Enhancement of Markov Chain-Based Linguistic Steganography with Binary Encoding for Securing Legal Documents

Ace Byrone Halili[1], Gideon Antony Salangsang[1]
[1]*Student, College of Information Systems and Technology Management, Pamantasan ng Lungsod ng Maynila (University of the City of Manila), Manila, Philippines*
Corresponding Author: acebyronehalili@gmail.com

***Abstract*: The Markov Chain-based linguistic steganography algorithm can effectively hide information within human-like cover text, but it is highly limited in processing speed. A traditional implementation relying on Huffman tree-based encoding mainly suffers from slow processing due to the computational overhead of building the tree itself. To address this issue, this study proposes an enhanced algorithm using binary indexing for constant time complexity. The results were experimentally calculated using models of varying state sizes derived from the same text corpus as a control variable. Perplexity analysis was also employed to evaluate imperceptibility and ensure there were no drawbacks to the cover media's integrity. The results indicate that the enhanced algorithm improves processing speed by up to 54 times across all state sizes without compromising imperceptibility. This establishes that the enhancements yielded a significantly faster processing speed for the existing algorithm while remaining secure in its concealment. In practice, the algorithm was applied in legal document storage to strengthen its security.**

***Keywords*: Steganography, Markov Chain, Security.**

1. Introduction

The concept of Markov Chains, named after the Russian mathematician Andrey Markov, encompasses a mathematical framework for depicting sequences of events where the probability of each event depends only on the state attained in the previous event. This characteristic of memory lessness, formally known as the Markov property, makes Markov Chains exceptionally useful in modeling and analyzing a variety of stochastic processes (Ribeiro, 2024). Steganography is a technique that allows for the secure transmission of sensitive information over an exposed public channel (Madison & Dickman, 2007). It does this by hiding said information within seemingly harmless media files such as images or video. Along with steganography, another popular way of concealing information is through cryptography. The benefit of the former over the latter lies in its ability to hide that a secret message is being transmitted in the first place (Mishra & Bhanodiya, 2015). This makes steganography a valuable research topic in cybersecurity. In the field of cybersecurity, particularly in text steganography, Markov Chains can be used to enhance both the concealment and security of encoded messages. The article investigates a sophisticated method that utilizes these chains not only for encoding but also to ensure that the generated texts mimic natural linguistic patterns. This advanced approach markedly improves upon traditional text steganography by embedding messages that adhere closely to the syntactic and semantic characteristics of the original language model. This strategy maintains textual integrity and increases the secrecy of communications, essential in an era of widespread digital surveillance (Moraldo, 2024). Various steganographic techniques mostly differ in the media they choose to hide information in. One of the most popular mediums used is through text, given its wide use over the internet. Despite its popularity and practicality, though, text-based steganography is infamously hard to conduct. According to Singh et al. (2009), most steganographic techniques use cover media such as images or sounds rather than text because it is much easier to find redundant bits within them. Information is also harder to embed in text because a slight change in a textual cover is more apparent than in a picture or video file cover.

Nonetheless, extensive research has been done to make such methods usable. Reliable linguistic steganographic applications generally come in two forms: format-based and content-based methods (Yang et al., 2018). Format based methods like line-

Manuscript revised December 14, 2024; accepted December 16, 2024. Date of publication December 18, 2024.
This paper available online at www.ijprse.com
ISSN (Online): 2582-7898; SJIF: 5.59

shifting and word-shifting are generally harder to detect but have less integrity (Roy & Manasmita, 2011). This means that bits in the encoded information are likely to be lost during transmission, losing information in the process.Content-based methods mainly use Natural Language Processing to embed information within text as lexical, syntactic, or semantic modifications. They can be more reliable in terms of data transmission since they can be passed through informational channels as raw unformatted text. A study by Yang et al. (2018) demonstrates a novel approach to this by applying a statistical model called a Markov Chain to describe the transitional probabilities of a given text corpus and then using Huffman Coding to traverse the model and generate natural language that will act as the cover media for the encoded text.Securing legal documents is paramount in today's digital age, where sensitive information can easily fall into the wrong hands. Steganography, specifically linguistic steganography, offers a unique solution for enhancing the security of legal documents by concealing them within ordinary text files. This method adds an extra layer of protection by hiding the existence of the documents themselves, making them less susceptible to unauthorized access or interception. However, most steganographic text generation techniques can very easily inflate the size of the encoded bitstream which can make it hard to both process and store larger documents. A steganographic algorithm that maximizes the embedding rate of the bitstream would be beneficial for scenarios where a large document is needed to be stored securely.

