Enhancing Contextual Compatibility of Textual Steganography Systems Based on Large Language Models

Nasouh AlOlabi
Higher Institute for Applied Sciences and Technology
Damascus, Syria

Riad Sonbol
Higher Institute for Applied Sciences and Technology
Damascus, Syria

ABSTRACT

This study presents a systematic literature review on linguistic steganography, with a particular focus on the transformative impact of Large Language Models (LLMs). We focus on the evolution of linguistic steganography within the Large Language Model (LLM) era. Our findings highlight that LLM-based approaches significantly enhance imperceptibility, embedding capacity, and naturalness in cover text generation, addressing long-standing challenges. The review underscores the shift towards more sophisticated, context-aware, and robust steganographic systems, paving the way for future research in secure and imperceptible covert communication. This study highlights the critical role of context and domain knowledge in advancing linguistic steganography, particularly in generative text steganography, where the goal is to achieve both perceptual and statistical imperceptibility. We observe how understanding the communicative context and leveraging domain-specific correlations between texts are crucial for overcoming limitations such as low embedding capacity and cognitive imperceptibility, ultimately leading to more secure and effective covert communication.

KEYWORDS

Systematic Literature Review, Linguistic Steganography, Large Language Models, LLMs, Natural Language Processing, NLP, Software Engineering

1 INTRODUCTION

This review explores how large language models (LLMs) are transforming linguistic steganography, the practice of hiding messages in text. We focus on the unique challenges and advances in using LLMs for secure, imperceptible, and high-capacity covert communication.

1.1 Overview of Information Security and Concealment Systems

Information security systems include **encryption**, **privacy**, and **concealment** (steganography).

1.1.1 Encryption Systems and Privacy Systems. These protect content but reveal that secret communication is happening, which can attract attention.

1.1.2 Concealment Systems (Steganography). Steganography hides the existence of information by embedding it in ordinary carriers (e.g., text, images). Text is a challenging carrier due to its low redundancy and strict semantics.

1.2 Introduction to Steganography

Steganography is often explained by the "Prisoners' Problem," where Alice and Bob must communicate secretly under surveillance. The goal is to embed messages so they are undetectable to an observer.

Steganography methods include **carrier selection**, **carrier modification**, and **carrier generation**.

- **Carrier modification:** Hide information in existing text with minimal changes.
- **Carrier generation:** Generate new text that encodes information, allowing higher capacity but requiring naturalness.

1.3 The Significance of Linguistic Steganography

Linguistic steganography enables covert communication, especially where encryption is suspicious. Text is a robust, ubiquitous carrier but presents challenges in balancing imperceptibility and capacity. Advances in deep learning and LLMs improve text quality and security, while related fields like watermarking focus on tracing content origin.

1.4 Scope of the Review

This review covers LLM-based linguistic steganography, focusing on methods, evaluation, challenges, and future directions.

2 STEGANOGRAPHY AND LARGE LANGUAGE MODELS

Large Language Models (LLMs) have emerged as a significant development in the field of natural language processing, profoundly impacting text generation and related applications like steganography and watermarking. Here's a breakdown of their emergence and impact:

2.1 Capabilities and Approximating Natural Communication

LLMs are **generative models** that can **approximate complex distributions like text-based communication**. They represent the best-known technique for this task. These models operate by taking context and parameters to output an explicit probability distribution over the next token (e.g., a character or a word). The next token is typically sampled randomly from this distribution, and the process repeats to generate output of a desired length. Training LLMs involves processing vast amounts of data to set parameters and structure, enabling their output distributions to approximate true distributions in the training data. The **quality of content generated by generative models is impressive** and continues

to improve. This has led to LLMs blurring the boundary of high-quality text generation between humans and machines. LLMs are increasingly used to generate data for specific tasks, such as tabular data, relational triples, sentence pairs, and instruction data, often achieving satisfactory generation quality in zero-shot learning for specific subject categories. They have also shown capabilities in mimicking language styles and semantics, and their generalization ability allows them to comprehend the semantics of context.

