Auto-Stega: An Agent-Driven System for Lifelong Strategy Evolution in LLM-Based Text Steganography

Anonymous ACL submission

Abstract

With the rapid progress of LLMs, high quality generative text has become widely available as a cover for text steganography. However, prevailing methods rely on hand-crafted or pre-specified strategies and struggle to balance efficiency, imperceptibility, and security, particularly at high embedding rates. Accordingly, we propose Auto-Stega, an agent-driven self-evolving framework that is the first to realize self-evolving steganographic strategies by automatically discovering, composing, and adapting strategies at inference time; the framework operates as a closed loop of generating, evaluating, summarizing, and updating that continually curates a structured strategy library and adapts across corpora, styles, and task constraints. A decoding LLM recovers the information under the shared strategy. To handle high embedding rates, we introduce PC-DNTE, a plug-and-play algorithm that keeps sampling close to the base model's conditional distribution at high embedding rates, preserving imperceptibility while enhancing security. Experimental results demonstrate that at higher embedding rates, Auto-Stega achieves superior performance with a 42.2% reduction in normalized perplexity and a 1.2% improvement in anti-steganalysis performance over SOTA methods.

1 Introduction

Steganography is an information hiding technique that embeds secret information in cover media to enable covert communication (Liao et al., 2025; Meng et al., 2025; Zhang et al., 2025a). The fundamental objectives of steganographic systems are characterized by three key metrics: the embedding rate, which quantifies the capacity of secret information that can be hidden; perceptual concealment, typically represented by the quality of the media after information embedding (e.g., text quality or degree of image modification); and anti-steganalysis performance (Hu and Wang, 2025; Zhou et al., 2025b), which measures the capability to resist machine detection. These correspond respectively to core requirements of efficiency, imperceptibility, and security.

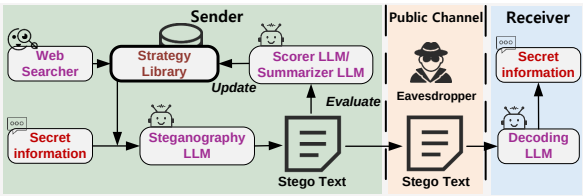

Figure 1: The pipeline of Auto-Stega.

By cover type, steganography can be categorized into text steganography (Wang et al., 2025a,b; Wu et al., 2024), image steganography (Zhou et al., 2025c; Cheng et al., 2025), audio steganography (Li et al., 2025; Zhang et al., 2025b), and video steganography (Meng et al., 2025). Among these, text steganography has emerged as an active research area in information security, driven by the prevalence of text-based social communication.

Early text steganography was predominantly based on manually designed content modifications, such as synonym substitution (Li et al., 2019), and spelling conversion (Shirali-Shahreza, 2008). Although these rule-based schemes can remain semantically unobtrusive, they usually provide a low embedding rate and induce a distributional change from natural text (Zhang et al., 2021), which compromises their resistance to machine detection and leads to inadequate anti-steganalysis performance.

Recent advances in large language models (LLMs) have made high-quality generative text broadly accessible and controllable, creating new opportunities for generative text steganography. Generative steganography (Zhou et al., 2025c; Peng et al., 2024) is a technique that embeds information by guiding probabilistic generative models during content synthesis. In text generation,

it typically embeds bits by sampling steering tokens within the model's conditional probability distribution. Zhang et al. (Zhang et al., 2021) achieve information-theoretic alignment via equal mass dynamic binning but rely on numerically fragile reconstruction; Ding et al. (Ding et al., 2023) strengthen security with distribution copies and seed-based decoding at the cost of synchronization and latency; Wu et al. (Wu et al., 2024) reduce shift using SVO and sentiment prompting, yet remain template and keyword dependent under domain shift. Taken together, these methods still rely on hand-crafted or pre-defined strategies and struggle to balance efficiency, imperceptibility, and security, especially under high embedding rates - a well-recognized challenge in the the steganography domain, known as the "trilemma."

To fundamentally address this trilemma and inspired by cognitive science showing that creative solutions often emerge from coherently combining seemingly unrelated knowledge (Koestler, 1964; Benedek et al., 2012; Lee and Chung, 2024), we propose Auto-Stega, an agent-driven, self-evolving framework (as shown in Fig. 1) that is the first to realize self-evolving steganographic strategies by automatically discovering, composing, and adapting strategies at inference time. The framework unifies strategy retrieval, stego text generation, calibrated multi-objective evaluation, and strategy summarization with library update in a closed-loop, continually maintaining a searchable strategy library and coordinating efficiency, imperceptibility, and security across tasks. For high embedding rates, we also introduce PC-DNTE, an optional algorithm of our own design that integrates without changing the overall operation. The contributions are as follows:

- **The first text steganography framework for self-evolving steganographic strategies.** It performs automatic strategy discovery and adaptation, coordinating efficiency, imperceptibility, and security through a closed loop comprising strategy retrieval, stego text generation, evaluation, and library updating.

- **Lifelong evolving steganographic strategy.** A summarizer LLM codifies successful strategies into structured, searchable entries; retrieval based on score changes and subsequent reranking drive continual improvement and reliable transfer across corpora and task settings.

- **Plug-and-play high embedding rates mapping.** We design PC-DNTE, an algorithm that keeps sampling close to the base model's conditional distribution and aligns probability mass at the bin level under high embedding rates.

- **Low-cost, requirement-oriented inference.** The pipeline runs entirely at inference time, without the need for training, fine-tuning or parameter access. It adapts to user defined requirements, delivering superior performance across various contexts.

2 Related Work

2.1 Large Language Models

LLMs are advanced neural networks trained on extensive text corpora, enabling them to generate and comprehend natural language. These models have been employed as task-specific data generators for tabular data, relational triples and instruction data, achieving reasonable zero-shot quality under simple class-conditional prompts (Chia et al., 2022; Schick and Schütze, 2021).

Beyond data generation, LLMs are increasingly framed as engines for scientific discovery (Karpatne et al., 2025; Luo et al., 2025). Tool-augmented, agentic systems illustrate this shift: ChemCrow (M. Bran et al., 2024) links LLMs with expert chemistry tools, and Co-scientist (Boiko et al., 2023) automates experiment planning and cloud lab execution. Recent work has also explored hypothesis discovery and evaluation through systems like ResearchAgent (Baek et al., 2025) and LiveIdeaBench (Ruan et al., 2024), which propose and refine ideas based on literature and single keyword prompts. Complementing these trends, jailbreak studies (Liu et al., 2025; Zhou et al., 2025a) demonstrate self-evolving jailbreak strategies, where agents autonomously discover and refine attacks over time.

2.2 Generative Text Steganography

Generative text steganography designs steganographic strategies that steer LLMs to select tokens so as to embed secret information within natural language. In this paradigm, a steganographic mapping associates bit strings with sampling decisions inside the model conditional token distribution, thereby controlling token selection by the LLM. From a provable security perspective, Zhang et al. (Zhang et al., 2021) propose Adaptive Dynamic

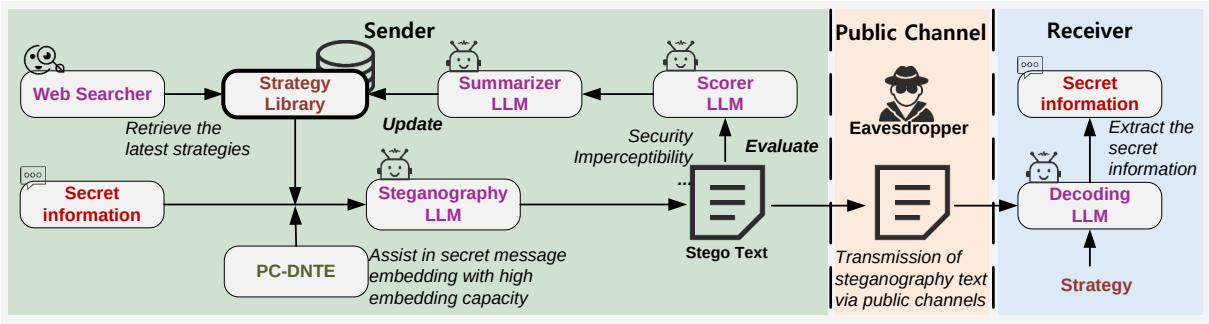

Figure 2: The overall framework of Auto-Stega.