2. Methodology

The proposed system architecture for the Markov Chain-based linguistic steganography system is developed using Django as its primary web framework to ensure a modular, scalable, and efficient infrastructure. The architecture is divided into three main layers: the presentation layer, the application layer, and the database layer. The presentation layer comprises the frontend, constructed using HTML, CSS, JavaScript, and Django Templates, with Bootstrap employed for responsive design. This layer provides an intuitive user interface where users can upload documents, monitor encoding and decoding progress, and view results through a dashboard. It also processes user inputs for algorithm parameter adjustments and displays dynamic data received from the backend. The application layer, powered by Django, is designed as a collection of modular components organized into view controllers. Key controllers include the User Management Controller for handling user authentication, the Document Management Controller for managing uploaded documents, the Encoding Controller for text steganography encoding, the Decoding Controller for extracting data, the Progress Status Controller for tracking encoding/decoding progress, and the Home Controller for managing the main dashboard view. The encoding process leverages enhanced Markov Chain-based algorithms with binary encoding and key injection mechanisms to produce steganographic text, while decoding traverses the Markov Chain to validate and reconstruct the original bitstream. The database layer uses an SQL database to securely store user credentials, uploaded documents, and logs of system operations. It also manages the training corpora for Markov models required for text generation. Secure storage mechanisms are implemented to ensure data integrity for uploaded files and generated outputs. This architecture ensures that the system is modular, scalable, and capable of effectively managing user interactions, encoding/decoding processes, and secure data storage.

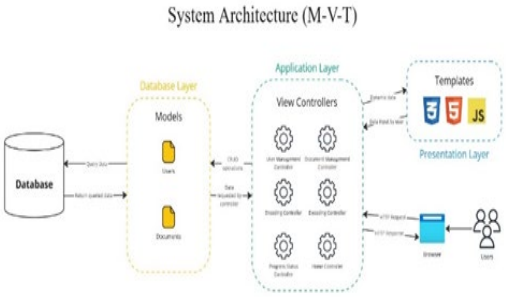
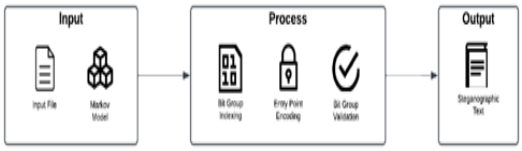


Fig.1. System Architecture

3. Markov Chain Generation Algorithm



A. Encoding Algorithm

1. Input secret bit stream.
2. Initialize entry point (keyword) list.
3. Index an entry point using the integer equivalent of the next bit group in the bit stream.
4. While it's not the end of the current sentence:
   - Query the transitional matrix of the current n-gram from the constructed Markov Chain.
   - Sort the possible transitions based on weight.
   - Index a transition using the integer equivalent of the next bit group in the bit stream. Always index end points if they exist.
   - Output the indexed transition and construct a new n-gram
   - If at the end of the current sentence: Index an entry point using the integer equivalent of the next bit group in the bit stream.
   - If at the last bit group: Set key as the length of the

bit group.
- Convert key to ASCII character starting from 97 (‘a’).
- Append character to random word in the output.

5. Return generated text.

B. Decoding Algorithm

1. Input generated text.
2. For n-gram in generated text:
   - Query the transitional matrix of the current n-gram from the constructed Markov Chain.
   - Sort the possible transitions based on weight.
   - Calculate the max bit length of the encoded index based on the length of the list of possible transitions.
   - Find the index of the next word and convert it into binary with trailing zeroes to satisfy the max bit length.
   - If the next word doesn’t exist in the transitions, convert its last character to its ASCII decimal equivalent and set it as the end key.
   - Output the decoded index in binary. If at last word: Use the end key to determine the length of the last bit group.
3. Return the generated bit stream.

4. Results And Discussion

The proposed enhancements to the Markov Chain-based linguistic steganography algorithm focus on improving three critical aspects: processing speed, embedding rate, and validity. These modifications address limitations of the traditional algorithm, ensuring efficiency, reliability, and scalability for secure steganographic text generation.

A. Processing Speed Optimization

To improve computational efficiency, the enhanced algorithm eliminates the need for Huffman tree construction by directly encoding bit groups as integer indices. This approach reduces the time complexity of encoding and decoding processes to O (1), enabling faster handling of large bitstreams. Instead of generating a Huffman tree for every transitional matrix, the algorithm leverages direct indexing to significantly lower overhead. Simulations show up to a 54-fold increase in processing speed, particularly in scenarios involving longer bitstreams and higher Markov model state sizes.