2.2 Role in Generative Linguistic Steganography

LLMs are considered **favorable for generative text steganography** due to their ability to generate high-quality text. Researchers propose using generative models as steganographic samplers to embed messages into realistic communication distributions, such as text. This is a departure from prior steganographic work and is motivated by the public availability of high-quality models and significant efficiency gains. LLMs like **GPT-2, LLaMA, and Baichuan2** are commonly used as basic generative models for steganography. Existing methods often use a language model and steganographic mapping, where secret messages are embedded by establishing a mapping between binary bits and the sampling probability of words within the training vocabulary. However, traditional "white-box" methods require sharing the exact language model and training vocabulary, which limits fluency, logic, and diversity compared to natural texts generated by LLMs. They also inevitably change the sampling probability distribution, posing security risks. New approaches, like **LLM-Stega**, explore **black-box generative text steganography using the user interfaces (UIs) of LLMs**, overcoming the need to access internal sampling distributions. This method constructs a keyword set and uses an encrypted steganographic mapping for embedding, proposing an optimization mechanism based on reject sampling for accurate extraction and rich semantics [9]. Another framework, **Co-Stega**, leverages LLMs to address the low capacity challenge in social media by increasing the text space for hiding messages (through context retrieval) and **raising the generated text's entropy via specific prompts** to increase embedding capacity. This approach also aims to maintain text quality, fluency, and relevance [4]. The concept of **zero-shot linguistic steganography** with LLMs utilizes in-context learning, where samples of covertext are used as context to generate more intelligible stegotext using a question-answer (QA) paradigm [5]. LLMs are also used in approaches like **ALiSa**, which directly conceals token-level secret messages in seemingly natural steganographic text generated by off-the-shelf BERT models equipped with Gibbs sampling [11]. The increasing popularity of deep generative models has made it feasible for provably secure steganography to be applied in real-world scenarios, as they fulfill requirements for perfect samplers and explicit data distributions [2, 3, 6].

2.3 LLM-Based Steganography Models

2.3.1 *Evaluation Metrics.*

Imperceptibility Metrics. Perceptual: PPL, Distinct-n, MAUVE, human evaluation. Statistical: KLD, JSD, anti-steganalysis accuracy, semantic similarity.

Embedding Capacity Metrics. Bits per token/word, embedding rate.

2.4 Challenges and Limitations in Steganography with LLMs

2.4.1 Perceptual vs. Statistical Imperceptibility (Psic Effect). Improving perceptual quality can reduce statistical security, and vice versa.

2.4.2 Low Embedding Capacity. Short texts and strict semantics limit how much information can be hidden.

2.4.3 Lack of Semantic Control and Contextual Consistency. Ensuring generated text matches intended meaning/context is difficult.

2.4.4 Challenges with LLMs in Steganography. LLMs may introduce unpredictability, bias, or leak information.

2.4.5 Segmentation Ambiguity. Tokenization can cause ambiguity in how information is embedded or extracted.

A primary challenge in steganography, particularly when utilizing Large Language Models (LLMs), revolves around the **distinction between white-box and black-box access**. Most current advanced generative text steganographic methods operate under a "white-box" paradigm, meaning they require direct access to the LLM's internal components, such as its training vocabulary and the sampling probabilities of words. This presents a significant limitation because many state-of-the-art LLMs are proprietary and are accessed by users primarily through black-box APIs or user interfaces [9]. Consequently, these white-box methods are often impractical for real-world deployment with popular commercial LLMs. Furthermore, methods that rely on modifying the sampling probability distribution to embed secret messages inherently introduce security risks because they alter the original distribution, making the steganographic text statistically distinguishable from normal text [2, 3, 9, 10].

Another significant hurdle is **ensuring both the quality and imperceptibility of the generated text**, encompassing perceptual, statistical, and cognitive imperceptibility. While advancements in deep neural networks have improved text fluency and embedding capacity, older models or certain embedding strategies can still produce texts that lack naturalness, logical coherence, or diversity compared to human-written content. Linguistic steganography methods often struggle to control the semantics and contextual characteristics of the generated text, leading to a decline in its "cognitive-imperceptibility" [1, 10]. This can make concealed messages easier for human or machine supervisors to detect. Although models like NMT-Stega and Hi-Stega aim to maintain semantic and contextual consistency by leveraging source texts or social media contexts, this remains a complex challenge [1, 8].

Channel entropy requirements and variability also pose a considerable challenge. Traditional universal steganographic schemes often demand that the communication channel maintains a minimum level of entropy, which is rarely consistent in real-world communication, especially in natural language. Moments of low or zero entropy can cause existing steganographic protocols to fail or necessitate the generation of extraordinarily long steganographic texts, making covert communication impractical. While schemes like Meteor attempt to adapt by fluidly changing the encoding rate

proportional to instantaneous entropy, overcoming this variability without increasing detectability is difficult. The "Psic Effect" (Perceptual-Statistical Imperceptibility Conflict Effect) highlights this dilemma, where optimizing for perceived quality might compromise statistical imperceptibility and vice-versa.

Furthermore, **segmentation ambiguity** introduced by subword-based language models, commonly used in high-performing Transformer architectures, presents a critical issue for provably secure linguistic steganography. When a sender detokenizes generated subword sequences into a continuous text (e.g., "any" + "thing" becoming "anything") before transmission, the receiver might retokenize it differently (e.g., as a single "anything" token), leading to decoding errors and affecting subsequent probability distributions. Existing disambiguation solutions typically involve modifying the token candidate pool or probability distributions, which renders them incompatible with the strict requirements of provably secure steganography that demand unchanged distributions [6]. While SyncPool attempts to address this without altering the distribution, it may still lead to a reduction in the embedding rate due to information loss [6].

