Text Based Steganography using Prepositional Encoding with LLMs

Vaishnavi Adidam

Abstract - **Steganography aims to conceal information without arousing suspicion. Traditional text-based steganographic methods rely on predefined cover texts, making them vulnerable to detection. This paper presents a novel preposition-based encoding scheme that leverages a commercial off-the-shelf large language model (LLM) to generate cover text. By utilizing LLM-generated prose that closely resembles human writing, the proposed system reduces detectability by both automated statistical analysis tools and human reviewers. This approach, however, incurs a substantially larger expansion ratio than the original message and exhibits an error rate that grows with message length. These findings highlight a fundamental trade-off among textual naturalness, message capacity, and transmission accuracy.**

Index Terms - Generative AI, Information Hiding, Keyword Encoding, Steganography

Introduction

Steganography is the practice of hiding information in plain sight [1]. Unlike cryptography, which obscures a message by rendering it unintelligible, steganography conceals the very existence of a secret. While an encrypted file implicitly signals the presence of sensitive information, a steganographic message is designed to appear benign and unremarkable.

This distinction can be critical in adversarial or authoritarian environments, where the mere presence of encrypted communication may raise suspicion. In such contexts, the most effective protection may be ensuring that a secret is not perceived to exist at all. Steganography is an ancient technique; as early as 440 BCE, the historian Herodotus described how Demaratus carved a warning into a wooden tablet and concealed it beneath a layer of wax. To guards, the tablet appeared blank, but the hidden message was revealed when the wax was removed.

Over time, steganographic methods became increasingly sophisticated. During World War I, German operatives employed seemingly mundane correspondence to disguise strategic intelligence:

> PRESIDENT'S EMBARGO RULING SHOULD HAVE IMMEDIATE NOTICE. GRAVE SITUATION AFFECTING INTERNATIONAL LAW. STATEMENT FORESHADOWS RUIN OF MANY NEUTRALS. YELLOW JOURNALS UNIFYING NATIONAL EXCITEMENT IMMENSELY.

When the first letter of each word is extracted, the concealed message reads:

> PERSHING SAILS FROM N.Y. JUNE 1

Current State of Steganography

Research in computer security has historically focused on cryptography, which forms the foundation of secure internet communication and privacy through protocols such as SSL/TLS. Contemporary steganographic techniques generally fall into two broad categories: digital media encoding and text-based methods [2].

Media-based steganography embeds data within digital artifacts such as images, audio, video, or PDF files by introducing imperceptible modifications. Although effective in controlled settings, these approaches are often fragile; common transformations such as format conversion or compression can inadvertently remove the hidden payload.

Text-based steganography, by contrast, exploits the flexibility of natural language [3]. These systems use either predefined or dynamically generated cover text and encode information through subtle linguistic modifications, including spelling variations, punctuation changes, or synonym substitution. The primary objective is to preserve semantic coherence while embedding covert data.

Large Language Models

Recent advances in generative artificial intelligence have been both rapid and transformative [4]. Considerable effort is typically devoted to mitigating hallucinations—model outputs that are factually incorrect or nonsensical. However, hallucinated text often remains linguistically plausible, particularly in narrative form, making it well suited for information hiding. This work exploits that plausibility by introducing a method that encodes information through constrained preposition usage while relying on an LLM to generate coherent cover text.

An important consequence of using LLM-generated text is increased resistance to steganalysis techniques that depend on detecting statistical irregularities in language distributions. Current taxonomies of steganographic text detection generally fall into two categories: modification-based and generation-based methods [5]. Generation-based approaches are specifically designed to detect LLM-generated steganographic text and range from statistical tests to deep learning–based classifiers. The objective of the proposed approach is to subvert both classes of detection mechanisms.

Algorithm

I. High-Level Overview

The proposed approach converts plaintext characters into their hexadecimal representations and maps each hexadecimal digit to a specific preposition. An LLM is then tasked with generating a coherent narrative that incorporates the resulting preposition sequence while adhering strictly to the predefined mapping.

II. Encoding Procedure

The encoding process consists of the following:

1. Each character in the plaintext—including whitespace and punctuation—is converted to its corresponding ASCII hexadecimal value. Hexadecimal encoding is used because it requires only 16 symbols, enabling a compact mapping.
2. Each hexadecimal digit is mapped to a predefined preposition according to an arbitrary but fixed lookup table.