Grouping (ADG), which recursively groups vocabulary tokens to embed information while aligning with the base distribution under stated assumptions. Ding et al. (Ding et al., 2023) introduce "Distribution Copies" (Discop), a mapping designed to preserve the model's original distribution during embedding. Wu et al. (Wu et al., 2024) implement a prompt based mapping that leverages interactions with LLM APIs to realize practical steganographic encoding. Despite these advances, current methods still face challenges in maintaining imperceptibility across high embedding rates and domains.

3 Method

Auto-Stega is an LLM-based text steganography framework that employs an agent-driven, self-evolving mechanism to continually update a strategy library and provide steganographic strategies tailored to the task. As shown in Fig. 2, the system consists of a sender and a receiver. On the sender side, a closed loop of "generation, evaluation, summarization, and update" operates. The web searcher retrieves up-to-date strategies according to task requirements and updates the library; the steganography LLM then generates stego text, and the evaluation module, via the scorer LLM, assigns scores on multiple metrics, including efficiency, imperceptibility, and security. The summarizer LLM writes structured strategy entries back to the library, thereby realizing continuous strategy evolution. On the receiver side, the decoding LLM accurately recovers the secret information according to the shared steganographic strategy.

3.1 Web Searcher

Starting from a library L of text steganography strategies, the web searcher uses extensible literature retrieval and information extraction templates to mine recent work and convert candidate methods into executable entries. It evaluates each method with standardized metrics, assigns recommended scenarios, and compares it with existing entries; candidates that exceed the threshold S_T are admitted to the lifelong evolving steganographic strategy library.

Notation	Description
T_c	Cover text
T_s	Stego text
B	Number of bins
S	Score (scalar)
θ	Fixed params of base AR LM
m	Target bin index
S_T	Score threshold
R	Evaluation response

Table 1: Notations

3.2 Steganographic Generator and Evaluator

Given a request M, the system first filters candidate strategies using the "Applicable Scenarios" field recorded in each library entry. It then performs multi-objective ranking of candidates based on evaluation metrics, selects the top-k strategies, and uses steganography LLM to generate stego text. The resulting outputs are assessed by the evaluation module; if the predefined acceptance criteria are not met, generation is re-initiated with the next strategy in the ranked set, and the process iterates until the stopping condition is satisfied.

The evaluation module, comprising the detection LLM, which estimates the probability that a given text is steganographic and provides explanatory rationales, and other components, applies predefined criteria, including embedding rate, perplexity (PPL) (Mikolov et al., 2010), and semantic similarity (SS). The embedding rate (ER) can be computed as:

$$ER = \frac{N_b}{W} \tag{1}$$

where N_b is the total number of bits in the embed-

ded secret information, and W is the total number of words in the stego text. PPL is a common quantitative measure in text generation, defined as:

$$\mathrm{PPL} = \exp\left(-\frac{1}{N_w}\sum_{i=1}^{N_w}\log p(w_i \mid w_1, \ldots, w_{i-1})\right). \tag{2}$$

where N_w is the length of the text, w_i is the i-th token in text, and $p(w_i \,|\, w_1, \ldots, w_{i-1})$ is the probability assigned by the language model to the i-th word given the preceding words. The scorer LLM then compares the response R and the score S produced by the evaluation module against predefined criteria to determine whether the steganographic requirements are satisfied. Appendix A.6 gives the full formulation of the scoring function S.

3.3 Strategy Library Construction

The steganographic strategy library L is organized into two stages: warm-up and lifelong learning.

In the warm-up stage, we execute a generate–evaluate loop and log (T_s, R, S) for each request. A summarizer LLM abstracts improvements between nearby or sampled records into structured strategy entries and writes them back to the library L after deduplication. Each entry stores motivation, implementation notes, and evaluation summaries. The library L is maintained as a key–value store whose keys κ are response embeddings for efficient retrieval. Admission is gated by thresholded scores S_T, and operational details are given in Fig. 3.

In the lifelong learning stage, given a request M and current L, the system retrieves and ranks candidate strategies, regenerates by evolving new strategies from existing entries in L when acceptance criteria are not met, admits improved entries under the same threshold rule, and terminates when scores exceed preset thresholds or the iteration budget is reached. Under the same thresholding rule, strategy retrieval produces a ranked shortlist Γ and a selected set EFFECTIVE. Each candidate strategy $\gamma \in \Gamma$ is evaluated with an overall score $S(\gamma)$, and the accepted subset is

$$\mathcal{A} \;=\; \{\gamma \in \Gamma : S(\gamma) \geq S_T\}. \tag{3}$$

The selection rule is then

$$\gamma^\star \;=\; \arg\max_{\gamma \in \mathcal{A}} S(\gamma). \tag{4}$$

$$\mathrm{EFFECTIVE} \;=\; \begin{cases} \{\gamma^\star\}, & |\mathcal{A}| = 1, \\ \mathcal{A}, & |\mathcal{A}| > 1, \\ \varnothing, & |\mathcal{A}| = 0. \end{cases} \tag{5}$$

Algorithm 1 Strategy Retrieval and Shortlisting

Require: Request M; library L (entry j with key κ_j); shortlist size k
Ensure: Shortlist Γ
Functions: GE: generate and evaluate (outputs T_S, R, S); EMB: embed R; STS: stable similarity top-$2k$; MD: metric discrepancy vs. current metrics; TOPK: pick k by criterion

```
(T_S, R, S) ← GE(M)                    // gen/eval
E_R ← EMB(R)                           // embed
C ← STS(L, E_R, 2k)                    // retrieve
for all j ∈ C do
    d_j ← MD(j, S)                     // score gap
end for
Γ ← TOPK(C, {d_j}, k)                  // shortlist
return Γ
```

When $\mathcal{A} \neq \varnothing$, the system proceeds with EFFECTIVE. A single accepted strategy is applied, while multiple accepted strategies are assembled to drive further evolution. Otherwise ($\mathcal{A} = \varnothing$), the shortlist is discarded and alternative discovery is triggered. This rule turns observed gains into reusable fragments and supports continual evolution.

3.4 Strategy Retrieval

Given a request M, the steganographic LLM and the evaluator module produce a stego text, a response, and an overall score S. We embed the response R and retrieve the top-$2k$ entries from L by similarity to their keys κ_j, with stable deterministic tie resolution. The candidates are reranked by metric discrepancy with the current metrics; the best k form the shortlist Γ. Note that in the first iteration, no response is available for retrieval reference. Thus the steganography LLM is prompted without employing a steganographic strategy in the first iteration. Algorithm 1 formalizes the retrieval and selection procedure.

3.5 Plug-and-Play High Embedding Rates Mapping

Prior heuristics such as top-p (nucleus) and locally typical sampling shape the candidate set near the model conditional distribution, improving fluency. However, they lack control over token selection that is calibrated for capacity and aware of the underlying distribution: per-step capacity remains uncalibrated, probability mass is unstructured, and the mapping from bits to tokens is unstable as candidate diversity varies. Accordingly, we propose PC-DNTE (Parity-Constrained Dynamic Nucleus-Typical Encoder).

