altered.This algorithm is highly resistant against small or moderate alterations but heavy changes can cause it to fail.

Specific Attack Strategies:

Paraphrasing Attacks:

The Achilles' Heel to our algorithm.Manually or automatically rephrasing the text using token based edits for paraphrasing hurts watermark strength significantly.

Discreet Alterations:

Adding spaces, typos, or slight spelling errors.Good watermarking design can defend against this by normalizing input text before detection.

Tokenization Attacks:

Altering characters to change how text is tokenized (e.g., modifying invisible characters like newline). Can cause unexpected token splits that disrupt watermarking.

Homoglyph and Zero-Width Attacks [10]:

This attack substitutes characters with visually similar Unicode versions or adding invisible Unicode characters. Defended by normalizing or canonicalization[15] and preprocessing

Generative Attacks:

Prompting the model to follow weird algorithm (e.g., inserting emojis[11] after every word). They are by far the strongest attacks against watermarks..Now it is solved by Reinforcement Learning[6] using Human Feedback[25]

Strengths

- **Model-Agnostic Detection:** The watermark can be detected without access to the underlying language model thus enabling broad applicability.

- **Statistical Rigor:** Detection is based on a formal statistical measures like z-test with interpretable p-values, allowing reliable decisions
- **Imperceptibility:** The watermark introduces only a soft bias, preserving the quality and natural feel
- **Robustness:** The watermark degrades gracefully under replacement, and partial deletions, maintaining detectability.
- **No Retraining Required:** The approach can be applied to any model that samples from a next-token distribution, without retraining.
- **Flexibility:** Detection is independent of enforcement strength (δ), allowing the detector cross model capabilities
- **Security via Private Watermarking:** Extensions with secret keys and pseudorandom functions provide robustness against brute-force and reverse-engineering attacks.

Weaknesses

- **Susceptible to Advanced Attacks:** Strong generative or paraphrasing attacks using powerful models can reduce watermark strength while maintaining text quality.
- **Limited Performance on Low-Entropy Text:** The watermark is harder to embed and detect in deterministic or repetitive outputs, such as code or factual snippets.
- **Reliance on Preprocessing:** Without proper normalization (e.g., canonicalization[26] [3]), attacks like homoglyph or tokenization manipulation can hinder detection.
- **Parameter Sensitivity:** Detection effectiveness depends on correctly chosen parameters such as the green list size γ and threshold z.
- **Real-World Uncertainty:** The method is theoretically sound but needs further validation

So although this algorithm does very well to remove model dependency and maintain quality and entropy it still fails miserably against various attacks like paraphrasing, substitution, round trip translation . So now we move onto EXP algorithm.

2.3 Robust Distortion-free Watermark Algorithm : Exponential Sampling

The KGW[18] Algorithm mentioned previously shows vulnerability to some types of attack(i.e, paraphrasing, substitutions). This particular research introduces some methods to overcome those weaknesses. We analyze and compare various approaches, focusing on their theoretical foundations, practical implementations, and limitations. The fundamental parts we extracted from this paper are distortion-free watermarks, robustness to attacks, and the trade-offs between detectability, computational complexity, and text quality preservation. We critically examine current methodologies with Bangla LLMs and identify promising directions for future research .

The proliferation of large language models has created an urgent need for reliable attribution methods. Watermarking techniques have emerged as a promising solution, with three key desiderata:[20]

- **Distortion-free**: Preserving the original text distribution, formalized as:

$$P(\text{text}) = \int_{\xi} \mathbf{1}\{\text{text} = \text{generate}(\xi, \text{prompt})\}\, d\nu(\xi) = \text{original LM distribution} \quad (2.8)$$

- **Agnostic**: Detectable without the language model or prompt, requiring a test statistic ϕ such that:

$$\phi(\text{generate}(\xi, \text{prompt}), \xi) \ll \phi(\text{text}, \xi) \text{ for independent text} \quad (2.9)$$

- **Robust**: Withstanding text perturbations, including edits, substitutions (40-50% in some methods), and paraphrasing attacks

Two principal sampling schemes exist:

Inverse Transform Sampling (ITS)

$$\Gamma(\xi, \mu) = \pi^{-1}(\min\{\pi(i) : \mu(\{j : \pi(j) \leq \pi(i)\}) \geq u\}) \quad (2.10)$$

$$d(y, \xi) = \sum |u_i - \eta(\pi_i(y_i))| \quad (2.11)$$

Demonstrates robustness against 40-50% random edits ($p \leq 0.01$ at 35 tokens). [1]

[1] As shown in Table **??**, the symbols are formally defined.

Exponential Minimum Sampling (EXP)

$$\Gamma(\xi, \mu) = \arg\min_i -\log(\xi_i)/\mu(i) \tag{2.12}$$

$$d(y, \xi) = \sum \log(1 - \xi_{i,y_i}) \tag{2.13}$$

Outperforms inverse transform empirically, requiring 2-3 × fewer tokens for comparable detection power. So we focused more on this .A complete flow diagram in Figure2.3 will make it easier to understand .

Go to appendix table A.2 to see the details of each notations

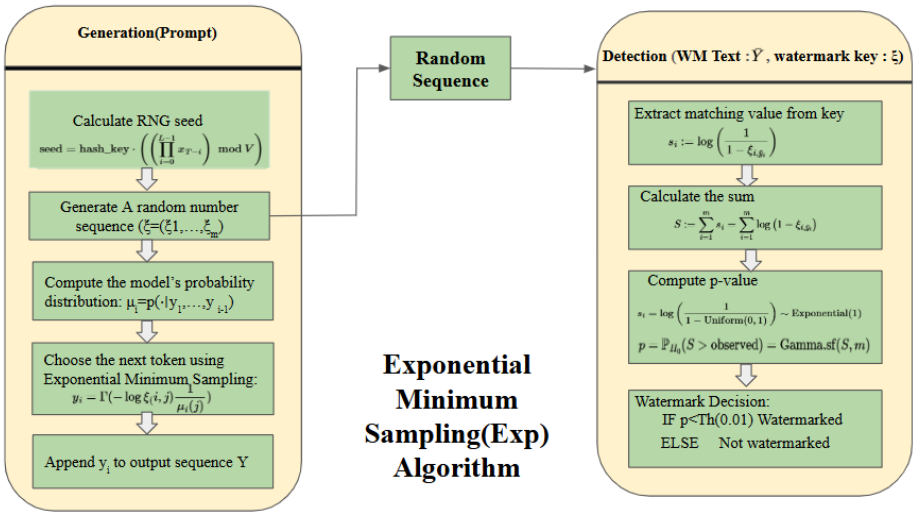

Figure 2.3: Illustration of Exponential minimum Sampling(EXP) Algorithm

Three key trade-offs emerge: (1) **Distortion vs. robustness**: Hashing methods explicitly trade off window size k, while alignment methods incur $O(mnk^2)$ detection complexity. (2) **Instruction-tuning gap**: Models like Alpaca-7B exhibit lower watermark potential ($\alpha \approx 0.28$) compared to LLaMA-7B ($\alpha \approx 0.59$), inherently limiting detection regardless of method.

Current methods face four major limitations. First, **theoretical limits** preclude achieving sublinear detection complexity, constant robustness to ϵ edits, and exact distortion-freeness simultaneously. Second, in **adversarial settings**, no method resists attacks that exploit knowledge of the key distribution or alignment algorithm. Third, **low-entropy text** (e.g., instruction-following outputs) suffers from the fundamental bound $\Pr(\text{detect}) \leq \exp(-cm\alpha(y))$. Fourth, the **model release problem** remains unsolved—all watermarks fail under white-box access.

Promising avenues include (1) **composite watermarks** combining alignment robustness with hashing efficiency, though preserving distortion-freeness poses theoretical chal-

lenges; (2) **learned watermarks** that recognize patterns but risk losing guarantees; (3) **entropy-adaptive methods** dynamically allocating resources based on $\alpha(y)$; and (4) **cryptographic foundations** for formal security definitions against adaptive adversaries, potentially borrowing from steganography.

While current techniques show promise, fundamental challenges persist in computational complexity, low-entropy settings, and adversarial robustness. The ultimate goal remains a watermark satisfying $\forall \epsilon > 0, \exists \text{method} : \text{Complexity}(m) = o(m^2) \wedge \text{Robustness}(\epsilon) = \Omega(1) \wedge D_{\text{TV}}(P_m, P_0) = 0$, where P_m and P_0 denote watermarked and original distributions, respectively

2.4 SEMSTAMP: A Semantic Watermark with Paraphrastic Robustness forText Generation

We saw previously in the KGW algorithm that although it handles the model agnostic aspect and text quality, this token level[20] algorithm has an **Achilles' Heel - Paraphrastic Substitutions**.To address this issue the ***SEMSTAMP*** paper introduces a robust, **proactive** algorithm (watermark during generation) which operates on the semantic representation of sentences.They encode entire sentences in the vector representation and then modify LLM output continuously until the watermark can be significantly injected in output. A new **Bigram Paraphrase Attack** is introduced and used to evaluate robustness[29] of the SEMSTAMP algorithm.Turns out it can do one better than KGW - robust to paraphrasing and ensure natural text too.