Additional limitations include: * **Computational Overhead**: LLMs, while powerful, incur a higher computational cost (3-5 times more than prior methods), which could impact real-time communication scenarios [5]. * **Data Integrity and Reversibility**: Some linguistic steganography methods are not reversible, meaning the original cover text cannot be perfectly recovered after message extraction, which is undesirable for sensitive applications [7, 12]. Text data is generally less prone to lossy compression issues than other media, but incompleteness of the steganographic text can still damage the embedded bitstream [5]. * **Ethical Concerns**: The use of pre-trained LLMs may inadvertently introduce ethical issues such as political biases, gender discrimination, or the generation of insulting content [5]. * **Provable Security and Rigor**: Despite decades of research into provably secure steganography, practical systems have been hampered by strict requirements like perfect samplers and explicit data distributions [2, 3]. Many works from the NLP community, while generating convincing text, often lack rigorous security analyses and fail to meet formal cryptographic definitions, making them vulnerable to detection [3].

Despite their capabilities, generative models are still **far from perfect** in imitating real communication. A significant challenge for practical steganography is the difficulty of finding samplers for non-trivial distributions like the English language, which continues to evolve. When using approximate samplers, there's a risk that an adversary can detect a steganographic message by distinguishing between the real channel and the approximation [3]. LLMs are known to make mistakes, including "hallucinations," which can lead to errors and erratic embedding during text generation, especially for long stego sequences. One critical issue is **segmentation ambiguity** in neural linguistic steganography. LLMs often use **subword tokenization**, meaning a single text can correspond to multiple token representations. If the sender and receiver have different understandings of segmentation, it can lead to incorrect message extraction and affect subsequent generation steps. Current provably secure methods have largely overlooked this. SyncPool is a proposed method to address this by grouping tokens with prefix relationships in the candidate pool without altering the original probability distribution. The **computational overhead of LLMs is higher** compared to prior methods (approximately 3x to 5x), potentially limiting real-time communication. The effectiveness of LLM-based steganography can be limited by the **entropy of the cover text** in social media contexts, as short, context-dependent replies have lower entropy, thus limiting hiding capacity [4].

3 LITERATURE REVIEW METHODOLOGY

3.1 Research questions

Here are the research questions addressed in this SLR:

- What is the state of published literature on steganographic techniques that leverage large language models (LLMs)?
- In which applications are steganographic techniques with LLMs being explored?
- What metrics and evaluation methods are used to assess the performance of steganographic techniques in LLMs, focusing on factors like capacity, security, and contextual compatibility?
- How are external knowledge sources (semantic resources) integrated into steganographic techniques with LLMs to enhance capacity or contextual relevance?
- What are the limitations and trade-offs associated with current steganographic techniques using LLMs, particularly concerning security, capacity, and contextual compatibility?
- What are the potential future research directions in steganography with LLMs, considering emerging trends and identified gaps in the literature?

3.2 Search query string

We used the following search query string for our initial literature search:

```
(steganography or watermark or "Information Hiding")
 and ("Large Language Model" or LLM or BERT or LAMA or GPT)
```

3.3 Study selection and quality assessment

We established the following inclusion and exclusion criteria for study selection:

3.3.1 *Inclusion Criteria.*

- **Full Text Access**: Studies for which the full text is available.
- **Language**: Publications written in English.
- **Peer-reviewed**: Articles published in peer-reviewed journals, conferences, or workshops.
- **Publication Date**: Studies published from 2018 onwards, to focus on recent advancements in LLMs.
- **Relevance**: Studies directly addressing steganography, watermarking, or information hiding techniques that utilize or are significantly impacted by Large Language Models (LLMs), BERT, LAMA, or GPT architectures.
- **Research Type**: Empirical studies, surveys, reviews, and theoretical contributions.

3.3.2 *Exclusion Criteria.*

- **Duplicated Studies**: Multiple publications reporting the same study will be excluded, with the most complete or recent version retained.

- **Incomplete or Abstract-only**: Studies for which only an abstract is available or the full text is incomplete.
- **Irrelevant Studies**: Publications not directly related to steganography with LLMs.
- **Non-English Publications**: Studies not published in English.
- **Non-peer-reviewed Sources**: Preprints, dissertations, theses, books, and book chapters (unless they are extended versions of peer-reviewed conference papers).

3.4 Bibliometric analysis

Briefly note if snowballing was used for additional sources.

4 CONDUCTING THE SEARCH

This section details the systematic process followed to identify and select relevant literature for this review. The search strategy was designed to ensure comprehensive coverage of the topic while adhering to predefined inclusion and exclusion criteria.

4.1 Initial Candidate Papers

Our initial automated search across selected digital libraries yielded a total of 1043 candidate papers. The distribution of these papers by source was as follows: ACM Digital Library (346), IEEE Digital Library (61), Science@Direct (209), Scopus (151), and Springer Link (276). This stage focused on broad keyword matching to capture all potentially relevant studies.