PC-DNTE is a training-free encoder. At each step, it adapts $p \in [p_{\min}, p_{\max}]$ to build a nucleus set under $p_\theta(\cdot \mid \mathrm{ctx} * t)$ and applies a local typicality filter. The remaining candidates are split into

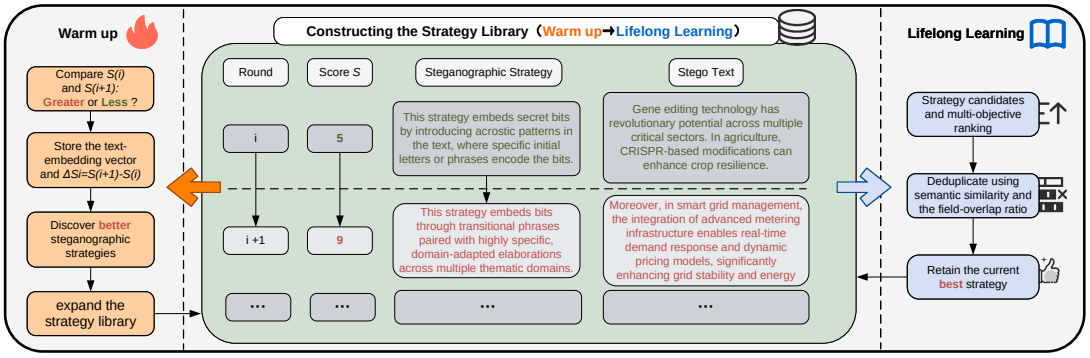

Figure 3: Steganographic strategy library construction: from warm-up to lifelong learning.

$B = 32$ equal-mass bins $\{G_m\}_{m=0}^{31}$. The bin index is driven by payload bits and is approximately uniform. Since each G_m carries the same probability mass under $p_\theta(\cdot \mid \mathrm{ctx}_t)$, the induced mass over bins aligns with the base distribution at the region level. With $B = 32$, a 5-bit index selects m, and one parity bit b^* further filters tokens in G_m by a seed-conditioned parity, enabling deterministic decoding. If the filtered set is empty, p is increased and the step is retried; if it remains empty, a no-embed step is recorded, the bits are deferred, and $\arg\max_{y \in G_m} p_\theta(y \mid \mathrm{ctx}_t)$ is emitted. Near sentence boundaries, a small log-odds boost to EOS and punctuation, together with a bounded closure routine, encourages well-formed breaks. This adjustment only affects the next-step candidate set, so embedded bits are unchanged. Selection within a bin is greedy, so token-level preservation is not exact. The goal is to keep sampling close to the base conditional distribution while sustaining stable capacity at high embedding rates (Algorithm 2).

3.6 Secret Information Extraction

In the secret extraction module, when configured with the same steganographic strategy as the encoding side, the decoding LLM is able to accurately decode the embedded secret information.

Algorithm 2 PC-DNTE

Require: conditional $p_\theta(\cdot \mid \mathrm{ctx})$; ciphertext BITS; $p_{\min}, p_{\max} \in (0, 1]$; bins $B{=}32$; scorer TYP$(\cdot)$; seed SEED; closure ε; budget K; punctuation set $\mathcal{P}$
Ensure: stego text $T_S = (t_1, \ldots, t_T)$
Functions: ADAP: adapt $p \in [p_{\min}, p_{\max}]$; DYNNUC: nucleus under p_θ at p; LOCTYP: local typicality filter; EQM: split to B equal-mass bins; READBITS: pop n bits; PH: seed-conditioned parity; INCP: enlarge p up to $p_{\max}$; NOEMB: record no-embed; QUEUEBITS: enqueue bits; UPDCTX: update context; SBLIKELY: boundary test; BOOSTEOS: boost $\mathcal{P}$; CLOSESTEPS: bounded closure

$T_S \leftarrow [\,]$; $t \leftarrow 1$
while NOTEOM(BITS) **do**
  NO_EMB $\leftarrow$ FALSE
  $p \leftarrow$ ADAP$(p_{\min}, p_{\max}, p_\theta(\cdot \mid \mathrm{ctx}_t))$ *// adapt*
  $C_t \leftarrow$ DYNNUC$(p_\theta(\cdot \mid \mathrm{ctx}_t), p)$; $C'_t \leftarrow$ LOCTYP$(C_t, \mathrm{ctx}_t, \mathrm{TYP})$; $\{G_0, \ldots, G_{31}\} \leftarrow$ EQM(C'_t, B)
  $m \leftarrow$ READBITS(BITS, 5); $b^* \leftarrow$ READBITS(BITS, 1)
  $S \leftarrow \{\, y \in G_m : \mathrm{PH}(y, \mathrm{ctx}_t, \mathrm{SEED}) = b^* \,\}$
  if $S = \varnothing$ **then**
    $p' \leftarrow$ INCP$(p, p_{\max})$
    $C_t \leftarrow$ DYNNUC$(p_\theta(\cdot \mid \mathrm{ctx}_t), p')$; $C'_t \leftarrow$ LOCTYP$(C_t, \mathrm{ctx}_t, \mathrm{TYP})$; $\{G_0, \ldots, G_{31}\} \leftarrow$ EQM(C'_t, B)
    $S \leftarrow \{\, y \in G_m : \mathrm{PH}(y, \mathrm{ctx}_t, \mathrm{SEED}) = b^* \,\}$
  end if
  if $S = \varnothing$ **then**
    NOEMB(); DEFBITS$(\{m, b^*\})$ *// enqueue bits*
    $y_t \leftarrow \arg\max_{y \in G_m} p_\theta(y \mid \mathrm{ctx}_t)$;
    NO_EMB $\leftarrow$ TRUE *// fallback*
  else
    $y_t \leftarrow \arg\max_{y \in S} p_\theta(y \mid \mathrm{ctx}_t)$ *// greedy*
  end if
  $T_S \leftarrow T_S \cup \{y_t\}$; $\mathrm{ctx}_{t+1} \leftarrow$ UPDCTX(ctx_t, y_t); $t \leftarrow t + 1$
  if SBLIKELY$(\mathrm{ctx}_t, \varepsilon) \vee$ NO_EMB **then**
    BOOSTEOS$(\mathcal{P}, \varepsilon)$; CLOSESTEPS$(\mathrm{ctx}_t, K)$ *// reweight*
  end if
end while
return T_S

4 Experiment

In this section, we evaluate the performance of Auto-Stega in terms of efficiency, imperceptibility, and security, as well as human evaluation. The details of the experimental setup and result analysis are presented in the following subsections. Additional implementation details and supplementary results are provided in the appendices. Appendix A.2 provides representative high scoring strategy entries, and Appendix A.3 provides the prompt templates for Auto-Stega. Appendix A.4 reports ablation studies. Appendix A.5 introduces an additional anti-steganalysis experiment.

4.1 Experimental Setup

Datasets. We evaluated Auto-Stega in three public corpora: the News Category Dataset (News) (Misra, 2022), the Large Movie Review dataset (Movie) (Maas et al., 2011), and Sentiment140 (Tweet) (Go et al., 2009). For quantitative comparability, we use 12,300 News items (300 per category across 41 categories), 20,000 Large Movie Review entries, and 20,000 Sentiment140 tweets. The combination spans domains, styles, and lengths,

Methods	News			Movie			Sentiment		
	ER↑	PPL*↓	SS↑	ER↑	PPL*↓	SS↑	ER↑	PPL*↓	SS↑
ADG	4.99	8.58	0.40	5.44	5.39	0.45	5.22	**0.53**	0.39
Discop	4.62	0.70	0.58	4.94	0.62	0.58	5.41	0.94	0.48
LLM-Stega	5.94	0.46	0.61	4.63	0.28	0.61	7.90	0.80	0.58
Ours	**5.97**	**0.01**	**0.63**	**5.69**	**0.04**	**0.62**	**8.14**	0.84	**0.60**

Table 2: Experimental results of PPL^* and SS under high embedding rates. ↑ higher is better, ↓ lower is better.

which allows us to assess efficiency, imperceptibility, and security under cross-scenario distribution shifts while maintaining sufficient scale for reliable estimation. Additional cross-lingual and cross-domain evaluation is provided in Appendix A.7 to assess generalization.