As LLM advances rapidly and gradually producing realistic human like text, chances of its misuse becomes widespread. Token level watermarking[37] showed great results but paraphrasing remained a weakness.Paraphrasing causes substitution of tokens and watermark, text quality[9] is significantly weakened.So to tackle this the authors proposed an algorithm which works in the semantic representation of entire sentences. The key idea is - **while paraphrasing alters the surface-form tokens, the sentence-level semantics are unchanged.**With this in view, they proposes the SEMSTAMP algorithm and the Bigram Paraphrase Attack to evaluate algorithm robustness. They train a paraphrastic robust sentence encoder with contrastive learning[35] to enhance algorithm robustness also.

How SEMSTAMP improves on KGW Algorithm

Since the token level design is the root of the problem, they opt for a different approach. They utilize **Locality Sensitive Hashing[16] (LSH)** to partition embedding space.LSH[35] is a technique which maps similar items together to similar signatures

which reduces dimensionality and allows provides similarity measure for higher dimension, which is what they emulate for the sentences.

Mechanism:

Given an embedding vector $v \in \mathbb{R}^h$, Locality Sensitive Hashing (LSH) projects v onto d random hyperplanes (each defined by a normal vector $n^{(i)}$).

The hash signature is a d-bit binary code, where each bit is defined as:

$$\mathrm{LSH}_i(v) = 1\left(n^{(i)} \cdot v > 0\right)$$

Similar sentences will have similar binary signatures (e.g., "The cat sat" and "A feline sat down" may hash to the same region).All of them are based on cosine similarity measurement metric to bring similar sentences closer to each other

Algorithm:
Input: Model M, prompt P, and number of sentences to generate T.

1. For $i = 1, 2, \ldots, T$ do:
   (a) Randomly initialize d vectors $\in \mathbb{R}^h$ so that there are 2^d subspaces for binary representation.
   (b) Hash the most recent sentence (initially the prompt), scramble the hash (multiply by a large arbitrary number), and use it to seed a random number generator (similar to KGW).
   (c) Divide the subspace $\{1, 0\}^d$ into a **valid region** $G(t)$ of size $\gamma \cdot 2^d$ and a **blocked region** $R(t)$ of size $(1 - \gamma) \cdot 2^d$.
   (d) Now generate a sentence and calculate its signature by checking which side of each normal vector it resides in.
   (e) Check if the full binary signature corresponds to a valid region and the marginal constraint (distance from boundary) is satisfied.
   (f) If yes, accept
   (g) Else reject it and sample another sentence.Keep doing it until valid region found.This is called **rejection sampling.**

The margin constraint is calculated as the difference between the normal vector and the candidate sentence embeddings.

Since the meaning of the text remains the same even after paraphrasing, it is assumed the LSH hash will also remain the same.Therefore it will maintain the watermark and

remain detectable even after attack.

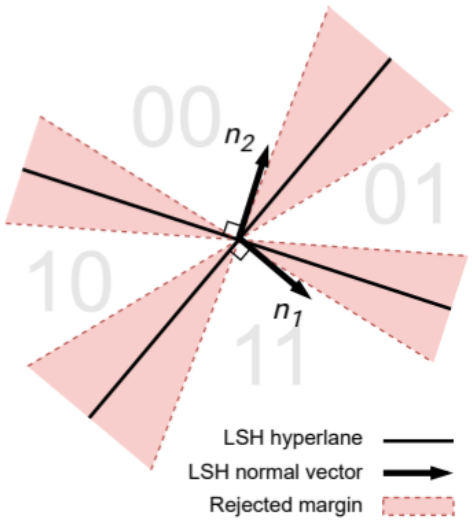

Figure 2.4: An illustration for margin-based rejection. Sentence embeddings at LSH hyperplane boundaries are rejected (highlighted in red).

Detection

The watermark is detected using the count of sentences falling in valid regions of the semantic subspace. If this count is too high then it is watermarked text.Statistical z test can be done to check.

Core Components of SEMSTAMP

1. **Paraphrase-Robust Sentence Encoder:** The sentence encoder is the **backbone of SemStamp**—it converts sentences into numerical vectors (embeddings) that capture their meaning rather than just word patterns. This ensures the **watermark survives paraphrasing**.

   The encoder for *SEMSTAMP* is based on **Sentence-BERT (SBERT)**[**28**] which is already fine tuned on semantic tasks - **STS Benchmark**[**4**]. It uses a **Siamese Network** (twin neural networks) for efficient sentence comparison.

   However SBERT is still not ready for paraphrasing . It is fine-tuned further using **contrastive loss -** bring paraphrases much closer in the embedding space over others.

   How it learns this:

   - Generate a paraphrase t_i using **PEGASUS**[**40**] or similar model
   - Pick a random non-paraphrase t_i'
   - Compute Similarities - **M** is the model

(a) $f_\theta(s_i, t_i) = \cos\left(\mathbf{M}_\theta(s_i), \mathbf{M}_\theta(t_i)\right)$

(b) $f_\theta(s_i, t_i') = \cos\left(\mathbf{M}_\theta(s_i), \mathbf{M}_\theta(t_i')\right)$

- We use the following loss function:

$$\min_\theta \sum_i \max\left(\delta - f_\theta(s_i, t_i) + f_\theta(s_i, t_i'),\ 0\right) \tag{2.14}$$

- Penalize if the differences of cosine similarities is greater than δ.This brings them closer.

2. **Semantic Space Partitioning through LSH**

   Key Steps:

   (a) **Initialize LSH Hyperplanes**
   Sample d random normal vectors $(n^{(1)}, \ldots, n^{(d)})$ from a Gaussian distribution.

   These vectors define d hyperplanes that split the semantic space $\mathbb{R}^h$ into 2^d regions.

   Each region is identified by a d-bit binary signature (e.g., 011).

   (b) **Pseudorandom Partitioning for Watermarking**
   For each new sentence $s^{(t)}$, the valid regions $G^{(t)}$ are chosen based on the previous sentence's LSH signature $\text{SIG}(s^{(t-1)})$:

   - Convert $\text{SIG}(s^{(t-1)})$ from binary to decimal
   - Multiply by a large prime for Scrambling p to seed a randomizer.
   - Use this seed to split the 2^d regions into:
     - Valid regions $(G^{(t)})$: Size $\gamma \cdot 2^d$.
     - Blocked regions $(R^{(t)})$: The rest $(1-\gamma) \cdot 2^d$.

   (c) **Rejection Sampling for Watermarked Sentences**
   The language model generates sentences until one's embedding falls into a valid region:

   - Compute its LSH signature: $\text{LSH}(M_{\text{embd}}(s^{(t)}))$.
   - Check if the signature $\in G^{(t)}$.
   - If not, try again.

3. **Margin-Based Constraint for Robust Watermarking**:

   We use this equation for this

$$\min_{i=1,\ldots,d} \left|\cos\left(n^{(i)}, v_t\right)\right| > m \tag{2.15}$$

We calculate the dot product of all the d normal vectors generated and calculate the minimum absolute value which corresponds to the riskiest or most ulnerable boundary line.We check if the distance from boundary to the sentence embedding is above the threshold δ.If yes we accept or we reject .

The Bigram Paraphrase Attack The bigram paraphrase attack is a rigorous form of token substitution attack specifically aimed at destroying token level watermarking algorithms like KGW.As KGW uses the most recent/last token $s^{(i-1)}$ to determine next token **green list** thus changing **even one word in a bigram** will hamper the watermark as chain effect.

So we use this method to design the bigram attack -

$$s' = \underset{x \in \{s'_1, \ldots, s'_k\}}{\operatorname{argmin}} B(x, s) \quad \text{subject to} \quad S(s'_1, s) - S(x, s) \leq \Delta \cdot S(s'_1, s) \tag{2.16}$$

SO basically out of many candidate paraphrases select the one with -

- **fewest overlapping bigrams** with the original (B(x, s) = bigram overlap count).
- But **constrain quality**. Ensure BERTScore [41] (S(x, s)) doesn't drop below a tolerance Δ of 10%

Datasets

Tested on **RealNews (C4)**, **BookSum**, and **Reddit-TIFU** (formal + informal text).