4.2 Duplicate Removal

Following the initial search, a rigorous process of duplicate removal was undertaken. After removing duplicates, 989 papers remained. This involved both automated tools and manual verification to ensure that each unique paper was considered only once, thereby streamlining the subsequent screening stages.

4.3 Multi-stage Filtering

The identified papers underwent a multi-stage filtering process based on their titles, abstracts, and full texts. After title and abstract filtering, 58 papers remained. Of these, 18 were accepted with PDFs available, and 14 are pending PDF acquisition. This systematic approach, guided by our predefined inclusion and exclusion criteria, progressively narrowed down the selection to the most pertinent studies.

4.4 Snowballing

To complement the automated search and ensure no critical papers were missed, a snowballing technique was applied. This involved examining the reference lists of included studies and identifying papers that met our selection criteria, further enriching our dataset.

4.5 Research Questions

Our systematic literature review is guided by the following research questions:

(1) What is the state of published literature on steganographic techniques that leverage large language models (LLMs)?
(2) In which applications are steganographic techniques with LLMs being explored?
(3) What metrics and evaluation methods are used to assess the performance of steganographic techniques in LLMs, focusing on factors like capacity, security, and contextual compatibility?
(4) How are external knowledge sources (semantic resources) integrated into steganographic techniques with LLMs to enhance capacity or contextual relevance?
(5) What are the limitations and trade-offs associated with current steganographic techniques using LLMs, particularly concerning security, capacity, and contextual compatibility?
(6) What are the potential future research directions in steganography with LLMs, considering emerging trends and identified gaps in the literature?

5 DATA EXTRACTION AND CLASSIFICATION

This section outlines the methodology employed for extracting and classifying data from the selected primary studies. A structured approach was adopted to ensure consistency and accuracy in data collection, facilitating a comprehensive analysis of the literature.

5.1 Data Extraction Form (DEF) Content

A Data Extraction Form (DEF) was developed to systematically collect relevant information from each primary study. The DEF was designed to capture key details necessary to answer the research questions, including:

- **Bibliometric Information:** Author(s), publication year, type of publication (e.g., journal, conference proceeding), and publication venue.
- **Targeted RE Task(s):** The specific Requirements Engineering tasks addressed by the study.
- **Proposed Solution:** Details of the approach, including the syntactic and semantic information used to represent requirements.
- **Evaluation Details:** Information regarding the evaluation dataset, metrics used, and the reported results.
- **Limitations and Constraints:** Any identified limitations or constraints of the proposed approach.

5.2 Data Classification

Following data extraction, studies were classified based on predefined categories derived from our research questions. This classification aimed to group similar studies and identify trends, patterns, and gaps in the existing literature, providing a structured overview of the research landscape.

6 DATA SYNTHESIS

This section describes the process of synthesizing the extracted and classified data to answer the research questions. The synthesis involved aggregating findings, identifying common themes, and analyzing patterns across the primary studies.

6.1 Presentation of Results

The results of the data synthesis are presented in a structured manner, often utilizing tables, figures, and descriptive statistics to

summarize key findings. This includes an overview of publication trends, distribution of studies across different categories, and the prevalence of various approaches and techniques.

6.2 Discussion in Relation to Research Questions

Each research question is addressed individually, with a detailed discussion of the synthesized data. This involves interpreting the findings, highlighting significant observations, and drawing conclusions based on the evidence gathered from the primary studies. The discussion also identifies areas where further research is needed and potential future directions.

7 RESULTS

8 LLM-BASED STEGANOGRAPHY APPROACHES

This section summarizes the main findings from the systematic literature review, focusing on the characteristics and performance of various LLM-based linguistic steganography and watermarking models.

8.1 LLM-Based Steganography Models

Our review identified several key LLM-based steganography models, each with unique approaches, strengths, and performance metrics. These models are discussed in terms of their underlying techniques, embedding capacities, and imperceptibility metrics, providing a comprehensive overview of the current landscape.

9 CONTEXT AWARENESS

Linguistic steganography has evolved from early methods to advanced deep learning models and Large Language Models (LLMs). This progression focuses on improving imperceptibility, embedding capacity, and maintaining naturalness and semantic coherence in cover text.

9.1 Context life cycle

9.2 Context-awareness in Steganography

Here's an overview of this evolution:

- **Early and Format-Based Approaches** Early steganography modified existing carriers, like using whitespace or linguistic idiosyncrasies (e.g., synonym substitution). These methods, often rule-based and context-neglecting, resulted in unnatural text and limited capacity (typically <1 BPT), making them easily detectable.
- **Transition to Carrier Generation and Early Text Generation Models** A significant shift involved "carrier generation based steganography," where the carrier text is generated to hide information, allowing greater freedom and higher embedding rates without altering original carrier statistics.
  - **Syntax Rules and Statistical Methods:** Early text generation for steganography, using syntax rules or statistical models like Markov models, produced easily recognizable texts, failing imperceptibility and security.
  - **Challenges of Early Generation:** The Psic Effect (perceptual-imperceptibility vs. statistical-imperceptibility conflict) was a key challenge. Generated text, though natural-looking, often had detectable statistical properties. Models also lacked semantic control, crucial for covert communication.
- **The Era of Custom Artificial Neural Network (ANN) Models** Advancements in ANNs and NLP led to sophisticated models for automatic steganographic text generation. These neural networks learned language models, encoding secret information by manipulating word probability distributions during generation.