Baselines. We compare Auto-Stega with three SOTA baselines, including ADG (Zhang et al., 2021), Discop (Ding et al., 2023), and LLM-Stega (Wu et al., 2024). To demonstrate that our method maintains strong performance across embedding capacities, we conduct experiments under both high embedding rates and low embedding rates settings and compare the stego text generated by different methods in each case.

Hyperparameters. Evaluation was conducted in two stages. An initial warm up exploration was performed on 50 steganographic requests, running $N = 150$ iterations to build the initial strategy library. Using this library, the runtime lifelong learning phase was then executed: for each request in the experimental datasets, 5 iterations were carried out. A full steganography round is defined as generating stego text for a given cover text and completing the evaluation. Unless otherwise stated, the maximum iteration budget was set at $T = 150$ and the stopping threshold to $S_T = 8.5$ for each data instance. During evaluation, the strategy library was frozen and an additional stego generation pass was conducted on the three datasets. For plug-and-play mapping, $p_{\min} = 0.88$ and $p_{\max} = 0.95$ were used to obtain sufficient candidate tokens while suppressing low typicality tails, and GPT-2 was used to compute next-token probabilities. We use DEEPSEEK-V3.2 for the steganography, decoding, and scorer LLMs, and GPT-4O for the summarizer LLM.

Efficiency. In the field of steganography, the embedding rate serves as a key evaluation metric to assess the efficiency of a method. In text steganography, the embedding rate is defined as the average number of secret bits embedded per word (bits per word, bpw), as shown in Formula 1. In PC-DNTE, the encoder operates at the token level, so capacity is naturally parameterized as bits per token. In the GPT-2 setup, this corresponds approximately to 1 bit/token $\approx$ 1.4 bit/word.

Methods	News		Movie		Sentiment	
	PPL*↓	SS↑	PPL*↓	SS↑	PPL*↓	SS↑
ADG	9.26	0.45	5.92	0.49	**0.39**	0.38
Discop	0.90	0.60	0.88	0.59	0.99	0.44
LLM-Stega	0.03	0.64	0.11	0.67	0.94	0.58
Ours	**0.01**	**0.66**	**0.08**	**0.72**	0.86	**0.65**

Table 3: Experimental results of PPL^* and SS at a low embedding rate of 0.1 bpw.

Imperceptibility. In text steganography, imperceptibility is typically categorized into text quality and statistical imperceptibility. Text quality captures human-perceived naturalness, including fluency, coherence, and semantic fidelity relative to the cover text. Statistical imperceptibility assesses the distributional alignment with the cover. Together, these facets provide a measure of whether a method conceals information without degrading readability or altering the underlying statistical characteristics of the text.

(1) Text Quality. PPL and SS are commonly used to evaluate the text quality of generated stego text. In this experiment, we choose the GPT-2 model to calculate the PPL values of different stego text. Prior work (Yang et al., 2018) indicates that social media text exhibits substantial stylistic variability, so human-written sentences deviate widely from any single language model; consequently, stego text whose PPL is closer to that of human text is considered more secure. To compare across corpora with different baseline perplexities, we use a normalized perplexity deviation, $\text{PPL}^* = \frac{|\text{PPL}_{\text{stego}} - \text{PPL}_{\text{cover}}|}{\text{PPL}_{\text{cover}}}$ where lower values indicate better text quality. For the SS, we choose the Sentence-bert (Reimers and Gurevych, 2019) method and use the roberta-base-nli-mean-tokens (Liu et al., 2019) model to extract sentence vectors, to calculate the cosine similarity between the cover text and the stego text.

(2) Statistical Imperceptibility. Kullback-Leibler Divergence (KLD) serves as an evaluation metric to measure the statistical imperceptibility of steganographic algorithms by comparing the distribution of the generated stego text with the distribution of the cover text. In our experiments, we adopt the KLD proposed by Zhang et al. (Zhang et al., 2021) to evaluate the statistical imperceptibility of the methods tested. It is formulated as follows:

$$\mathrm{KLD}(\mu_x, \sigma_x, \mu_y, \sigma_y) = \sum\left[\log\left(\frac{\sigma_y}{\sigma_x}\right) + \frac{\sigma_x^2 + (\mu_x - \mu_y)^2}{2\sigma_y^2} - \frac{1}{2}\right] \tag{6}$$

Where μ_x and σ_x are the mean and standard deviation of the cover text, while μ_y and σ_y represent those of the stego text.

Security. Steganography and steganalysis are locked in an ongoing adversarial game. As such, anti-steganalysis performance serves as the paramount metric for evaluating the security of a steganographic algorithm. In this experiment, we leverage three advanced steganalysis methods, including SANet (Xue et al., 2024), BiLSTM-Dense (BD) (Yang et al., 2020), and Bert-FT (BF) (Peng et al., 2021). The results are reported in terms of steganalysis accuracy—defined as the rate of correctly identifying whether a given text contains hidden information—where values closer to 50% (equivalent to random guessing) indicate stronger anti-steganalysis performance.

Human Evaluation. For the human evaluation, we focus on evaluating three key aspects of the generated texts: fluency, coherence, and relevance. Our human evaluation was conducted by five researchers who were not involved in the study, and each one holds a master's degree. The evaluation process involved a mixed assessment sheet containing both generated stego and cover text; only stego text scores were selected for analysis. The researchers, unaware of the underlying algorithms, labels, or any other details, were tasked solely with assessing and scoring text quality. Each researcher independently rated the texts on a five-point scale (ranging from 'very poor' to 'very good'). The higher evaluation score denotes the better generated stego text.

4.2 Results and Analysis

(1) Efficiency and Imperceptibility. In generative text steganography, efficiency and imperceptibility are intrinsically linked: changes to the embedding rate (ER) modify sampling behavior and can materially affect both text quality and statistical imperceptibility. Accordingly, we analyze efficiency and imperceptibility jointly in this section.

Payload	Methods	News	Movie	Sentiment
		KLD↓	KLD↓	KLD↓
Low	ADG	2.63	2.28	**1.90**
	Discop	2.78	**1.94**	2.17
	LLM-Stega	2.41	2.83	2.43
	Ours	**2.21**	2.13	2.11
High	ADG	2.63	2.27	**1.89**
	Discop	1.94	**1.96**	2.17
	LLM-Stega	2.33	2.22	2.19
	Ours	**1.87**	1.98	2.05

Table 4: The experimental results of the statistical imperceptibility under high and low embedding rates. The best results are highlighted in bold red, and the second bests are shown in red without bold. ↓ lower is better.

Text Quality. The experimental results for text quality and ER are listed in Tables 2 and 3. Auto-Stega achieves a higher ER while delivering superior performance. In our experiments, the average PPL of cover sentences is 113.89 on the News corpus, 111.12 on the Movie corpus, and 1224.79 on the Tweet corpus. Auto-Stega achieves the highest SS across all datasets and attains the lowest PPL^* on News and Movie. On Tweet, the stego text exhibit a slightly higher PPL^*; nevertheless, Auto-Stega achieves a 42.2% relative reduction compared with the SOTA baseline. To quantify the robustness of the gains and assess statistical significance, all methods were rerun with multiple random seeds. For each corpus, paired t-tests were conducted on sentence-level PPL^*. The tests indicate that the improvements over the best baselines are statistically significant. These corpus characteristics highlight the importance of maintaining stylistic consistency with the cover text, thereby improving overall textual quality. Within Auto-Stega, PC-DNTE treats text quality as an explicit objective during generation and selection, thereby generating high quality stego text. Qualitative steganographic examples produced by the proposed method and prior baselines are provided in Appendix A.1.