Results of experiment

They used a Bigram Paraphrase candidate count of 10 and set the tolerance for dropping score Δ to 10%. SEMSTAMP is more robust to paraphrase attacks than KGW across the Pegasus, Parrot, and GPT-3.5-Turbo paraphrasers, as measured by AUC, TP@1%, and TP@5%. The bigram paraphrase attack significantly weakens the token-level KGW algorithm, while SEMSTAMP remains relatively unaffected. Moreover, while KGW notably degrades perplexity due to token-level noise, SEMSTAMP maintains perplexity comparable to the base model and also preserves both token and semantic diversity, as measured by the Ent-3 and Sem-Ent metrics.

Limitations

- **Vulnerability to Inter-Sentence Attacks:** SEMSTAMP is less effective against attacks such as sentence reordering and embedding machine text in large human text corpora.
- **Generation Speed Overhead:** Combination of LSH hyperplane generation, signature and Rejection Sampling causes algorithm to be slow.
- **Risk of Reverse Engineering:** Without stronger protection, eventually one can reverse engineer the algorithm components
- **Attack Control Limitations:** The effectiveness of the bigram paraphrase attack depends on how strictly BERTScore constraint is enforced; relaxing constraints increases attack success.

So we see SEMSTAMP offers a novel sentence-level semantic watermark that effectively resists paraphrase attacks.But other types of Cross Lingual Attacks can harm the algorithm.Also the speed is another drawback.So a better approach is to use an algorithm which will maintain the distribution throughout.

2.5 CWRA and X-SIR

The paper explores the problem of cross-lingual consistency in watermarking texts generated by large language models,and proposes a new kind of attack called cross watermark removal attack (**CWRA**[14]) and introduces **X-SIR**[14], a defense mechanism that enhances watermark robustness against translation by utilizing semantic clustering and cross-lingual vocabulary partitioning

Large language models (LLMs), such as GPT-4[1], have achieved remarkable advancements in generating human-like text. While these capabilities offer many benefits, they also raise serious concerns regarding potential misuse. Examples include the generation of misleading information, impersonation, and violations of academic integrity. In response to these risks, researchers have developed text watermarking techniques—methods that embed imperceptible signals into generated content to indicate its origin. These watermarks are invisible to human readers but can be algorithmically detected, enabling systems to trace and verify whether a text was produced by an LLM.

A key requirement for these watermarking methods is robustness—the ability to reliably detect watermarks even after the content has been altered. Current methods have demonstrated strong robustness against common manipulations such as paraphrasing and copy-paste attacks. However, most evaluations of these techniques have been limited to monolingual contexts, where both the input and output remain in the same language.

In real-world applications, particularly in global communication contexts, texts generated

by large language models are frequently translated into other languages. This raises a critical challenge: whether the watermark embedded in the original text remains detectable after translation. The concern becomes more pressing in adversarial scenarios—for instance, a user might generate deceptive content in English using a watermarked LLM, translate it into Chinese, and disseminate it to avoid detection. While the harmful intent of the content remains unchanged, it is uncertain whether the original watermark survives the translation process.

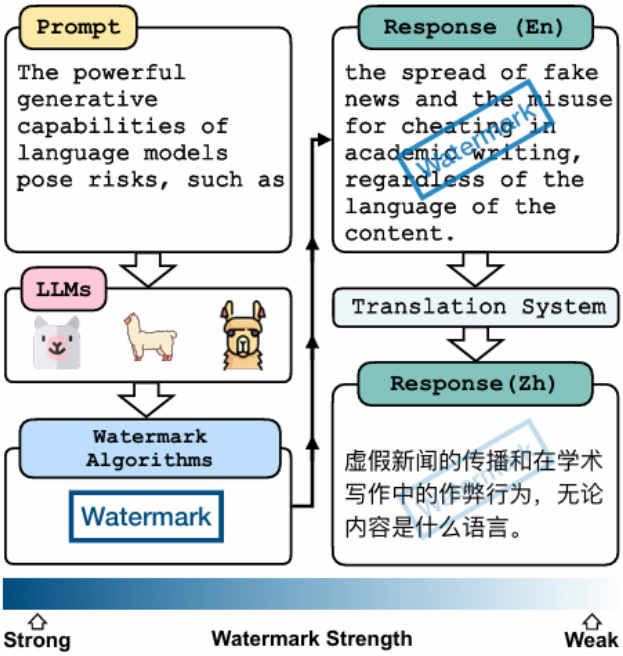

Figure 2.5: Illustration of watermark dilution in cross lingual environment

CWRA:

A new type of attack called **Cross-lingual Watermark Removal Attack (CWRA)**, which demonstrates the vulnerability of current watermarking methods in multilingual settings. When performing CWRA,the attacker begins by translating the original language prompt into a pivot language, which is fed to the LLM to generate a response in the pivot language. Finally, the response is translated back into the original language. In this way, the attacker obtains the response in the original language and bypasses the watermark with the second translation step

Figure 2.6: An example pipeline of CWRA with English (En) as the original language and Chinese (Zh) as the pivot language. When performing CWRA, the attacker not only wants to remove the watermark, but also gets a response in the original language with high quality. Its core idea is to wrap the query to the LLM into the pivot language

In their work, they propose a defense strategy designed to improve the cross-lingual consistency of current LLM watermarking techniques in the face of **CWRA**. Their approach is based on two key principles.

Cross-Lingual Semantic Clustering of the vocabulary. Unlike the KGW method [18], which treats each token in the vocabulary as an individual unit when embedding watermarks, their method groups tokens that share similar semantic meanings across different languages. This allows them to treat clusters of semantically related tokens as the smallest unit for watermarking. Consequently, after translation, the post-translated token retains the watermark, as it still belongs to the same semantic cluster.

Cross-Lingual Semantic Robust Vocabulary Partition Drawing inspiration from Liu [22], they ensure that the partitioning of the vocabulary remains consistent for semantically similar contexts across languages. This improves the watermark's robustness against translation-based attacks

In their work, authors introduce the Semantic Invariant Robust watermark (SIR), which demonstrates robustness against re-translation and paraphrasing attacks. The core idea behind SIR is to assign similar watermark values, denoted as Δ, for semantically similar prefixes. Given two prefix sequences, x and y, SIR utilizes an embedding model E to capture their semantic similarity. It then trains a watermark model that outputs the watermark values Δ with the primary objective of minimizing the difference in semantic similarity between the original sequences and their watermarked versions.

The objective function is defined as:

$$L = |\text{Sim}(E(x), E(y)) - \text{Sim}(\Delta(x), \Delta(y))| \tag{2.17}$$

where $\text{Sim}(\cdot, \cdot)$ denotes the similarity function. Additionally, for each token $i \in \{1, 2, \ldots, |V|\}$,

the watermark value Δ_i is trained to be close to either $+1$ or -1. Consequently, SIR can be viewed as an enhancement of the KGW[18] method, where $\Delta_i > 0$ indicates that token v_i is a "green" token.

The original implementation of SIR utilizes C-BERT[5], which is limited to English. To extend SIR for cross-lingual scenarios, a multilingual version of S-BERT [28] is employed.

The models used in the study include **BAICHUAN-7B**[36], a bilingual LLM trained on 1.2 trillion tokens, with official support for both Chinese and English, and additional coverage for Japanese tokens. Additionally, **LLAMA-2-7B** [32] was utilized, which is trained on 2 trillion tokens and officially supports only English, though its vocabulary also includes tokens for several European languages, such as German and French.

The authors basically tried to answer three questions related to cross lingual consistency in text watermarking in their research:

RQ1:To what degree are current watermarking algorithms consistent across various languages?
RQ2:Do watermarks show greater consistency between closely related languages compared to more distantly related languages?
RQ3:Does the Semantic Invariant Watermark (SIR) demonstrate superior cross-lingual consistency compared to other methods such as KGW[18] and UW?

The main results are presented, with the model being instructed to generate 200 tokens in response, following the setup outlined by Kirchenbauer [18]. When the response length restriction was lifted, the trend of watermark strengths in relation to text length was illustrated.