B. Embedding Rate Enhancement

The algorithm optimizes data embedding by utilizing entry points as bit carriers and ensuring efficient transitions within the Markov Chain. Unlike the traditional method that randomly selects entry points, the enhanced algorithm indexes entry points deterministically using bit groups, allowing for higher data capacity in smaller output texts. This design also ensures that transitions always lead to valid end states, maximizing data utilization. Tests conducted with varying Markov model state sizes demonstrated a marked increase in embedding rates, particularly for state sizes of 2 and 3, while maintaining compact and natural-looking output.

C. Validity Improvements

To address errors in encoding and decoding, the enhanced algorithm implements a validation mechanism for incomplete bit groups. It encodes the length of the last bit group as a key embedded in the output text, ensuring accurate decoding of all bitstreams regardless of size or content. This approach eliminates the issue of invalid leaf nodes caused by insufficient bitstream lengths. Simulations confirmed a 100% success rate for encoding and decoding operations, compared to a 4% success rate in the traditional algorithm, establishing the enhanced version as a more reliable tool for secure steganographic text generation.

D. Conclusion

The enhancements introduced to the Markov Chain-based linguistic steganography algorithm effectively address the core challenges of traditional methods, significantly improving processing speed, embedding rate, and validity. By eliminating Huffman tree dependencies, leveraging deterministic entry points, and incorporating a validation mechanism for incomplete bit groups, the enhanced algorithm ensures efficiency, scalability, and reliability. These advancements establish the algorithm as a robust and practical solution for secure text-based steganography, with potential applications in sensitive data protection and privacy-critical domains.

5. Conclusion

This study successfully enhanced the Markov Chain-based linguistic steganography algorithm by addressing three key challenges: computational efficiency, embedding rate, and validity. The transition from Huffman tree construction to direct binary integer encoding significantly improved processing speed, achieving up to 54 times faster performance for longer bitstreams. Embedding rates were optimized by utilizing entry points as bit carriers, allowing the algorithm to encode more data within smaller outputs. Finally, the introduction of a validation mechanism ensured 100% decoding accuracy by addressing issues with incomplete bit groups. These enhancements establish the algorithm as a reliable and efficient tool for secure text-based steganography.

References

[1] Al-Dmour, H. & Al-Ani, A., 2016. A steganography embedding method based on edge identification and XOR coding. Expert Systems with Applications, 46, pp.293–306.
[2] Awati, R., 2022. What is binary? - Definition from WhatIs.com. WhatIs.com.
[3] Davies, N., 2022. How to Calculate Algorithm Efficiency. KDnuggets.

[4] Gurunath, R., Alahmadi, A., Samanta, D., Khan, M. & Alahmadi, A., 2021. A novel approach for linguistic steganography evaluation based on artificial neural networks. IEEE Xplore.
[5] Huffman, D., 1952. A Method for the Construction of Minimum-Redundancy Codes. Proceedings of the IRE, 40(9), pp.1098–1101.
[6] Liu, L., Tang, L. & Zheng, W., 2022. Lossless image steganography based on invertible neural networks. Entropy, 24(12), p.1762.
[7] Lockwood, R. & Curran, K., 2017. Text-based steganography. International Journal of Information Privacy, Security and Integrity, 3(2), p.134.
[8] Luo, Y., Li, F. & Chang, C., 2016. Text steganography based on Ci-poetry generation using Markov chain model. KSII Transactions on Internet and Information Systems, 10(9).
[9] Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y. & Potts, C., 2011. Learning Word Vectors for Sentiment Analysis. The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).
[10] Mishra, R. & Bhanodiya, P., 2015. A review on steganography and cryptography. IEEE Xplore.
[11] Moffat, A., 2019. Huffman coding. ACM Computing Surveys, 52(4), pp.1–35.
[12] Nick, H., 2020. How Markovify works. Napsterinblue.github.io.
[13] Payong, A., 2024. Baeldung on computer science. Baeldung.
[14] Ribeiro, D., 2024. Markov Chains: A Comprehensive Guide to Stochastic Processes and the Chapman-Kolmogorov Equation. Data and beyond. 19 February.
[15] Roy, S. & Manasmita, M., 2011. A novel approach to format-based text steganography. Proceedings of the 2011 International Conference on Communication, Computing & Security - ICCCS '11.
[16] Umut Topkara, M., Mercan Topkara, M. & Atallah, M.J., 2006. The hiding virtues of ambiguity. ACM Workshop on Multimedia and Security.