Statistical Imperceptibility. In this part, we compute sentence-level KLD using distributions derived from a pre-trained BERT encoder, comparing the cover and stego sets. In our experiments, we take the logarithm of the KLD values to facilitate a clearer comparison of statistical imperceptibility between algorithms. As shown in Table 4, under both high and low embedding rates, Auto-Stega achieves the lowest KLD on the News dataset, and the second lowest KLD on the Movie and Tweet

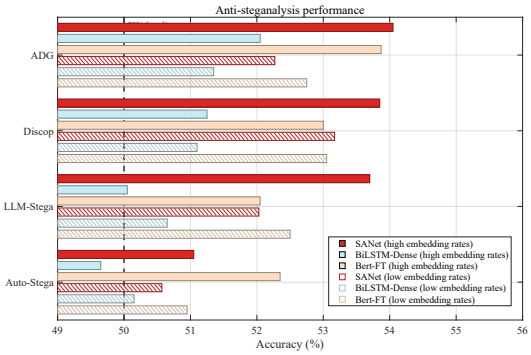

Figure 4: The results of the anti-steganalysis performance across low and high embedding rates.

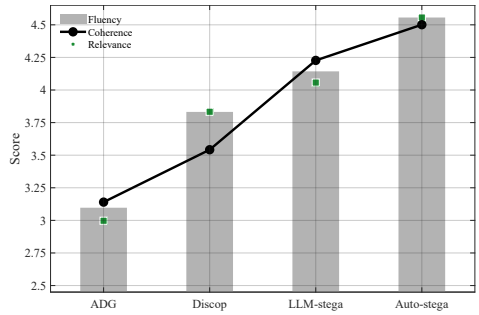

Figure 5: The results of the human evaluation.

corpora. ADG (Zhang et al., 2021) performs well on tweets, whereas Discop (Ding et al., 2023) performs well on reviews, because their equiprobable grouping and distribution preserving sampling better match the peaked token distributions and formulaic phrasing characteristic of these datasets. In addition, our method can be configured to incorporate sentiment conditioning or template-based generation, which can reduce KLD on these two corpora while preserving semantic fidelity.

Taken together, these results demonstrate that Auto-Stega achieves the strongest overall imperceptibility across embedding rates, indicating that it maintains superior perceptual concealment even under high embedding rates and is well-suited for practical deployment in real-world applications.

(2) Security. We used splits of 10,000 training pairs, 1,000 validation pairs, and 1,000 test pairs, each pair containing a cover text and its corresponding stego text. Fig. 4 presents experimental results on anti-steganalysis performance across different embedding rates. The proposed method, Auto-stega, continues to exhibit superior performance across all metrics when compared to other steganography methods. Notably, Auto-Stega achieves SANet of 51.05%, BD of 49.65%, and BF of 52.35%, representing an average 1.2% reduction relative to the SOTA method. This improvement reflects that, during iterative generation and selection, Auto-Stega explicitly evaluates anti-steganalysis performance and admits candidates that score better on this criterion, thereby producing stego text that is harder to detect. Overall, these results indicate that Auto-Stega achieves the strongest security among evaluated methods.

(3) Human Evaluation. To further assess the effectiveness of the proposed Auto-Stega, we conducted a human evaluation, with detailed results shown in Figure 5. Human ratings demonstrate that Auto-Stega outperforms the three baseline methods, producing stego text that is more fluent, coherent, relevant, and more imperceptible. This finding is particularly significant in the context of text steganography, underscoring the potential of Auto-Stega in applications where information concealment is critical. Moreover, as shown in Appendix A.1, we randomly selected several examples of stego text generated by Auto-Stega and the baselines for qualitative analysis. The results indicate that Auto-Stega generates texts with high fluency, grammatical correctness, and semantic coherence.

5 Conclusion

We presented Auto-Stega, an agent-driven, self-evolving framework that discovers, composes, and adapts steganographic strategies entirely at inference time. By continually curating a searchable strategy library without retraining and by optionally employing the plug-and-play PC-DNTE to keep sampling close to the base model's conditional distribution, the system supports higher embedding rates while maintaining imperceptibility and security. To our knowledge, Auto-Stega is the first text steganography framework in an LLM-agent setting with self-evolving strategies. The results suggest that this paradigm already captures part of the potential of LLMs for steganography. As LLM capabilities continue to improve, further gains in steganographic performance are expected. Experiments demonstrate a consistently superior trade-off among efficiency, imperceptibility, and security compared with SOTA baselines. Future work will extend the framework to multilingual and multimodal covers, strengthen robustness against adaptive steganalyzers and noisy channels, and reduce inference latency.

Limitations

While Auto-Stega achieves efficient covert text communication across diverse textual domains, it does not yet address multimodal settings in which information may be distributed across heterogeneous cover media. As future work, we will extend Auto-Stega to multimodal steganography with multiple covers by integrating modality aware generators and evaluators with the strategy library, enabling coordinated cover selection, adaptive embedding rate allocation, and cross modal redundancy. This extension is expected to substantially improve practical utility and covertness in real world deployments.

References

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. ResearchAgent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6709–6738.

Mathias Benedek, Tanja Könen, and Aljoscha C Neubauer. 2012. Associative abilities underlying creativity. *Psychology of Aesthetics, Creativity, and the Arts*, 6(3):273.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. 2023. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578.

Yu Cheng, Jiuan Zhou, Jiawei Chen, Zhaoxia Yin, and Xinpeng Zhang. 2025. Rfnns: Robust fixed neural network steganography with popular deep generative models. *arXiv preprint arXiv:2505.04116*.

Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. 2022. RelationPrompt: Leveraging prompts to generate synthetic data for zero-shot relation triplet extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 45–57.

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*, pages 2238–2255.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Mingzhi Hu and Hongxia Wang. 2025. Mutual information-optimized steganalysis for generative steganography. *IEEE Transactions on Information Forensics and Security*, 20:1852–1865.

Anuj Karpatne, Aryan Deshwal, Xiaowei Jia, Wei Ding, Michael Steinbach, Aidong Zhang, and Vipin Kumar. 2025. Ai-enabled scientific revolution in the age of generative ai: second nsf workshop report. *npj Artificial Intelligence*, 1(1):18.

Arthur Koestler. 1964. The act of creation.

Byung Cheol Lee and Jaeyeon Chung. 2024. An empirical investigation of the impact of chatgpt on creativity. *Nature Human Behaviour*, 8(10):1906–1914.

Mengdi Li, Kai Mu, Ping Zhong, Juan Wen, and Yiming Xue. 2019. Generating steganographic image description by dynamic synonym substitution. *Signal Processing*, 164:193–201.

Yiming Li, Kejiang Chen, Yaofei Wang, Xin Zhang, Guanjie Wang, Weiming Zhang, and Nenghai Yu. 2025. Coas: Composite audio steganography based on text and speech synthesis. *IEEE Transactions on Information Forensics and Security*, 20:5978–5991.

Guorui Liao, Jinshuai Yang, Weizhi Shao, and Yongfeng Huang. 2025. A framework for designing provably secure steganography. In *34th USENIX Security Symposium*, pages 6837–6856.

Xiaogeng Liu, Peiran Li, G. Edward Suh, Yevgeniy Vorobeychik, Zhuoqing Mao, Somesh Jha, Patrick McDaniel, Huan Sun, Bo Li, and Chaowei Xiao. 2025. Autodan-turbo: A lifelong agent for strategy self-exploration to jailbreak llms. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ziming Luo, Zonglin Yang, Zexin Xu, Wei Yang, and Xinya Du. 2025. Llm4sr: A survey on large language models for scientific research. *arXiv preprint arXiv:2501.04306*.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. 2024. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6(5):525–535.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Laijin Meng, Xinghao Jiang, Qiang Xu, and Tanfeng Sun. 2025. A robust coverless video steganography based on two-level dct features against video attacks. *IEEE Transactions on Multimedia*, 27:6434–6448.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048.