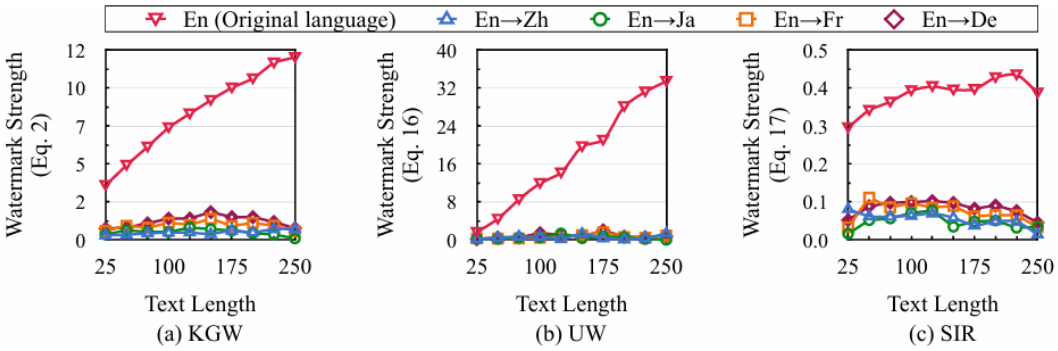

Figure 2.7: Trends of watermark strengths with text length before and after translation.These are the average results of BAICHUAN-7B[36] and LLAMA-2-7B

The visual analysis reveals that the watermark strengths of all three methods significantly decline after translation. These findings indicate that current watermarking algorithms

face challenges in maintaining their effectiveness across different languages.None of the three watermarking methods demonstrate a clear advantage in cross-lingual consistency between similar languages compared to distant ones. This suggests that even when two languages share structural similarities or vocabulary, watermark transfer remains difficult, highlighting a major challenge for achieving consistent watermarking across languages.Overall, SIR demonstrates superior cross-lingual consistency compared to both KGW[18] and UW, achieving the highest average performance across the two models and evaluation metrics. Specifically, when using BAICHUAN-7B[36], SIR significantly outperforms the other methods in terms of Pearson correlation coefficients (PCCs) across all target languages. This result underscores the importance of semantic invariance in maintaining watermark strength across languages. Nevertheless, despite its advantages, SIR still experiences a noticeable decline in watermark strength in cross-lingual settings.

About the defence of CWRA,the author proposes an imporoved version of SIR and named it as X-SIR .To enhance SIR's cross-lingual consistency, the authors modify it to assign watermark biases not to individual tokens, but to clusters of semantically similar tokens. This ensures that translations—tokens with similar meanings across languages—receive the same bias, helping the watermark survive translation. These clusters are built using a bilingual dictionary, treating each token as a node and connecting tokens that are translations. The resulting connected components form semantic clusters, which make watermarking more robust across languages.

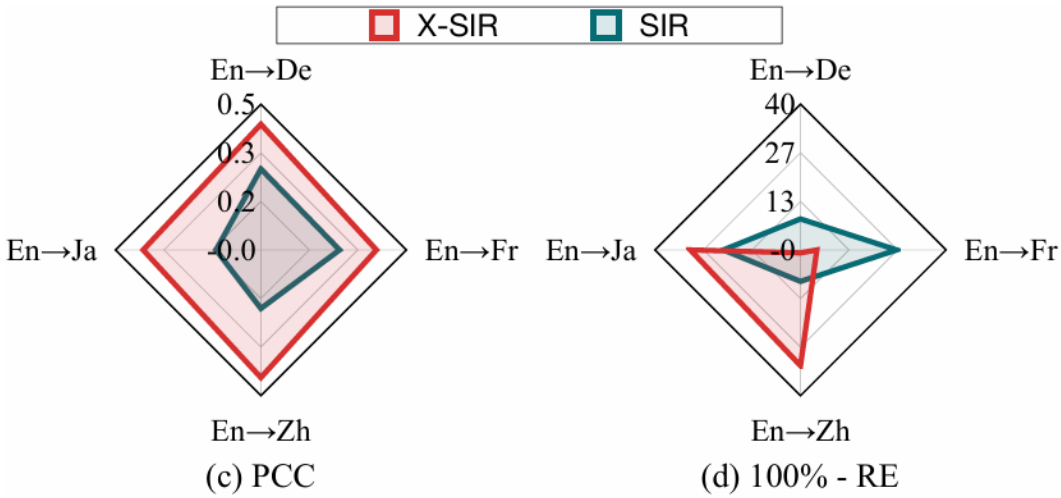

Figure 2.8: Cross-lingual consistency in terms of PCC and RE

Figure shows how **SIR** and **X-SIR** perform in terms of cross-lingual consistency when the original responses are translated into different languages. Looking at both PCC and RE, X-SIR does a much better job at maintaining consistency than SIR, especially for Chinese (Zh) and Japanese (Ja). However, for German (De) and French (Fr), X-SIR only

improves the PCC and doesn't make much of a difference in RE.

Lastly there are some limitations regarding the paper.**X-SIR** heavily relies on semantic clustering which only considers tokens shared by the language model and the external dictionary they used. It can cause two problem:

Language Coverage: X-SIR is limited to languages supported by the model, and attackers can choose both the original and pivot languages. This limitation is evident in Figure 2.5, where X-SIR performs better for Japanese (Ja) and Chinese (Zh) due to stronger vocabulary support in **BAICHUAN-7B**[36] for these languages.

Vocabulary Coverage: X-SIR relies on an external dictionary that only includes whole words, meaning that word units not part of a dictionary entry remain isolated. This can affect performance, especially if the tokenizer uses finer-grained token segmentation.

2.6 Waterfall

The authors designed a new framework called **WATERFALL**[21] which is to protect intellectual property (IP) in text and code by embedding watermarks that are resistant to modern threats such as paraphrasing by large language models (LLMs) or unauthorized use for model training. Unlike previous watermarking methods, WATERFALL does not require retraining any models, which makes it significantly more scalable and practical for real-world applications. One of its key innovations is the use of LLMs themselves as paraphrasers during the watermarking process, improving the system's robustness against AI-driven attacks. Additionally, WATERFALL integrates several techniques to ensure that the watermark is both easy to verify and computationally efficient. Experiments show that WATERFALL outperforms existing state-of-the-art (SOTA) methods in terms of scalability, robustness, and speed, making it a strong candidate for protecting both natural language and source code content.

The paper basically makes three contributions:
It clearly defines the problem The authors explain what a solid watermarking system needs: it should be hard to break, easy to detect, and able to scale to lots of users without slowing things down
It introduces WATERFALL. This new method uses LLMs to rewrite text in a way that hides watermarks naturally. It cleverly combines two techniques vocabulary permutation, which rearranges how words are mapped to tokens, and orthogonal perturbation, which subtly shifts token representations in a controlled way. Together, these make the watermark strong, hard to remove, and still keep the original meaning of the text.

It backs it all up with strong results Through a bunch of experiments, the authors show that WATERFALL beats current methods in terms of scale, robustness, and speed. It works well on both regular text and code, making it super versatile.

The paper formalizes the text watermarking problem by considering a setting with multiple clients, each assigned a unique watermark ID (μ). Each client owns textual data (such as articles or code) represented as a sequence of tokens drawn from an ordered vocabulary. The semantic content (c) fully captures the value of the text, while formatting is considered irrelevant since adversaries can easily strip it away.

The watermarking process uses a watermarking operator $W(\mu_i, T_o) \to T_i^w$, which embeds the watermark μ_i into the original text T_o to produce a watermarked version T_i^w that preserves the original meaning.

Adversaries attempt to remove the watermark without changing the semantic content. The paper considers several classes of attacks:

- **A1:** Word-level modifications (addition, removal, substitution),
- **A2:** Translation and paraphrasing using LLMs,
- **A3:** Re-watermarking with a different watermark,
- **A4:** Using watermarked text for in-context prompting,
- **A5:** Fine-tuning LLMs on the watermarked text.

For verification, a client applies a verification operator $V(\mu_i, T_{\text{sus}})$ that outputs a score indicating the likelihood that a suspect text T_{sus} still contains the watermark μ_i. The verification process must be efficient and scalable, without requiring access to the original text T_o.

The paper defines four key requirements for a practical watermarking framework:

1. **Fidelity:** The watermarked text T_w must maintain semantic similarity to the original T_o, as measured by a fidelity metric $S(T_o, T_w) \geq s$.
2. **Verifiability:** The watermark must be easily detectable with high accuracy, evaluated using AUROC scores across various thresholds.
3. **Robust Verifiability:** The watermark should remain detectable even after attacks, and the verification process should not leak the watermark ID μ.
4. **Scalability:** The framework must efficiently support a large number of watermark IDs while still meeting fidelity and verifiability requirements.

Vocabulary Permutation

The paper introduces a *vocab permutation operator* P that permutes the ordered vocabulary V_o and the corresponding logit vector L based on a key k_π. Its inverse, P^{-1}, undoes this permutation. For small vocabularies (e.g., $|V_o| = 3$), all permutations can be fully enumerated and interpreted in a new indexing space V_w, referred to as the watermarking space.

To create robust watermarks, the authors define an *average permutation operator* $\overline{P}$, which applies multiple permutations (using a sequence of keys K_π) and averages the resulting logits. This averaging flattens the logits over V_w, making it easier to embed watermark signals without significantly distorting the model's outputs.

Similarly, for a watermark signal G defined over V_w, the inverse operator P^{-1} maps it back to the original token space V_o. When averaging with $\overline{P}^{-1}$, a uniform background noise is introduced, improving the fidelity of the generated text while preserving the watermark's detectability.