Table 1: Summary of Results from Reviewed Papers

Paper	Result
1	PPL: 28.879, ΔMP: 0.242, KLD: 3.302, JSD: 10.411, Acc: 0.600, R: 0.616
2	BPW=0.5335 F1=0.9402 PPL=134.2199
3	SR1: 60.87%, SR2: 98.55%, Gen. Capacity: 44.91 bits, Entropy: 49.21 bits, BPW: 2.31, PPL: 16.75, SimCSE: 0.69
4	PPL=13.917 KLD=2.904 SIM=0.812 ER=0.365 (BN=2) Best Acc=0.575 (BERT classifier) FLOPs=1.834G
5	p=1.00 Total Time (seconds)=362.63 Ave Time ↓ (seconds/bit)=6.29E-03 Ave KLD ↓ (bits/token)=0 Max KLD ↓ (bits/token)=0 Capacity (bits/token)=5.76 Entropy (bits/token)=6.08 Utilization ↑=0.95 Text Generation (FCN): 50.10%. Text Generation (R-BiLSTM-C): 50.45%. Text Generation (BiLSTM-Dense): 49.95%
6	Length: 13.333 (words). BPW: 5.93 bpw PPL: 165.76. Semantic Similarity (SS): 0.5881 LS-CNN Acc: 51.55%. BiLSTM-Dense Acc: 49.20%. Bert-FT Acc: 50.00%. KLD (Log, lower is better): 2.02 .
7	GPT-2: 3.09 bits/token
8	PPL: 8.81. JSDfull: 17.90 (x10ˆ-2). JSDhalf: 16.86 (x10ˆ-2). JSDzero: 13.40 (x10ˆ-2) TS-BiRNN: 80.29%. R-BiLSTM-C: 84.34%. BERT-C: 89.61%
9	Total Error: 0%, Ave KLD: 0, Max KLD: 0, Ave PPL: 3.19 (EN), 7.49 (ZH), Capacity: 1.03–3.05 bits/token, Utilization: 0.66–0.74, Ave Time: 4μs/bit
10	Bit acc: 0.994 (K=None), 1.000 (DAE), 0.978 (Adaptive+K=S); Meteor Drop: 0.057; SBERT ↑: 1.227; Ownership Rate: 1.0 (no attack), 0.978 (adaptive+K=S)
11	BLEU: 30.5, PPL: 22.5, ER: 0.29, KL: 0.02, SIM: 0.86, Stego detection 16%
12	100% accuracy (multi-synonym, 10-sentence), mSMS: 0.9892, TPR: 0.83, FNR: 0.17, Detection: 0.00188s, Insertion: 0.27931s
13	ppl: 109.60, MAUVE: 0.2051, ER2: 10.42, Δ(cosine): 0.0088, Δ(simcse): 0.0191
14	Classifier Accuracy: 0.9880; Loop Count: 1.0160; PPL: 13.9565; Anti-Steganalysis Accuracy: 0.5
15	[Not specified]
[7]	LS07 P@1: 58.3, GAP: 65.1; CoInCo P@1: 62.6, GAP: 60.7; Text Recoverability: 88–90%
17	BPTS: 4.0, BPTC+S: 4.0, PPL: 62.1, Mean: 44.4, Variance: 2.1e04, Acc: 8.9%
18	PPL: Natural = 13.91, ALiSa = 14.85; LS-RNN/LS-BERT Acc & F1 = 0.50; Outperforms GPT-AC/ADG in all cases