Rishabh Misra. 2022. News category dataset. *arXiv preprint arXiv:2209.11429*.

Wanli Peng, Jinyu Zhang, Yiming Xue, and Zhenghong Yang. 2021. Real-time text steganalysis based on multi-stage transfer learning. *IEEE Signal Processing Letters*, 28:1510–1514.

Yinyin Peng, Yaofei Wang, Donghui Hu, Kejiang Chen, Xianjin Rong, and Weiming Zhang. 2024. Ldstega: Practical and robust generative image steganography based on latent diffusion models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 3001–3009.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992.

Kai Ruan, Xuan Wang, Jixiang Hong, Peng Wang, Yang Liu, and Hao Sun. 2024. Liveideabench: Evaluating llms' divergent thinking for scientific idea generation with minimal context. *arXiv preprint arXiv:2412.17596*.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951.

Mohammad Shirali-Shahreza. 2008. Text steganography by changing words spelling. In *2008 10th international conference on advanced communication technology*, volume 3, pages 1912–1913.

Jie Wang, Yihao Wang, Yihao Li, Ru Zhang, and Jianyi Liu. 2025a. Multi-classification of linguistic steganography driven by large language models. *IEEE Signal Processing Letters*, 32:1440–1444.

Zhuang Wang, Linna Zhou, Xuekai Chen, Zhili Zhou, and Zhongliang Yang. 2025b. Stlc-kg: A social text steganalysis method combining large-scale language models and common-sense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25461–25469.

Juan Wen, Xuejing Zhou, Ping Zhong, and Yiming Xue. 2019. Convolutional neural network based text steganalysis. *IEEE Signal Processing Letters*, 26(3):460–464.

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*, pages 10345–10353.

Yiming Xue, Jiaxuan Wu, Ronghua Ji, Ping Zhong, Juan Wen, and Wanli Peng. 2024. Adaptive domain-invariant feature extraction for cross-domain linguistic steganalysis. *IEEE Transactions on Information Forensics and Security*, 19:920–933.

Hao Yang, YongJian Bao, Zhongliang Yang, Sheng Liu, Yongfeng Huang, and Saimei Jiao. 2020. Linguistic steganalysis via densely connected lstm with feature pyramid. In *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, pages 5–10.

Zhongliang Yang, Xiaoqing Guo, Ziming Chen, Yongfeng Huang, and Yujin Zhang. 2018. Rnn-stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295.

Siyu Zhang, Zhongliang Yang, Jinshuai Yang, and Yongfeng Huang. 2021. Provably secure generative linguistic steganography. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3046–3055.

Xin Zhang, Kejiang Chen, Na Zhao, Weiming Zhang, and Nenghai Yu. 2025a. Provably secure public-key steganography based on admissible encoding. *IEEE Transactions on Information Forensics and Security*, 20:3161–3175.

Ziping Zhang, Jiamin Zeng, Yan Xu, Xiaowei Yi, Yun Cao, and Changjun Liu. 2025b. Triple-stage robust audio steganography framework with aac encoding for lossy social media channels. In *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, pages 131–141.

Andy Zhou, Kevin Wu, Francesco Pinto, Zhaorun Chen, Yi Zeng, Yu Yang, Shuang Yang, Sanmi Koyejo, James Zou, and Bo Li. 2025a. Autoredteamer: Autonomous red teaming with lifelong attack integration. *arXiv preprint arXiv:2503.15754*.

Pengcheng Zhou, Zhengyang Fang, Zhongliang Yang, Zhili Zhou, and Linna Zhou. 2025b. Efficient streaming voice steganalysis in challenging detection scenarios. *IEEE Transactions on Information Forensics and Security*, 20:5966–5977.

Qing Zhou, Ping Wei, Zhenxing Qian, Xinpeng Zhang, Sheng Li, and Chuan Qin. 2025c. Conditional flow-based generative steganography. *IEEE Transactions on Dependable and Secure Computing*, 22(5):5632–5647.

A Appendix

A.1 Comparison of Steganographic Examples Generated by Proposed Method and Existing Methods.

In this section, we analyze randomly sampled stego text generated by Auto-Stega and three baselines (ADG, Discop, and LLM-Stega). As shown in Table 5, texts from ADG and Discop frequently exhibit surface issues, such as awkward collocations, agreement errors, and missing function words, especially near high entropy positions induced by their mappings. LLM-Stega generally maintains sentence level fluency, but shows intersentential drift and occasional semantic inconsistencies at higher embedding rates, reflecting template and keyword rigidity. In contrast, Auto-Stega produces texts that are fluent, syntactically well formed, and semantically consistent across sentences, while more closely matching the style of the source corpus (for example, concise headlines in News and an informal register in Tweets). Manual inspection also finds fewer detectability cues, consistent with the smallest difference from cover perplexity observed in the experimental results. Overall, the results indicate that Auto-Stega yields more coherent and less detectable stego text than the compared methods.

A.2 Evolution of Steganographic Strategies

In this section, we present steganographic strategy entries from the library spanning a range of scores. Each entry follows a standardized schema comprising *Strategy*, *Definition*, *Technique*, *Applicable Scenarios*, *Characteristics*, and *Good Examples*, where *Good Examples* include representative stego text with the corresponding metric scores. Across iterations, we observe a clear progression: early strategies yield limited embedding capacity, lower text quality, and weaker anti-steganalysis performance, whereas later strategies achieve higher embedding rate, better fluency and coherence, and improved statistical invisibility. The strategy evolution process in Table 6 shows that Auto-Stega can autonomously develop effective strategies for text steganography, underscoring its practical value for covert communication. The evolution of steganographic strategies is shown in Fig. 6; as the library is progressively updated, the generated stego text exhibits improved empirical performance.

A.3 Full prompts for the Steganography LLM and Summarizer LLM

In this section, we present the prompt templates that operationalize the two core components of Auto-Stega. The Summarizer LLM is prompted to distill observed improvements into structured strategy entries under strict output and safety constraints, as shown in Fig. 7. During the warm up stage, the steganographic LLM embeds a given secret into a cover subject to fluency and coherence requirements, with controlled length variation and avoidance of statistical artifacts, as shown in Fig. 8. During the lifelong learning stage, the generator conditions on retrieved strategy headers, either a single strategy or a composed set, and applies them to produce candidates under the same quality requirements, as shown in Fig. 9. These templates standardize inputs and outputs across iterations, enable compositional use of strategies, and support reliable extraction and consistent evaluation throughout the Auto-Stega framework.

A.4 Ablation Experiment

To clarify the contribution of PC-DNTE versus the self-evolving strategy library, an additional ablation is conducted at the same high embedding rate as in the main experiments: the Auto-Stega framework and PC-DNTE are kept unchanged, the strategy library is first evolved for three summarization–update rounds and then frozen (no further evolution), and stego texts are generated once from this fixed, pre-evolved library. Under this setting, PC-DNTE already achieves a strong trade-off between efficiency and imperceptibility and outperforms prior methods on most metrics. As shown in Table 7, the average PPL^* over the three datasets is reduced by about 35% compared with the best previous method, while SS and KLD remain competitive or better. These results suggest that PC-DNTE accounts for most of the gains at high embedding rates, delivering strong quality-efficiency trade-offs even with a frozen, pre-evolved library. Using the full Auto-Stega framework yields an additional improvement (about 7.8% in average PPL^*).

A.5 Anti-Steganalysis Performance Evaluation

Auto-Stega is further evaluated with the linguistic steganalysis network LS-CNN (LC) (Wen et al., 2019). The experimental settings are the same as those in Section 4.2. In these experiments, LS-

CNN serves as an external detector and its binary detection accuracy on stego texts from each method is reported. An accuracy close to 0.5 indicates that the generated stego texts are nearly indistinguishable from cover texts. As shown in Table 8, Auto-Stega attains accuracies closest to 0.5 on News, Movie, and Sentiment, indicating higher security.