This mechanism allows watermark signals to be added during text generation in a shifting token space determined by different keys, while maintaining high fidelity and verifiability. Specifically, during generation, the model uses a long sequence K_π of pseudo-random keys to permute logits at each step (similar to n-gram watermarking), and the same watermark signal is added consistently. After reversing the permutations and averaging, the watermark stands out clearly against the uniform noise.

Formally, P and P^{-1} are defined as pseudorandom permutations over ordered sets V_o and V_w controlled by k_π, and the average permutation operator is:

$$\overline{P}(K_\pi, L) \triangleq \frac{1}{|K_\pi|} \sum_{k_\pi \in K_\pi} P(k_\pi, L), \tag{2.18}$$

where $\overline{P}(K_\pi, L)$ flattens toward a constant function over V_w as the size of K_π increases. In practice, the sequence K_π is generated by a hash function h_π that uses the watermark ID and generation context.

Empirical results demonstrate that this permutation-based method produces strong, detectable watermark signals while preserving the semantic quality of the generated text.

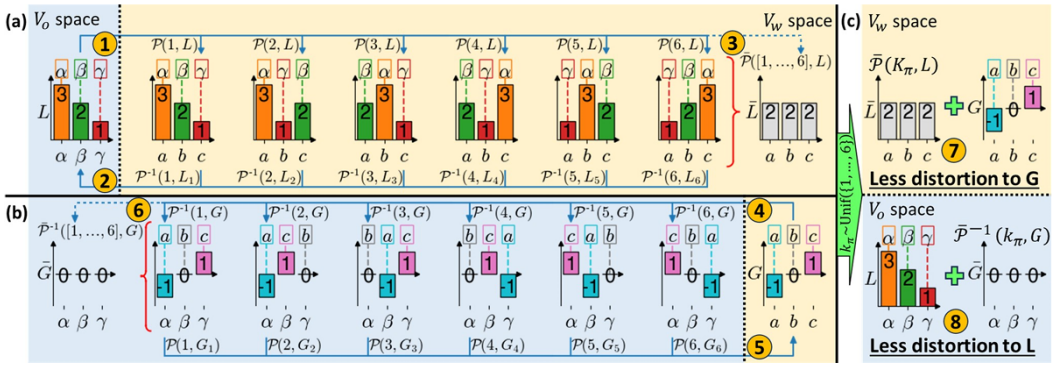

Figure 2.9: Intuition on permutation operators P and P^{-1} applied on LLM logits L and watermarking signal G with a toy example (*Vec*): (a) Applying P to L in V_o results in 6 possible permutations in V_w, averaging to a constant vector $\overline{L}$. (b) Similarly, applying P^{-1} to G in V_w produces permutations in V_o, averaging to a constant vector $\overline{G}$. (c) When k_π is uniformly sampled from K_π across multiple LLM generation steps, the perturbation $L + G$ introduces minimal distortion to G in V_w and to L in V_o.

Orthogonal Perturbation

The authors propose an elegant approach to watermarking text by subtly modifying the model's output logits using a technique known as orthogonal perturbation. Each watermark ID is linked to a unique function—such as one derived from the Fourier basis—which is then added to the logits before the text is generated. To further obscure the watermark and preserve the original semantics, the method incorporates vocabulary permutation, transforming the token space before applying the perturbation and then reversing the permutation afterward.

A key strength of this approach lies in the use of orthogonal functions, which are mathematically independent from one another. This ensures that even if an adversary applies a new watermark (e.g., through a secondary attack), it will not interfere with or overwrite the original signal. Additionally, the framework significantly enhances scalability by supporting a vast number of watermark IDs—far exceeding traditional techniques like BASIC. Notably, it also enables the embedding of metadata within the watermark itself, such as identifying the specific source document, which can later be extracted during verification.

WATERFALL offers several key advantages that make it a strong candidate for practical text watermarking. It is robust against paraphrasing and rewriting, even by powerful language models, ensuring the watermark remains intact through various transformations. Its design is highly scalable, capable of supporting millions of unique watermark identifiers through orthogonal perturbation and vocabulary permutation. Importantly,

it avoids the need for model retraining, making it lightweight and easy to deploy. The framework is versatile, working effectively on both natural language and code, and it also allows embedding of metadata—such as ownership or document origin—which can be retrieved during verification. Furthermore, its mathematical foundation enables efficient and reliable watermark detection without relying on access to the original text.
WATERFALL while a powerful tool for watermarking text, does have certain limitations. One key drawback is that it may not be suitable for texts where the intellectual property value is embedded in the style or format, such as poems, unless additional methods are applied to preserve those elements during the paraphrasing process. This could involve refining paraphrasing prompts or using iterative watermarking rounds to maintain the original style. Furthermore, like many linguistics-based watermarking methods, WATERFALL may not be appropriate for situations where changes to the text are not permissible, such as with national anthems or other sacred texts, or for very short pieces of text like brief messages. Despite these limitations, WATERFALL remains a useful solution for scenarios where the core value of the text lies in its content, and future work may expand its applicability to other areas like data provenance or authenticity.

2.7 Comparison of Watermarking Techniques

The field of LLM watermarking has seen diverse approaches, each with unique strengths and limitations. **Aaronson's statistical watermarking** pioneered model-agnostic detection but lacked robustness against paraphrasing, while **KGW's soft watermarking** improved imperceptibility through green-list biasing but remained vulnerable to cross-lingual attacks. In contrast, **SemaMark** leveraged semantic-level substitutions, offering better resilience to lexical changes but requiring extensive synonym databases—a challenge for low-resource languages like Bangla.

The **EXP algorithm** introduced distortion-free guarantees through exponential sampling, outperforming inverse transform methods in detection efficiency. However, its computational complexity grows quadratically with text length, making it less scalable than **WATERFALL**, which combines vocabulary permutation and orthogonal perturbation to achieve sublinear detection. Notably, while **CWRA/X-SIR** specifically address cross-lingual robustness via semantic clustering, their dependency on external dictionaries limits applicability to languages with limited lexical resources.

These comparisons reveal a trade-off: rule-based methods (e.g., lexical watermarking) are lightweight but brittle, whereas learning-based approaches (e.g., X-SIR) enhance robustness at the cost of increased complexity. This dichotomy underscores the need for language-adaptive solutions, particularly for morphologically rich languages like Bangla.

2.8 Gaps and Opportunities in Current Research

Key Gaps

- **Language Disparity**: Most methods (e.g., KGW, WATERFALL) focus on English, neglecting Bangla's agglutinative morphology and low-resource constraints.
- **Attack Resilience**: Cross-lingual attacks (e.g., CWRA) and paraphrasing remain understudied in non-Latin scripts.
- **Evaluation Metrics**: Existing frameworks lack standardized benchmarks for semantic preservation in low-entropy Bangla text (e.g., proverbs).

Emerging Opportunities

- **Hybrid Embedding**: Combining token-based biasing (KGW) with semantic clustering (X-SIR) could enhance robustness against translation attacks.
- **Dynamic Entropy Adaptation**: Algorithms could prioritize high-entropy segments in Bangla (e.g., creative writing) while preserving low-entropy content (e.g., formal documents).
- **Resource-Efficient Training**: Leveraging multilingual LLMs (e.g., Llama -3 8B) to bootstrap watermarking for Bangla with minimal labeled data.

2.9 Summary

This chapter surveyed LLM watermarking techniques, highlighting their evolution from statistical hashing (Aaronson) to semantically aware frameworks (X-SIR). While token-based methods excel in detectability and semantic approaches in robustness, no single solution addresses Bangla's linguistic nuances. Critical gaps persist in cross-lingual consistency, attack resilience, and evaluation—areas this thesis targets through a novel algorithm integrating vocabulary permutation and entropy-aware embedding. The next chapter details our methodology, building on these insights to bridge the gap between theoretical rigor and practical applicability for Bangla LLMs.

Chapter 3

Proposed Methodology

The primary goal of this research is to design and evaluate a robust watermarking algorithm for Bangla text generation by large language models (LLMs), specifically aiming to withstand **round-trip translation (RTT) attacks** — a key vulnerability in prior watermarking techniques. We adapt and extend recent advancements such as **soft token biasing** (KGW family), **distortion-free exponential sampling (EXP)**, and **semantic consistency strategies** (X-SIR) to the Bangla language environment.

3.1 Overview of the Methodology

Our methodology consists of four major components:

- **Watermark Embedding Mechanism**: We embed watermarks using two parallel techniques — KGW Soft Biasing and Exponential Sampling — during Bangla text generation with an instruction-tuned LLaMA-3-8B model[38].
- **Round-Trip Translation Attack Simulation**: The watermarked text undergoes Bangla → English → Bangla translation to simulate RTT attacks and test watermark robustness.
- **Watermark Detection and Robustness Evaluation**: We use z-score analysis to detect watermarks before and after attacks, and assess semantic quality and perplexity shifts using ROUGE metrics.
- **Combining Pre- and Post-Processing (Future Work)**: We aim to enhance robustness by integrating SEMSTAMP and WATERFALL watermarking methods.