Table 2: Models and Datasets Used in Reviewed Papers

Paper	Llm	Dataset
1	BERTBASE (BERT-LSTM) (LSTM-LSTM) model was trained from scratch	Twitter (2.6M sentences) IMDB (1.2M sentences) preprocessed
2	BERTBase	BookCorpus
3	Llama-2-7B-chat, GPT-2 (fine-tuned), Llama-2-13B	Tweet dataset (for GPT-2 fine-tuning), Twitter (real-time testing)
4	LSTM + attention for temporal context. GAT for spatial token relationships. BERT MLM for deep semantic context in substitution.	OPUS
5	GPT-2	IMDB
6	Any	[Not specified]
7	GPT-2	Hutter Prize, HTTP GET requests
8	LLaMA2-Chat-7B (as the stegotext generator / QA model). GPT-2 (for NLS baseline and JSD evaluation)	IMDB, Twitter
9	LLaMA2-7b (English), Baichuan2-7b (Chinese)	IMDb dataset (100 texts/sample, 3 English sentences + Chinese translations)
10	Transformer-based encoder/decoder; BERT for distillation	Web Transformer 2
11	BERT (encoder), LSTM (decoder)	WMT18 News Commentary (train/test), Yang et al. bits, Doc2Vec, 5,000 stego pairs (8:1:1 split)
12	Model-independent; tested with OPT-2.7B	Dolly ChatGPT (train/validate), C4 (test), robustness & sentence-level test sets
13	GPT-2	Yahoo! News (titles, bodies, comments); 2,400 titles used
14	CTRL (generation), BERT (semantic classifier)	5,000 CTRL-generated texts per semanteme (n = 2–16); 1,000 user-generated texts for anti-steganalysis
15	[Not specified]	Custom word sets for specific topics (e.g., 16×10-word sets for music reviews)
[7]	Transformer (Paraphraser), BART (BARTScore), BERT (BLEURT, comparisons)	ParaBank2, LS07, CoInCo, Novels, WikiText-2, IMDB, NgNews
17	BART (bart-base2)	Movie, News, Tweet
18	BERT (Google's BERTBase, Uncased)	BookCorpus (10,000 natural texts for evaluation)

10 DISCUSSION

Discuss implications and interpretation of the results.

11 CONCLUSION

Summarize the main findings and takeaways of the study.

REFERENCES

[1] Changhao Ding, Zhangjie Fu, Zhongliang Yang, Qi Yu, Daqiu Li, and Yongfeng Huang. 2023. Context-aware linguistic steganography model based on neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 32 (2023), 868–878.

[2] Jinyang Ding, Kejiang Chen, Yaofei Wang, Na Zhao, Weiming Zhang, and Nenghai Yu. 2023. Discop: Provably secure steganography in practice based on" distribution copies". In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2238–2255.

[3] Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin. 2021. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 1529–1548.

[4] Guorui Liao, Jinshuai Yang, Kaiyi Pang, and Yongfeng Huang. 2024. Co-stega: Collaborative linguistic steganography for the low capacity challenge in social media. In *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security*. 7–12.

[5] Ke Lin, Yiyang Luo, Zijian Zhang, and Ping Luo. 2024. Zero-shot generative linguistic steganography. *arXiv preprint arXiv:2403.10856* (2024).

[6] Yuang Qi, Kejiang Chen, Kai Zeng, Weiming Zhang, and Nenghai Yu. 2024. Provably secure disambiguating neural linguistic steganography. *IEEE Transactions on Dependable and Secure Computing* (2024).

[7] Jipeng Qiang, Shiyu Zhu, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2023. Natural language watermarking via paraphraser-based lexical substitution. *Artificial Intelligence* 317 (2023), 103859.

[8] Huili Wang, Zhongliang Yang, Jinshuai Yang, Yue Gao, and Yongfeng Huang. 2023. Hi-stega: A hierarchical linguistic steganography framework combining retrieval and generation. In *International Conference on Neural Information Processing*. Springer, 41–54.

[9] Jiaxuan Wu, Zhengxian Wu, Yiming Xue, Juan Wen, and Wanli Peng. 2024. Generative text steganography with large language model. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 10345–10353.

[10] Zhong-Liang Yang, Si-Yu Zhang, Yu-Ting Hu, Zhi-Wen Hu, and Yong-Feng Huang. 2020. VAE-Stega: linguistic steganography based on variational auto-encoder. *IEEE Transactions on Information Forensics and Security* 16 (2020), 880–895.

[11] Biao Yi, Hanzhou Wu, Guorui Feng, and Xinpeng Zhang. 2022. ALiSa: Acrostic linguistic steganography based on BERT and Gibbs sampling. *IEEE Signal Processing Letters* 29 (2022), 687–691.

[12] Xiaoyan Zheng, Yurun Fang, and Hanzhou Wu. 2022. General framework for reversible data hiding in texts based on masked language modeling. In *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 1–6.