A.6 Full Formulation of the Scoring Function

This appendix specifies the global scoring function S used in §3.2. Each criterion is mapped to a score $s \in [1, 10]$, where larger is better. The final score is obtained by weighted aggregation.

Perplexity closeness between cover and stego texts is quantified by

$$\begin{aligned} c_{\mathrm{PPL}} &= \exp\Big(-\big|\log \mathrm{PPL}_{\text{stego}} - \log \mathrm{PPL}_{\text{cover}}\big|\Big), \\ s_{\mathrm{PPL}} &= 1 + 9\, c_{\mathrm{PPL}}. \end{aligned} \tag{7}$$

Semantic similarity is measured using the cosine similarity of Sentence-BERT embeddings,

$$s_{\mathrm{SS}} = 1 + 9 \cdot \frac{\cos(\text{stego}, \text{cover}) + 1}{2}. \tag{8}$$

Statistical imperceptibility is captured by the divergence term in §3.2 and mapped as

$$s_{\mathrm{KLD}} = \frac{10}{1 + \mathrm{KLD}}. \tag{9}$$

Anti-steganalysis performance is computed from an external detector output p_{det} (stego probability),

$$s_{\mathrm{det}} = 1 + 9\,(1 - p_{\mathrm{det}}). \tag{10}$$

Capacity efficiency is represented by the embedding rate ER. Fixed thresholds $0 < \tau_1 < \cdots < \tau_9$ define a monotone interval mapping,

$$s_{\mathrm{ER}}(\mathrm{ER}) \;=\; 1 + \sum_{k=1}^{9} \mathbf{1}[\mathrm{ER} \geq \tau_k]\,, \tag{11}$$

which yields $s_{\mathrm{ER}} \in \{1, \ldots, 10\}$.

The global score aggregates the metric scores by a weighted sum:

$$\begin{aligned} S \;&= w_{\mathrm{ER}} s_{\mathrm{ER}} + w_{\mathrm{PPL}} s_{\mathrm{PPL}} \\ &\quad + w_{\mathrm{SS}} s_{\mathrm{SS}} + w_{\mathrm{KLD}} s_{\mathrm{KLD}} + w_{\mathrm{det}} s_{\mathrm{det}}, \\ &\sum\nolimits_m w_m = 1. \end{aligned} \tag{10}$$

The weights used in the experiments are

$$\begin{aligned} w_{\mathrm{ER}} = 0.15, \qquad w_{\mathrm{det}} = 0.25, \\ w_{\mathrm{PPL}} = w_{\mathrm{SS}} = w_{\mathrm{KLD}} = 0.20. \end{aligned} \tag{11}$$

A.7 Cross-Lingual and Cross-Domain Evaluation

Additional evaluation is conducted on cross-domain and cross-lingual corpora to assess generalization. The testbed covers Arxiv (scientific abstracts and metadata), an English Wikipedia dump, and THUCNews (Chinese news articles). For each corpus, Table 9 reports embedding rate (ER), normalized PPL*, SS, and KLD. Auto-Stega achieves the highest ER and SS on all three corpora, while keeping PPL* and KLD comparable to, or better than, the strongest baselines. These results suggest consistent performance across scientific, encyclopedic, and news domains, as well as across English and Chinese, with a favorable balance between capacity, efficiency, and imperceptibility.

Methods	Stego text example
ADG	putting for me can eat me can go well
Discop	I'm a Ser, its well over 20 years old and I just started to play. And thats it :D I'd be happy to give back to you please respond up till hour 30 More bugs I'm sure around here there are no edit black on black bars? See below. As if it wasn't clear..in game this spreadsheet just explains what blocks are. Although it didn't make clear to me what kind of bones were I carrying. When you cast it double kill yourself as if you. // "Do you want to... trust me?..." "No... everyone IS right. I definitely agree, even just... wanting a girl you have never known." {K...center, cums, gesture one more time}.....I trust your feelings, you really have." In other words it wasn't a grown up. Shame. DISCONTINUED // "Ah, Engh...huh. Thank you, am I right...?" "Yes. Your Majesty, on behalf
LLM-Stega	Officials announce an exciting new team that promises to bring fresh talent and enthusiasm to the entertainment industry.
Ours	The future might be ready: robots like computers in some scenarios would rapidly replace humans.

Table 5: Stego text examples produced by different methods.

Strategy	Core idea (one sentence)	Technique	Scenarios	Key traits	Representative stego excerpt	Score
Thematic Domain Diversification with Consistent Transitional Enc	Combine semantic preservation with light rewriting to smooth statistical signals and reduce detectability while balancing capacity and robustness (esp. back-translation).	semantic	robustness; low detectability (LLM)	semantic preservation; back-translation robust; hard for detectors	*"Quantum computing will usher in a new era … optimize trading strategies by processing market data … "*	6.125
Domain-Specific Semantic Precision with Transitional Depth	Use domain-specific, non-generic elaborations with consistent transitional phrases; varied transitions align with encoded bits and thematic depth to raise naturalness and lower detectability.	Context-aware semantic embed-ding	general	data-driven; non-generic; varied transitions	*"Moreover, the implementation of quantum-resistant cryptographic algorithms necessitates evaluation of lattice-based schemes … "*	7.625
Domain-Specific Semantic Depth with Multi-Domain Redundancy	Embed via consistent transitions plus rich, domain-adapted elaborations across multiple domains; introduce paragraph-level redundancy to improve error tolerance without generic phrasing.	context-aware semantic embed-ding	general	data-driven; semantic depth; redundancy; low detectability	*"Moreover, in smart grid management, advanced metering enables real-time demand response … enhancing grid stability … "*	9.000

Table 6: Strategy evolution summary for stego text generation (concise view).

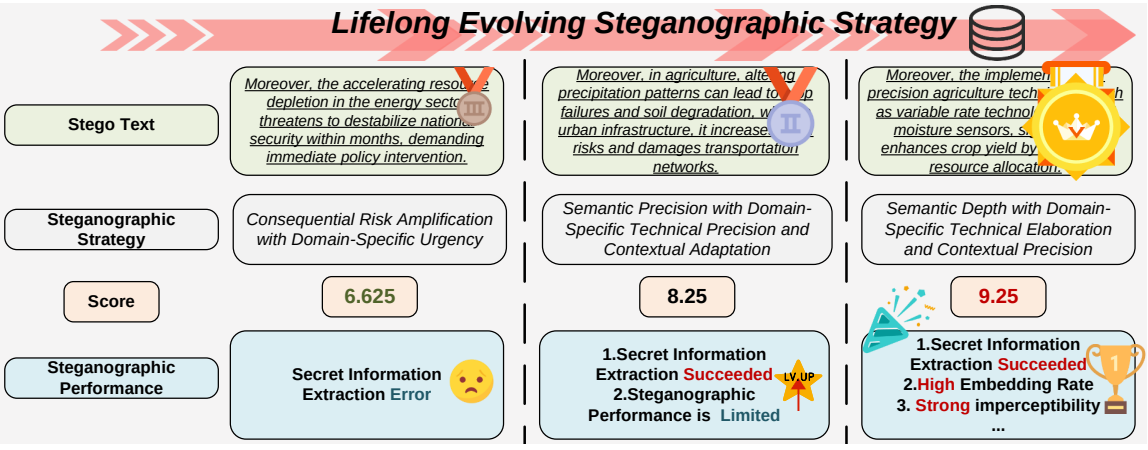

Figure 6: Lifelong evolution of steganographic strategies.