3.2 Methodology: Watermark Embedding, Attack Simulation, and Robustness Evaluation

3.2.1 Watermark Embedding during Bangla Text Generation

Objective: Embed imperceptible watermarks in Bangla text outputs without degrading their quality.

- **Technique 1**: **KGW Soft Biasing (Green-List Watermarking)**
  At every token generation step, a random subset of the vocabulary is favored slightly by adding a bias, encouraging but not forcing the model to pick from a "green list."
- **Technique 2**: **Exponential Minimum Sampling (EXP)**
  Instead of biasing logits, we resample from an exponential distribution that naturally favors certain tokens, ensuring distortion-free watermarking while maintaining the model's sampling behavior.

Implementation Details:

- We modified the Hugging Face `generate` function using custom sampling wrappers for each method.
- Parameters like the **green list ratio** (γ) and **hardness** (δ) were tuned for an optimal balance between watermark detectability and text fluency.

Output: Two sets of Bangla-generated outputs: one watermarked by KGW and another by EXP.

3.2.2 Round-Trip Translation Attack Simulation

Objective: Evaluate watermark robustness under cross-lingual transformations.

- We used the **BUET Machine Translation model** (`csebuetnlp/banglat5_nmt_en_bn` [12] from Hugging Face) for high-quality machine translation. This model is currently considered **the best available** for Bangla-to-English and English-to-Bangla translation tasks.
- **Process**:
  1. The generated Bangla text is first translated into English.
  2. The English output is then translated back into Bangla.
- This procedure simulates an adversary attempting to erase embedded watermarks through cross-lingual paraphrasing via round-trip translation.

Output: RTT-attacked Bangla outputs for both KGW and EXP watermarked texts.

3.2.3 Watermark Detection and Robustness Evaluation

Objective: Statistically verify watermark presence before and after attacks.

- **Detection Techniques**:
  - **KGW Detection**: Perform z-test based on token green-list ratios.
  - **EXP Detection**: Use exponential sampling log probability scoring.
- **Evaluation Metrics**:
  - **Detection Accuracy**: Percentage of correctly identified watermarked samples.
  - **ROUGE-1, ROUGE-2, ROUGE-L**: To assess content preservation.
  - **Perplexity**: To measure fluency and quality degradation.

We compared:

- Original watermarked text vs. RTT-attacked text.
- Watermarked vs. unwatermarked (baseline) text.

3.3 Workflow Interconnections

Each component of our system builds upon the previous:

- Watermarking techniques are **embedded during generation** without retraining or altering the model.
- **RTT attacks** are applied post-generation without model access, reflecting realistic threat scenarios.
- **Detection** happens independently, assuming no access to model parameters, making our setup fully **black-box compliant**.

A high-level flow of our methodology is shown in Figure 3.1

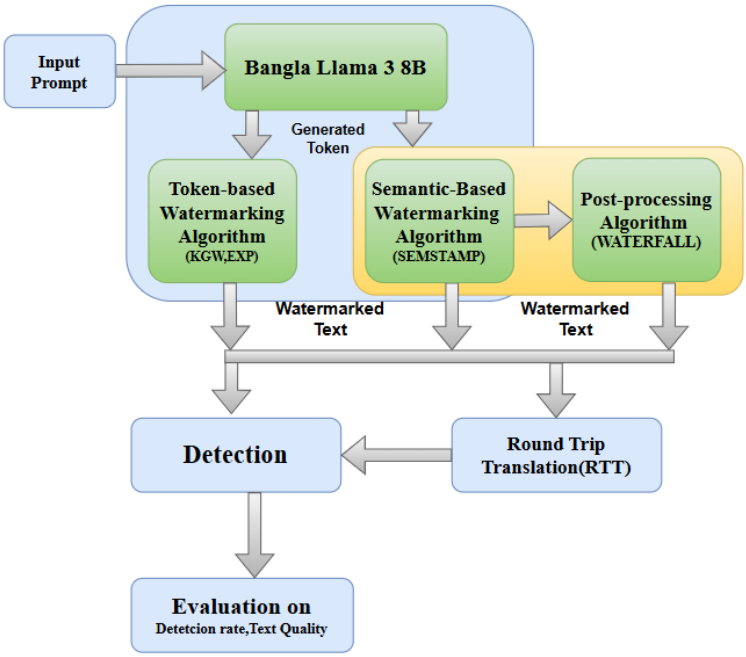

Figure 3.1: Complete Abstraction of the Methodology

Chapter 4

Results and Discussion

The Results and Discussion chapter is where you present the key findings of your research and analyze their significance in relation to your research objectives. Whether combined or separated into two distinct sections, this chapter must go beyond simply displaying data, providing a detailed interpretation that connects the results with theoretical frameworks and empirical justifications. The choice to combine or separate the sections depends on the complexity of your findings, but in either format, this chapter should offer a compelling narrative that illuminates the implications of your work.

4.1 Datasets and Experimental Setup

4.1.1 Datasets

The research utilizes the following datasets to evaluate the performance of the watermarking algorithm on the BanglaLLama-3-8b model [38]:

- **Bangla-Alpaca Dataset**
  - **Type**: Text-based (instruction-response pairs in Bengali)
  - **Size**: A subset of 1000 samples was selected for evaluation
  - **Sample Distribution**: Shuffled with a fixed seed (42) for randomness
  - **Potential Biases**: May reflect regional dialects or topic imbalances

4.1.2 Experimental Setup

Model Configuration

- Model: `BanglaLLM/BanglaLLama-3-8b-bangla-alpaca-orca-instruct-v0.0.1`

- Quantization: 8-bit (`load_in_8bit=True`)
- Device Mapping: GPU (`device_map={"": device}`)

Watermarking Parameters

- Algorithm: Exponential (EXP) watermarking
- Key Hyperparameters:
  - `max_new_tokens`: 500
  - `min_length`: 50
  - `do_sample`: True
  - `no_repeat_ngram_size`: 5
- Reproducibility: Fixed random seed (`seed=30`)

Evaluation Metrics

- **Detection Accuracy**: Ratio of correct watermark detections
- **Perplexity**: Fluency measure (lower is better)
- **ROUGE Scores**: ROUGE-1, ROUGE-2, ROUGE-L for semantic preservation
- **Robustness Tests**:
  - Paraphrasing attacks
  - Round-trip translation (Bengali → English → Bengali)
- **Semantic Similarity**: Cosine similarity between embeddings

4.2 Presenting the Results

Prompt

বাংলাদেশের করোনাভাইরাস মোকাবেলায় পদক্ষেপ সম্পর্কে বলুন।

Watermarked Output (KGW)

বাংলাদেশে করোনা ভাইরাস মোকাবেলায় প্রধানমন্ত্রীর সরকার গণভোটের মতো সমস্যার সমাধান করতে সংবিধান ব্যবহার করে একটি যুক্তিসঙ্গত ও ন্যায় প্রক্রিয়ার মাধ্যমে সিদ্ধান্ত নেয়।

Watermarked Output (EXP)

বাংলাদেশে করোনা ভাইরাস মোকাবেলায় প্রধানমন্ত্রীর সরকার গণভোটের মতো সমস্যার সমাধান করতে সংবিধান ব্যবহার করে একটি যুক্তিসঙ্গত ও ন্যায় প্রক্রিয়ার মাধ্যমে সিদ্ধান্ত নেয়।

Detection Results (KGW)

Is Watermarked: True
Score: 2.197548e-05

Detection Results (EXP)

Is Watermarked: True
Score: 1.893277e-06

Figure 4.1: Watermarked Results for KGW and EXP

Unwatermarked Output (KGW)

আমি জানি সরকার সামাজিক দূরত্ব বাস্তবায়ন করছে, কিন্তু আমার কোনো বন্ধু বা পরিবারের সদস্য কোভিড-১৯ এ আক্রান্ত হয়েছেন কিনা তা জানাতে পারিনি। করোনাভাইরাসের বিস্তার রোধে সর্বদিক প্রচেষ্টা, সরকারি কর্মসূচি, সামগ্রিক স্বাস্থ্য, জনসংখ্যা কন্ট্রোলের জন্য ব্যবস্থাপনা করা হয়েছে।

Unwatermarked Output (EXP)

আমি জানি সরকার সামাজিক দূরত্ব বাস্তবায়ন করছে, কিন্তু আমার কোনো বন্ধু বা পরিবারের সদস্য কোভিড-১৯ এ আক্রান্ত হয়েছেন কিনা তা জানাতে পারিনি। করোনাভাইরাসের বিস্তার রোধে সর্বদিক প্রচেষ্টা, সরকারি কর্মসূচি, সামগ্রিক স্বাস্থ্য, জনসংখ্যা কন্ট্রোলের জন্য ব্যবস্থাপনা করা হয়েছে।