Table 3: Context-Related Fields in Reviewed Papers

Paper	Context Aware	Categ Context	Representation Context	Context Usage In Method Detail Text
1	non-explicit	pre-text	text	[Not specified]
2	non-explicit	pre-text	text	[Not specified]
3	explicit	Social Media	text	Context is used for embedding and generation: the retrieval module embeds part of the message by selecting posts matching the query, and the generation module encodes the rest in replies conditioned on that context to balance capacity and relevance.
4	explicit	pre-text	text	[Not specified]
5	non-explicit	tuning + pretext	text	[Not specified]
6	explicit	[Not specified]	[Not specified]	[Not specified]
7	non-explicit	tuning + pretext	text	[Not specified]
8	explicit	zero-shot + prompt	text	[Not specified]
9	non-explicit	pretext	text	pre-text
10	Yes; semantic-level embedding; synonym substitution using BERT	Yes; watermark message assigned categorical label (e.g., 4-bit $\rightarrow$ 1-of-16)	Yes; semantic embeddings via transformer encoder and BERT; SBERT distance as metric	[Not specified]
11	Yes	[Not specified]	GCF (global context), LMR (language model reference), Multi-head attention	[Not specified]
12	NO	[Not specified]	[Not specified]	[Not specified]
13	explicit	Social Media	Text	The method retrieves a context data carrier from a social corpus, using it as input to the language model. This context embeds some secret content naturally, while a Prompt Learning Module ensures template-based, semantically coherent stegotext generation. This boosts realism and imperceptibility in social settings.
14	implicit	Text	Semanteme (α) as a vector in semantic spac	The semanteme (α) shapes the generated text to match a topic, ensuring it appears natural and blends into regular communication. This contextual alignment helps conceal the message and enables the classifier to recover it using the same semantic grounding.
15	Explicit	Specific Genre/Topic Text	Text	The system uses the topic as core context. The Key Generator builds keyword sets typical of the topic, and the Embedder ensures generated text appears natural and topic-appropriate. ChatGPT's contextual capabilities maintain coherence while embedding messages seamlessly.
[7]	Explicit	[Not specified]	text	ParaLS uses context to generate semantically appropriate substitutions. Decoder predictions begin with prefix words, and attention uses cumulative source weights. Semantic similarity scores rank substitutes, ensuring watermarked sentences retain meaning. Context ensures reproducibility of embedding and extraction through identical candidate sets.
17	not Explicit	[Not specified]	[Not specified]	[Not specified]
18	No	[Not specified]	[Not specified]	[Not specified]

Table 4: Categories, Main Strengths, and Weaknesses of Reviewed Papers

Paper	Type	Main Strengths	Main Weaknesses
1	Steganography	statistical fidelity	Not AutoRegressive
2	Steganography	[Not specified]	[Not specified]
3	Steganography	High capacity, fluency, semantic relevance, EES boosts entropy, ϵ-security satisfied	Context limits capacity, retrieval vulnerable to mismatch, small models limit query handling
4	Steganography	[Not specified]	[Not specified]
5	Steganography	[Not specified]	[Not specified]
6	Steganography	[Not specified]	[Not specified]
7	Steganography	[Not specified]	[Not specified]
8	Steganography	[Not specified]	[Not specified]
9	Steganography	Zero decoding error, provable security, perceptual imperceptibility, multilingual support, transferable design	Reduced entropy utilization due to token prefix loss in detokenization/retokenization
10	Watermarking	[Not specified]	[Not specified]
11	Steganography	Context control, Semantic match	Old models lacked context, low capacity (waiting), probabilistic cliffs, possible meaning mismatch
12	Watermarking	Blindness, robustness, imperceptibility, high accuracy, automatic, model-independent, compact, fast, sentence-level	Training data dependence, needs pre-watermarking, limited on short/diverse texts, insertion overhead, rewrite circumvention
13	Steganography	High payload, semantic coherence, high text quality, strong resistance to steganalysis	Slightly reduced diversity, limited retrieval scope
14	Steganography	Symbol-free, efficient, imperceptible, accurate extraction, complements symbolic steganography	Classifier struggles with large n, higher loop count, semanteme text quality varies, orthogonality issues
15	Steganography	Effective LLM-based stego text, natural concealment, scalable, supports prompt iteration	NL complexity, duplication, hallucinations, errors in long texts, statistical artifacts
[7]	Watermarking	High meaning preservation, strong performance vs. BERT-based NLW, no annotated data needed, robust decoding, high text recoverability	Limited reversibility, hard for short texts, beam decoding limits substitution, lower capacity in some cases
17	Steganography	High capacity, naturalness, reduced vigilance, denoising, real-world efficiency	Similarity drops with high BPTS, may underperform in detection accuracy vs edit-based
18	Steganography	Simple	Poor Quality, low cap