Methods	News			Movie			Sentiment		
	PPL* ↓	SS↑	KLD↓	PPL* ↓	SS↑	KLD↓	PPL* ↓	SS↑	KLD↓
ADG	8.13	0.42	2.61	8.51	0.43	2.27	0.52	0.40	**1.90**
Discop	0.70	0.59	1.93	8.61	0.58	**1.95**	0.95	0.49	2.18
LLM-Stega	0.45	0.62	2.33	0.28	0.61	2.23	**0.80**	0.59	2.20
Ours	**0.02**	**0.64**	**1.87**	**0.06**	**0.62**	1.97	0.88	**0.61**	2.01

Table 7: Results on News, Movie, and Sentiment. PPL* and KLD: lower is better; SS: higher is better.

Payload	Methods	Accuracy (%)
Low	ADG	52.05
	Discop	51.90
	LLM-Stega	52.35
	Auto-Stega	**51.47**
High	ADG	53.96
	Discop	53.05
	LLM-Stega	53.35
	Auto-Stega	**51.67**

Table 8: Steganalyzer accuracy (%) under low and high payloads. Values closer to 50 indicate better statistical imperceptibility, and the closest-to-50 result in each payload setting is highlighted in bold.

Methods	Arxiv				Wiki				THUCNews			
	ER↑	PPL* ↓	SS↑	KLD↓	ER↑	PPL* ↓	SS↑	KLD↓	ER↑	PPL* ↓	SS↑	KLD↓
ADG	3.17	0.96	0.49	2.31	3.44	3.50	0.51	2.03	6.16	2.11	0.75	2.77
Discop	4.94	**0.59**	0.62	1.94	4.19	0.60	0.59	1.93	5.05	0.94	0.70	2.94
LLM-Stega	5.68	2.11	0.49	2.37	5.72	0.51	0.61	2.31	7.05	0.92	0.81	2.51
Ours	**6.30**	0.71	**0.63**	**1.81**	**6.42**	**0.47**	**0.64**	**1.91**	**8.91**	**0.84**	**0.86**	**2.11**

Table 9: Results on Arxiv, Wiki, and THUCNews. ER and SS: higher is better; PPL* and KLD: lower is better.

```python
def strategy_summarizer_prompt():
    return (
"You are a steganography strategy summarizer.\n"
"Think through the task step by step INTERNALLY and summarize your reasoning, but DO NOT reveal your reasoning.\n"
"Output ONLY a valid JSON object and nothing else.\n"
"\n"
"When unsure or if required information is missing, return the empty JSON object: {}.\n"
"\n"
"=== Output contract ===\n"
"Return a JSON object with EXACTLY these keys in this order:\n"
"  name (string),\n"
"  definition (string),\n"
"  technique (array of strings),\n"
"  applicable_scenarios (array of strings),\n"
"  characteristics (array of strings),\n"
"  examples (array of objects with keys: stego_excerpt (string), overall_score (number), scores (object)).\n"
"- Use double quotes for all strings; no trailing commas; UTF-8 text.\n"
"- Match the output language to the cover_text language if detectable; otherwise use English.\n"
"- Never include secrets or secret_bits_preview in the output; never echo raw secret bits.\n"
"\n"
"=== Internal step-by-step plan (do not expose) ===\n"
"1) Parse inputs (stego_text, cover_text, evaluation, used_strategies). Identify signals such as: "
"equiprobable LM, acrostic AM, cover continuation, bin size, temperature, decoding hints, etc.\n"
"2) Draft a concise, distinctive NAME (<= 60 chars, Title Case). Prefer pattern: "
"'<Core Mechanism> + <Key Modifier> (optional params)'.\n"
"3) Write a 1-2 sentence DEFINITION that states mechanism (how), purpose (why), and limits (if any).\n"
"4) TECHNIQUE: list concrete methods/ingredients (e.g., 'equiprobable token binning (bin=64)', "
"'deterministic acrostic A-M->0/N-Z->1', 'cover-conditioned continuation', 'low-temp decoding').\n"
"5) APPLICABLE_SCENARIOS: list contexts where this strategy is appropriate (e.g., 'short social posts', "
"'news-style paragraphs', 'low-detectability requirement', 'receiver knows cover').\n"
"6) CHARACTERISTICS: list crisp traits, mixing strengths/risks/requirements (e.g., 'capacity approx 6 bpw', "
"'good fluency under low temperature', 'robustness limited by rephrasing', 'needs shared cover').\n"
"7) EXAMPLES: choose 1-3 short stego excerpts from stego_text (<= 240 chars each, no ellipsis if avoidable). "
"For each example, set overall_score and scores using evaluation if available; otherwise make a conservative estimate "
"from provided fields. Round numbers to one decimal and clamp to [0,10].\n"
"\n"
"=== Scoring normalization (internal) ===\n"
"- If evaluation provides per-dimension scores in [0,10], copy directly.\n"
"- If given probabilities (e.g., detection probability in [0,1]), map to score as (10 - 10*p), rounded to one decimal.\n"
"- If only raw metrics exist (e.g., PPL), use any provided *_score fields; do NOT invent unseen dimensions.\n"
"- If no usable signals exist for scoring, output the empty JSON object {}.\n"
"\n"
"=== Safety & formatting rules ===\n"
"- Do not include explanations, headings, or Markdown -- JSON only.\n"
"- Do not fabricate technical parameters that are not implied by inputs.\n"
"- Express the strategy unambiguously; given the stego_text and its corresponding strategy description, the embedded secret bits can be decoded with 100% reliability.\n"
)
```

Figure 7: Prompts used by the Summarizer LLM in Auto-Stega.

```python
def base_generate_prompt(secret, cover_stub, method_line=""):
    return f"""
System:
You are a professional text steganography expert. Your task is to embed a secret naturally into carrier text.

User:
You are participating in a steganography research project. Your goal is to produce high-quality stego text. Embed the given secret while keeping the text natural and fluent.

Requirements:
1) Secret bitstring to embed: {secret}
2) Carrier text: {cover_stub}
3) Keep the text natural and fluent; avoid statistical artifacts
4) Limit length change to within +/- 10% (if you write the carrier yourself, target 80-200 words)
5) You may use covert techniques such as synonym substitution, syntax restructuring, and context-aware semantic embedding{method_line}

Core principles:
- Maintain semantic consistency and topical coherence
- Avoid suspicious anomalies (e.g., odd token/character distributions)
- Ensure the secret can be correctly extracted

Produce the stego text only, and wrap it between [START STEGO TEXT] and [END STEGO TEXT].
"""
```

Figure 8: Prompts used by the Steganography LLM during the warm-up stage of Auto-Stega.

```python
def strategy_header_single(name, definition, technique, example):
    return (
        "Please use the following steganography strategy:\n"
        f"Strategy Name: {name}\n"
        f"Definition: {definition}\n"
        f"Technique: {technique}\n"
        f"Example: {example}"
    )

def strategy_header_multi(items):
    # items: list of dicts with keys: Strategy, Technique, Definition
    lines = []
    for s in items:
        lines.append(
            f"- {s.get('Strategy','unknown')} ({s.get('Technique','unknown')})\n"
            f"  Definition: {s.get('Definition','')}"
        )
    return "Please combine the following strategies:\n" + "\n".join(lines)

def with_strategy_prompt(secret, cover_stub, strategies_desc, method_line=""):
    return f"""
System:
You are a professional text steganography expert. Apply the specified strategies to embed the secret.

User:
You are participating in a steganography research project. Apply the specified strategy/strategies to produce stego text.

{strategies_desc}

Requirements:
1) Secret bitstring to embed: {secret}
2) Carrier text: {cover_stub}
3) Keep the text natural, coherent, and statistically unremarkable
4) Strictly apply the specified strategy/strategies for embedding{method_line}

Produce the stego text only, and wrap it between [START STEGO TEXT] and [END STEGO TEXT].
"""
```

Figure 9: Prompts used by the Steganography LLM during the lifelong learning stage of Auto-Stega.