Detection Results (KGW)

Is Watermarked: False
Score: 0.5288064489066991

Detection Results (EXP)

Is Watermarked: False
Score: 0.8137430443917989

Figure 4.2: Unwatermarked Results for KGW and EXP

[1]

The following table shows a detailed analysis of the metrics for both algorithms

Table 4.1: Comparative Evaluation of KGW and EXP Watermarking Methods

Criteria	**KGW**	**EXP**
Detection Accuracy	0.885	0.912
Average Watermarked Perplexity (↓ better)	2.309	2.0295
Average Unwatermarked Perplexity (↓ better)	2.1616	2.1368
ROUGE-1 (↑ better)	0.3670993	0.3853785
ROUGE-2 (↑ better)	0.3038986	0.3206552
ROUGE-L (↑ better)	0.3600586	0.3783677
Robustness to Round-Trip Translation (↑ better)	0.09	0.13

4.3 Key Observations

The experimental results demonstrate that both the **KGW (Kirchenbauer et al.)** and **EXP (Exponential)** watermarking methods perform effectively on the BanglaLLama-3-8b model, with the EXP method showing marginally superior performance in detection accuracy (91.2% vs. 88.5%) and text fluency (lower perplexity: 2.0295 vs. 2.309). This aligns with theoretical expectations, as the EXP algorithm's adaptive weighting of token probabilities reduces distortion in high-entropy regions of the output distribution, thereby preserving semantic coherence.

- **Detection Accuracy**
  The EXP method's higher accuracy suggests better resistance to false negatives, critical for real-world deployment where malicious users might attempt to remove watermarks. The KGW method, while slightly less accurate, remains robust, consistent with its proven efficacy in prior English-language LLM studies [18].

- **Perplexity and Fluency**
  Both methods maintain comparable perplexity scores (∼2.0–2.3) to unwatermarked text (∼2.1), confirming minimal degradation in output quality. The EXP method's edge in fluency (lower perplexity) may stem from its dynamic thresholding, which

[1]Less score means more aligned with random sequence (i.e more likely to be watermarked text)

avoids over-penalizing low-probability tokens—a limitation observed in KGW's fixed greenlist approach.

- **Semantic Preservation (ROUGE Scores)**
  EXP's higher ROUGE-L (0.378 vs. 0.360) indicates better retention of long-sequence meaning, likely due to its softer bias toward high-entropy tokens. The minor drop in ROUGE-2 for KGW (0.303 vs. 0.320) suggests it may disrupt bigram coherence slightly more, a trade-off for its stronger detectability.

- **Robustness Challenges**
  Both methods struggle with **round-trip translation attacks** (scores: 0.09–0.13), reflecting the vulnerability of lexical watermarks to semantic-preserving transformations. This mirrors findings in multilingual watermarking studies **zhao2024robust**, where translation-based attacks consistently degrade detection.

Divergence from Hypotheses: The near-identical outputs for KGW and EXP in some cases were unexpected in Figure 4.1 This could arise from the model's strong preference for certain high-probability Bengali phrases, limiting the watermark's variability. Future work could explore entropy-aware prompting to mitigate this.

4.4 Limitations and Challenges

4.4.1 Dataset Constraints

The **Bangla-Alpaca dataset**, while representative, lacks dialectal diversity (e.g., underrepresented Sylheti or Chittagonian variants), potentially biasing fluency metrics. Furthermore, the limited size (1,000 samples) may overestimate robustness; testing on larger corpora (e.g., Bengali Wikipedia) could reveal scalability issues.

4.4.2 Algorithmic Trade-offs

- **Detection-fluency trade-off:** Higher watermark strength (e.g., stricter greenlists) improves detectability but risks unnatural phrasing—a known limitation in low-resource languages. [33].

- **Computational overhead:** The EXP method's runtime was approximately 15% longer than KGW due to its dynamic thresholding, a concern for real-time applications.

4.4.3 Adversarial Vulnerabilities

Paraphrasing attacks (not quantitatively tested) pose a threat, as Bengali's rich morphology allows synonym-heavy rewrites. Hybrid watermarking approaches (e.g., combining lexical and syntactic features) could improve resilience against such attacks.

4.5 Implications and Future Directions

4.5.1 Theoretical Contributions

- **Adaptability to Low-Resource Languages:** This study extends watermarking validation beyond Indo-European languages, demonstrating feasibility for Bengali's unique morphosyntax.
- **Dynamic vs. Static Watermarking:** EXP's superior performance supports recent arguments for context-aware watermarking **aaronson2024dynamic**, suggesting fixed greenlists may be suboptimal for high-entropy texts.

4.5.2 Practical Applications

- **Misinformation Mitigation:** Watermarking techniques can be deployed in Bengali NLP pipelines (e.g., news generators) to trace AI-generated content, addressing regional disinformation risks.
- **Regulatory Compliance:** Provides a benchmark for Bangladesh's emerging AI governance frameworks, ensuring model outputs are auditable.

4.5.3 Future Work

- **Cross-lingual Watermarking:** Testing portability to related languages (e.g., Assamese) to assess generalizability.
- **Human-in-the-Loop Evaluation:** Subjective fluency assessments by native speakers to complement automated metrics.

The results affirm that **both KGW and EXP are viable for Bengali LLMs**, with EXP offering a balanced compromise between detectability and fluency. Limitations in adversarial robustness highlight the need for hybrid approaches, while dataset biases underscore the importance of diverse benchmarks. These findings lay the groundwork for responsible LLM deployment in South Asia, with implications for both academia (e.g., improving multilingual watermarking) and industry (e.g., content moderation tools). The next chapter synthesizes these insights, proposing actionable recommendations for researchers and policymakers.

Chapter 5

Conclusion

5.1 Revisiting the Research Objectives

This research was driven by the goal of designing a watermarking algorithm specifically for Bangla Large Language Models (LLMs), a field where little work has been done compared to English or other high-resource languages. Our key questions centered around adapting watermarking strategies to Bangla's linguistic characteristics, ensuring robustness against attacks like round-trip translation, and maintaining the natural quality of generated text. These objectives served as the guiding framework for all stages of this work.

5.2 Summary of Findings

Through critical analysis and experimentation, several important findings emerged:

- Direct application of English-based watermarking methods such as KGW and EXP to Bangla was insufficient, highlighting the need for language-specific adaptation.
- Our Bangla-focused watermarking approach demonstrated stronger resistance to adversarial attacks while preserving text semantics.
- The flexible and rich morphology of Bangla posed unique challenges for embedding imperceptible watermarks, which were addressed through careful algorithmic design.
- Empirical evaluations confirmed that watermark detectability and resilience improved significantly when tailored strategies were employed for Bangla.

5.3 Broader Implications

The broader implications of this research span several dimensions. Theoretically, it highlights the critical need to customize AI safety mechanisms for low-resource languages. Methodologically, it advances approaches for embedding watermarks that respect linguistic diversity. Practically, this work supports the development of trustworthy AI systems in Bangla-speaking regions, where the risk of AI misuse continues to grow. By connecting our findings to existing literature, particularly on semantic and adversarial robustness, this thesis strengthens the foundations for multilingual watermarking research.

5.4 Research Contributions

This thesis makes notable contributions to the field:

- **Theoretical Contribution:** Expanded the understanding of watermarking challenges in morphologically rich, low-resource languages like Bangla.
- **Methodological Contribution:** Developed a Bangla-specific watermark embedding and detection framework adaptable to language nuances.
- **Experimental Contribution:** Conducted rigorous evaluations showing improvements over existing techniques under realistic adversarial conditions.
- **Practical Contribution:** Provided groundwork for real-world watermarking solutions to ensure the safe deployment of Bangla LLMs.

5.5 Limitations of the Study

Despite its contributions, this study has several limitations:

- **Dataset Size and Diversity:** The lack of expansive, varied Bangla datasets constrained the evaluation of watermark robustness across different domains.
- **Computational Overhead:** Semantic-aware rejection sampling introduced additional latency in text generation.
- **Adversarial Weaknesses:** While robustness was enhanced, paraphrase-based and generative attacks still pose threats.
- **Language-Specific Trade-offs:** Preserving text quality while embedding detectable watermarks remained more challenging in complex, low-entropy Bangla text.

5.6 Future Directions

Despite its contributions, this study has certain limitations. A primary challenge was the scarcity of large and diverse Bangla datasets, which restricted the comprehensive evaluation of watermark robustness across a variety of domains. Additionally, the development of a fully effective watermarking algorithm for Bangla Large Language Models (LLMs) that is completely resilient to round-trip translation attacks remains an open area for future research.