Hex Value to Preposition
A Way to Convert Prepositions to Hex

Hex	Type of Text
0	In
1	On
2	At
3	By
4	For
5	With
6	About
7	Against
8	Between
9	Into
a	Through
b	During
c	Before
d	After
e	Above
f	Below

3. A model-specific prompt instructs the LLM to generate cover text that incorporates the prepositions in the exact specified order, while prohibiting the use of any additional prepositions.

III. Decoding Procedure

Decoding reverses the encoding process:

1. Prepositions are extracted from the cover text in sequence.
2. Each preposition is mapped back to its associated 4-bit value.
3. The 4-bit values are combined into 8-bit (ASCII) or 8- to 32-bit (Unicode) representations.
4. The resulting binary values are translated back into plaintext.

Example

To illustrate the proposed method, this section presents the decoding process prior to encoding, providing context for how the generated cover text is structured.

I. Decoding Example

Given the following cover text:

> *The wind pushed against the ancient stone, a silent battle waged for dominion. Whispers spread about the mountain's secrets, tales shared with cautious awe. The peak stood defiant against the sky's relentless assault, carved deep by millennia. Life itself struggled against the harsh elements, enduring for another dawn.*

1. Extracted prepositions (in order): against, for, about, with, against, by, against, for.
2. Corresponding hexadecimal values: 74, 65, 73, 74.
3. ASCII decoding yields the plaintext: "test".

II. Encoding Example

If we wanted to encode “test."

1. Encoding the plaintext "test" proceeds as follows:
2. The string is converted to hexadecimal values: 74, 65, 73, 74.
3. Each hexadecimal digit is replaced with its corresponding preposition.
4. A constrained prompt directs the LLM to generate grammatically correct cover text containing the prepositions in sequence.

Craft a concise and imaginative text, that seamlessly integrates the following word group. Word Group: against, for, about, with, against, by, against, for. Each word must be used only once, in the exact same order, and use no other prepositions.

4. Google Gemini resulted in the following:

The wind pushed against the ancient stone, a silent battle waged for dominion. Whispers spread about the mountain's secrets, tales shared with cautious awe. The peak stood defiant against the sky's relentless assault, carved deep by millennia. Life itself struggled against the harsh elements, enduring for another dawn.

Implementation and Performance

I. Implementation

The proof-of-concept implementation comprises a Node.js front end and a Google Gemini back end. The front end performs binary conversion and prepositional mapping, while the back end generates steganographic cover text using the LLM. Shown in Figures 1 and 2 for encoding and decoding respectively:

Figure 1

Figure 2

II. Performance Evaluation

Because the system relies on a commercial cloud-based LLM that undergoes frequent updates, performance characteristics may vary over time. Two primary metrics are evaluated: expansion ratio and decoding error rate.

IIa. Expansion Ratio

Experiments were conducted using message lengths increasing in five-character increments, with ten trials per length. On average, the system produced approximately 50 characters of cover text for each character of encoded plaintext. As shown in Figure 3 respectively:

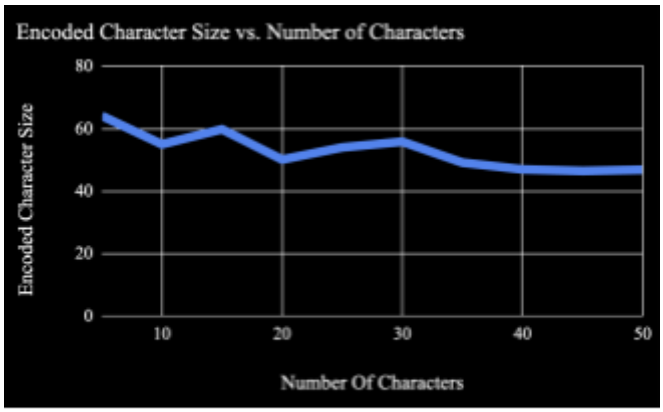

Figure 3

IIb.Error Rate

The decoding error rate is defined as the character-level discrepancy between the original plaintext and the decoded output. Due to the nondeterministic nature of LLMs, strict adherence to prepositional constraints cannot be guaranteed. Empirical results indicate that error rates remain low for short messages but increase sharply when the plaintext exceeds approximately 50 characters. As shown in Figure 4 respectively:

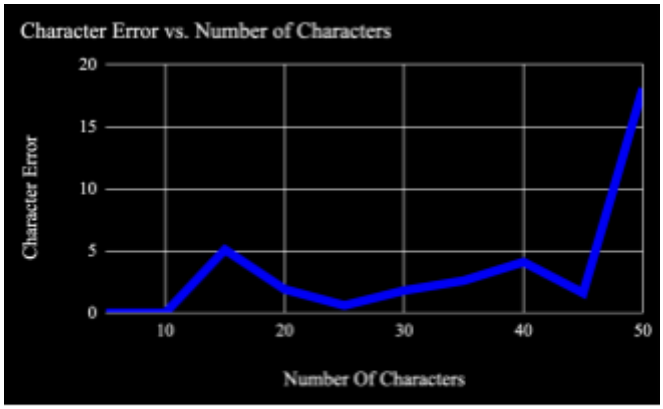

Figure 4

III. Comparison with StegGPT

StegGPT represents one of the few contemporary LLM-based steganography systems with publicly available sample outputs. Compared to the character-level encoding strategy employed by StegGPT, the proposed

preposition-based method produces substantially larger encoded outputs, reflecting a trade-off between linguistic naturalness and message capacity [6].

To compare with StegGPT character vs word based approach:

Test Number	StegGPT Message	StegGPT Encoded Length	Google Gemini Encoded Length
1	Sezkgpyrtlq	28	695
2	xgelwoxlxh	34	619
3	aobeaboirg	26	702
4	nlfiixgtql	33	479
5	ywtqxbjysc	32	737
6	kfilqfjicz	35	581
7	smflttuqng	35	560
8	aeauizmlsy	31	776
9	bdkdnwulip	30	400
10	usiukgexck	27	639

Conclusion

This paper presents a conceptually sound preposition-based steganographic encoding scheme that leverages LLM-generated text to conceal information within linguistically natural prose. Experimental results demonstrate that the approach is effective for short messages, successfully producing cover text that is difficult for both automated steganalysis tools and human reviewers to distinguish from ordinary writing. By shifting the burden of natural-language generation to a large language model, the method avoids many of the statistical artifacts that plague traditional text-based steganographic techniques.

Despite these advantages, the approach does not scale well to longer messages. The primary limitation arises from the nondeterministic behavior of general-purpose LLMs, which can deviate from strict prepositional constraints as input length increases. This deviation leads to decoding errors and limits the practical message capacity of the system. As such, the proposed method highlights an inherent trade-off between linguistic naturalness, encoding capacity, and reliability. Addressing this trade-off will be essential for deploying LLM-based steganography in real-world communication scenarios.

Future Work

This proof of concept employs sixteen prepositions, each encoding four bits of information, to demonstrate the feasibility of preposition-based encoding. However, English contains more than 100 commonly recognized prepositions. Expanding the mapping table to incorporate a larger prepositional vocabulary could substantially improve the expansion ratio and increase overall message capacity.

Future research should investigate whether large language models can maintain grammatical correctness and stylistic coherence when subjected to increasingly dense and restrictive prepositional constraints. Potential directions include fine-tuning existing LLMs specifically for constrained text generation, incorporating reinforcement learning to penalize constraint violations, or developing specialized language models designed explicitly for steganographic applications. Additionally, systematic evaluation against modern steganalysis techniques will be necessary to quantify the security benefits of expanded mappings.

References

1. Lookwood, R., & Curran, K. (2017). *Text-based steganography. International Journal of Information Privacy, Security and Integrity*, 3(2), 134–153. https://doi.org/10.1504/IJIPSI.2017.088700
2. Krishnan, B., Thandra, K. P., & Baba, S. M. (2017). An overview of text steganography. In *Proceedings of the Fourth International Conference on Signal Processing, Communication and Networking* (ICSCN) (pp. 1–6). IEEE. https://doi.org/10.1109/ICSCN.2017.8085643 https://doi.org/10.1109/ACCESS.2025.3568339
3. Chapman, M., Davida, G. I., & Rennhard, M. (2001). A practical and effective approach to large-scale automated linguistic steganography. In *Proceedings of the 4th International Conference on Information Hiding* (pp. 156–165). Springer.
4. Bick, A., Blandin, A., & Deming, D. J. (2026). The rapid adoption of generative AI. *Management Science*. https://doi.org/10.1287/mnsc.2025.02523
5. Tang, Y., Wang, Y., Zhang, R., & Liu, J. (2024). Rethinking LLM and linguistic steganalysis: An efficient detection of strongly concealed stego. IEEE Signal Processing Letters, 31, 1599–1603. https://doi.org/10.1109/LSP.2024.3413552
6. Ahmed Adeeb, O. F., & Kabudian, S. J. (2025). StegGPT: A novel foundation-model-based character-level linguistic steganography method utilizing large language models. *IEEE* Access, 13, 89915–89925.

Author Information

Vaishnavi Adidam, Student, Langley High School, Mclean VA, USA