Table 5: Approach/Pipeline Method Used in Reviewed Papers

Paper	Pipline Method Used
1	Input text → Encoder (LSTM/BERT) → latent vector z N(0,1) z → Decoder (LSTM) → conditional distribution → top-m Candidate Pool Candidate Pool + secret bits → Coding (Huffman/Arithmetic) → word output Repeat → sentence with hidden bits Shared z + Decoder → conditional distribution → CP reconstruction Observed word + CP → decode bits
2	Initialized text (secret words fixed, others masked) → Masked LM → conditional distribution over masked slots → secret bits guide word selection → Marked Text Marked Text = original words + data-encoded generated words → Receiver (with key) → decode secret bits + recover original text
3	Message → split into query bits + remaining bits Query bits → Search engine → retrieve matching posts Remaining bits + retrieved context → LLM (with entropy prompts) → generate reply → Reply embeds remaining bits
4	pre-text → LSTM → Temporal context (H_time) → self-attention → H_time' Temporal context (H_time) + sliding-window graph → GAT → Spatial context (H_space) GAT + H_time' + H_space → fusion → next-token distribution
5	sampling using the distribution copy
6	Keyword Prompt → LLM → 4 keyword subsets: - Subject, Predicate, Object (16 words each) - Emotion (3 words: negative, neutral, positive) → Evaluation Prompt → LLM optimizes sampling probabilities
7	Next-word distribution → partitioned into intervals → random r selects word → shared r prefix → message bits → r XOR PRG mask → secure transmission → Receiver unmasks r → decodes bits → Bit rate ∝ entropy of next word
8	Secret text → Huffman + Edge Flipping → binary bitstream → LM generates candidates (filtered by threshold τ) → Embedding Module selects word matching bitstream prefix + Annealing Selection (temperature control) + Repeat Penalty (discourages repetition) → Word output per step → encoded sentence
9	Context → token distribution prediction → ambiguous token pooling → CSPRNG + message bits select tokens from pools → detokenize → coherent stegotext
10	Original Text (S), Watermark bits (m), Side info (K) → Encoder (Transformer) → paraphrase under distortion constraints → Watermarked Text (X) Attacker (omniscient) → corrupts X → Corrupted Text (Y) Decoder (Transformer + linear) + K → estimate watermark (m) Train via rate-distortion game maximizing mutual info, minimizing distortion, with adversarial robustness
11	Context (C), Secret Message (m), Shared Key (K) → Language Model (Transformer) → next-token distribution (Pc, V) SyncPool Ambiguity Pool Grouping → ambiguous tokens grouped (V_amb), preserving original distribution Embed message via ENCODE + CSPRNGsteg → select Ambiguity Pool If ambiguous → CSPRNGsync → synchronous token selection Detokenize → Stegotext (S) → transmit Receiver DECODE + retokenization + CSPRNGsteg + CSPRNGsync → extract message bits, preserving provable security
12	Input text → Identify candidate words (exclude stopwords/punctuation) Candidates → Word2Vec → top-n semantically similar words Secret bits + candidates → Guided substitution → sentence proposals Proposals → USE → select highest similarity → Marked Text Marked Text = original + synonym substitutions (watermarked) Marked Text → BERT-based classifier → detect watermark (blind detection)
13	Secret message m → split into data info + control info Data info → retrieve carrier c from corpus C → maximize m ∩ c Control info = (keywords k = m c, positions i = m ∩ c) → format → bitstream b Bitstream b + keywords k + carrier c → Prompt Learning + Keyword Guidance → generate stegotext s s embeds keywords k + hidden bitstream b → high semantic coherence (Δcos, Δsimcse) → cognitive imperceptibility Only c or s ≠ full m → layer separation → security ↑, anti-steganalysis ↑ Embedding rate (ER2) ↑ → high payload per word with low perplexity
14	Secret message → Semantic mapping → semanteme α (discrete space, log(n)-bit) α → CTRL (Transformer) → stegotext x (conditional distribution pθ(x|α)) Stegotext x → BERT classifier → extracted semanteme α' α' ≠ α → Rejection sampling → repeat generation until α' = α Final stegotext → Bob decodes → secret message m'
15	Key Generator: Topic-specific word sets → labeled subsets (e.g., A, B) → no duplicates, single words only Message → sequence of subset labels (stegosequence, e.g., "AABBAABB") Embedder (ChatGPT 4.0): Stegosequence + topic + word sets → stego cover text
[7]	Original sentence → Paraphraser (Transformer) → paraphrases → extract substitute candidates for target word xi Novel decoding → force prefix x<i → predict vocabulary distribution for yi → top-K substitutes Substitutes → rank by Semantic Textual Similarity (BARTScore + BLEURT + prediction scores) → select best candidate Exchangeability test → ensure xi and yi are mutually interchangeable → form candidate set C = c0, c1 Watermark bit (0/1) → replace xi with csignal from C → watermarked sentence x' Extraction → mirror embedding → generate C for x' → compare xi with c0/c1 → recover watermark bit
17	Cover text (Y) + Conditional Code → BART Encoder → encoded representation Secret message (S) → bit stream → 2ˆn vocabulary groups, each with n-bit code BART Decoder + Group-Wise Masked Decoding → mask non-aligned token probabilities → stego text (Y') Beam search → K candidate stego texts GPT2 → Perplexity (PPL) scoring → select lowest PPL candidate as final stego text
18	Secret tokens → initialize stego text s with [MASK] tokens Secret plaintext w → insert tokens at positions p in s Remaining [MASK] tokens → BERT MLM → probability distribution Gibbs sampling → sample tokens from conditional distribution → fill s Iterate T times → refine non-secret positions → fluent stego text BERT + Gibbs → low Perplexity (PPL) → high fluency Conditional sampling → distribution mimics natural text → strong anti-steganalysis Receiver → extract plaintext from positions p