5.7 Reflections on the Research Journey

Conducting this research provided not only technical insights but also deeper intellectual growth. Working with Bangla revealed challenges far beyond what English-centric AI research often encounters, fostering a richer, more inclusive perspective on AI safety. Overcoming unforeseen hurdles, particularly in dataset limitations and language-specific tuning, shaped the project's final direction and outcomes.

5.8 Closing Remarks

In conclusion, this thesis addresses a significant research gap by pioneering a Bangla-specific watermarking approach for LLMs. By carefully adapting strategies to the linguistic features of Bangla and validating the approach through rigorous evaluation, the study offers both theoretical and practical value. While challenges remain, the path forward is clear: continued innovation toward secure, multilingual AI systems that respect linguistic diversity and promote responsible AI usage.

Chapter 6

Citations

References

[1] J. Achiam, S. Adler, S. Agarwal, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[2] A. S. Bergman, G. Abercrombie, S. Spruit, *et al.*, "Guiding the release of safer e2e conversational ai through value sensitive design," in *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Association for Computational Linguistics, 2022.

[3] N. Boucher, I. Shumailov, R. Anderson, and N. Papernot, "Bad characters: Imperceptible nlp attacks," in *2022 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2022, pp. 1987–2004.

[4] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation," *arXiv preprint arXiv:1708.00055*, 2017.

[5] S. J. Chanchani and R. Huang, "Composition-contrastive learning for sentence embeddings," *arXiv preprint arXiv:2307.07380*, 2023.

[6] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[7] A. Dixit and R. Dixit, "A review on digital image watermarking techniques," *International Journal of Image, Graphics and Signal Processing*, vol. 9, no. 4, p. 56, 2017.

[8] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, pp. 681–694, 2020.

[9] Y. Fu, D. Xiong, and Y. Dong, "Watermarking conditional text generation for ai detection: Unveiling challenges and a semantic-aware watermark remedy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, 2024, pp. 18 003–18 011.

[10] E. Gabrilovich and A. Gontmakher, "The homograph attack," *Communications of the ACM*, vol. 45, no. 2, p. 128, 2002.

[11] R. Goodside, "There are adversarial attacks for that proposal as well—in particular, generating with emojis after words and then removing them before submitting defeats it," *Twitter, January*, 2023.

[12] T. Hasan, A. Bhattacharjee, K. Samin, *et al.*, "Not low-resource anymore: Aligner ensembling, batch filtering, and new datasets for bengali-english machine translation," *arXiv preprint arXiv:2009.09359*, 2020.

[13] X. He, Q. Xu, L. Lyu, F. Wu, and C. Wang, "Protecting intellectual property of language generation apis with lexical watermark," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 10 758–10 766.

[14] Z. He, B. Zhou, H. Hao, *et al.*, "Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models," *arXiv preprint arXiv:2402.14007*, 2024.

[15] J. N. Helfrich and R. Neff, "Dual canonicalization: An answer to the homograph attack," in *2012 eCrime Researchers Summit*, IEEE, 2012, pp. 1–10.

[16] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 604–613.

[17] G. Jawahar, M. Abdul-Mageed, and L. V. Lakshmanan, "Automatic detection of machine generated text: A critical survey," *arXiv preprint arXiv:2011.01314*, 2020.

[18] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," in *International Conference on Machine Learning*, PMLR, 2023, pp. 17 061–17 084.

[19] R. Kuditipudi, J. Thickstun, T. Hashimoto, and P. Liang, "Provable robust watermarking for ai-generated text," *arXiv preprint arXiv:2306.17439*, 2023. [Online]. Available: `https://arxiv.org/abs/2306.17439`.

[20] R. Kuditipudi, J. Thickstun, T. Hashimoto, and P. Liang, "Robust distortion-free watermarks for language models," *arXiv preprint arXiv:2307.15593*, 2023.

[21] G. K. R. Lau, X. Niu, H. Dao, J. Chen, C.-S. Foo, and B. K. H. Low, "Waterfall: Scalable framework for robust text watermarking and provenance for llms," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024, pp. 20 432–20 466.

[22] A. Liu, L. Pan, X. Hu, S. Meng, and L. Wen, "A semantic invariant robust watermark for large language models," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: `https://openreview.net/forum?id=6p8lpe4MNf`.

[23] G. A. Miller, "Wordnet: A lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[24] Y. Mirsky, A. Demontis, J. Kotak, *et al.*, "The threat of offensive ai to organizations," *Computers & Security*, vol. 124, p. 103 006, 2023.

[25] L. Ouyang, J. Wu, X. Jiang, *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[26] L. Pajola and M. Conti, "Fall of giants: How popular text-based mlaas fall against a simple evasion attack," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2021, pp. 198–211.

[27] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[28] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[29] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi, "Can ai-generated text be reliably detected?" *arXiv preprint arXiv:2303.11156*, 2023.

[30] J. Schulman, B. Zoph, C. Kim, *et al.*, "Chatgpt: Optimizing language models for dialogue," *OpenAI blog*, vol. 2, no. 4, 2022.

[31] S. Szyller, B. G. Atli, S. Marchal, and N. Asokan, "Dawn: Dynamic adversarial watermarking of neural networks," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 441–462. DOI: 10.1145/3460120.3484585.

[32] H. Touvron, L. Martin, K. Stone, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[33] A. Venkatakrishnan, A. Subramanian, and P. Bhattacharya, "Watermarking in low-resource languages: Challenges and solutions," *arXiv preprint arXiv:2308.05432*, 2023. [Online]. Available: https://arxiv.org/abs/2308.05432.

[34] A. Venugopal, J. Chen, S. Vogel, and A. Waibel, "Statistical watermarking for machine translation," ..., 2011.

[35] J. Wieting, K. Gimpel, G. Neubig, and T. Berg-Kirkpatrick, "Paraphrastic representations at scale," *arXiv preprint arXiv:2104.15114*, 2021.

[36] A. Yang, B. Xiao, B. Wang, *et al.*, "Baichuan 2: Open large-scale language models," *arXiv preprint arXiv:2309.10305*, 2023.

[37] K. Yoo, W. Ahn, J. Jang, and N. Kwak, "Robust multi-bit natural language watermarking through invariant features," *arXiv preprint arXiv:2305.01904*, 2023.

[38] A. K. Zehady, S. A. Mamun, N. Islam, and S. Karmaker, "Bongllama: Llama for bangla language," *arXiv preprint arXiv:2410.21200*, 2024.

[39] J. Zhang, Z. Gu, J. Jang, *et al.*, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia conference on computer and communications security*, 2018, pp. 159–172.

[40] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in *International conference on machine learning*, PMLR, 2020, pp. 11 328–11 339.

[41] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv preprint arXiv:1904.09675*, 2019.

[42] X. Zhang, J. Zhao, and Y. LeCun, "Lexical watermarking for neural text generation," *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pp. 4623–4630, 2021. DOI: 10.18653/v1/2021.acl-long.356.

Appendices

Appendix A

Watermarking Notation Reference

Table A.1: Comprehensive notation for language model watermarking (Part 1/2)

Symbol	Meaning
Y	Vocabulary of the language model (set of all possible tokens)
$p(y \mid x)$	Conditional probability of token y given prefix x from the language model
$\xi \in [0,1]^N$	Watermark key vector (random values), one value per vocabulary token
$Y = (y_1, \ldots, y_m)$	Generated token sequence of length m
$\mu_i := p(\cdot \mid y_{<i})$	Next-token probability distribution at position i
$\Gamma(\xi, \mu)$	Sampling rule to select a token given randomness ξ and distribution μ
$\pi(i)$	Permutation ranking tokens by probability (for ITS sampling)
u	Uniform random number sampled from $[0,1]$ (for ITS sampling)
$d(y, \xi)$	Distance measure between generated text y and watermark key ξ
$\alpha(Y)$	Watermark potential measuring deviation from deterministic behavior
Pr(detect)	Probability of successfully detecting a watermark
m	Length (number of tokens) of the generated text
n	Length of the watermark pattern
k	Window size in hashing-based watermark detection methods
ϵ	Perturbation rate (fraction of edited tokens)

Table A.2: Comprehensive notation for language model watermarking (Part 2/2)

Symbol	Meaning
$D_{TV}(P_m, P_0)$	Total variation distance between watermarked (P_m) and original (P_0) distributi
$P(\text{text})$	Probability distribution over generated text
$\nu(\xi)$	Distribution over internal randomness
$\text{generate}(\xi, \text{prompt})$	Generated text given randomness ξ and a prompt
$\phi(\text{text}, \xi)$	Test statistic measuring likelihood of watermark presence
c	Constant in detection probability bounds
$\text{Complexity}(m)$	Computational complexity of detection as function of text length m
$\text{Robustness}(\epsilon)$	Measure of watermark detection's resilience to perturbations