Breaking the "Provable Security": Detecting Finite-Precision Artifacts in LLM-based Steganography via Low-Probability Vanishing

Wenzhao Cao, Yaofei Wang*, Donghui Hu
Hefei University of Technology, China
wenzhao.cao@mail.hfut.edu.cn, wyf@hfut.edu.cn, hudh@hfut.edu.cn

Abstract

Recent advances in Large Language Models have fostered a new class of generative linguistic steganography, claim "provably secure" by theoretically aligning the steganographic distribution with the language model's natural distribution. We challenge this premise by exposing Low-Probability Vanishing (LPV), an inevitable vulnerability arising from finite-precision arithmetic. To exploit this, we propose RRNs-HT, a novel steganalysis framework based on Representative Random Numbers and Hypothesis Testing, which transforms the detection task from semantic classification to a statistical audit of the sampling mechanism. Crucially, unlike previous work that contrasts machine text against human text, we validate our method in a rigorous homologous setting to strictly isolate sampling artifacts. Experiments demonstrate that RRNs-HT effectively breaks the security of AC and Meteor with high detection accuracy, whereas state-of-the-art semantic steganalyzers degrade to random guessing. Our findings prove that theoretical security is unattainable in practice without addressing finite-precision leakage.

1 Introduction

Generative steganography has evolved from heuristic cover modification to a rigorous discipline rooted in information theory. With the advent of Large Language Models (LLMs), interval-based schemes such as Arithmetic Coding (AC) (Ziegler et al., 2019; Shen et al., 2020) and Meteor (Kaptchuk et al., 2021) have become a central route toward high-capacity linguistic steganography, inspiring many follow-up variants (Wang et al., 2023; Huang et al., 2024, 2025). These methods embed information directly during autoregressive sampling and are often described as "provably secure" because, in the ideal infinite-precision setting, the stego distribution P_S can match the cover distribution P_C, yielding $D_{\mathrm{KL}}(P_C \parallel P_S) \to 0$. Under this premise, the generated text should be statistically indistinguishable from ordinary model outputs and thus resistant to traditional semantic steganalysis.

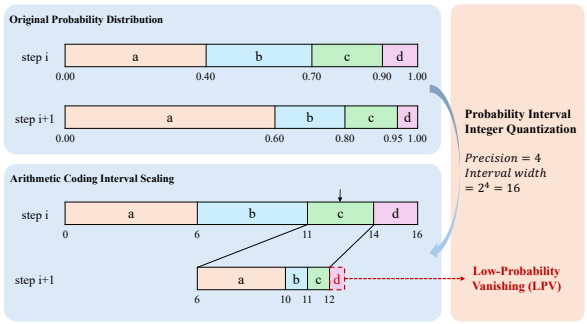


Figure 1: Illustration of the Low-Probability Vanishing (LPV) problem. Finite precision forces the truncation of the distribution's long tail, creating a statistical artifact.

However, a gap remains between this theoretical guarantee and deployed systems. Practical implementations must operate with finite-precision arithmetic, which quantizes probability intervals and can eliminate extremely small tails during interval scaling. We call this phenomenon Low-Probability Vanishing (LPV, Figure 1). Prior work has discussed precision-induced mismatch and mitigation strategies (Ding et al., 2023; Pan et al., 2025; Wang et al., 2025a), but the security implication of this tail loss has not been studied as a detection target under matched generation conditions. We argue that LPV is an implementation-level statistical side-channel: even when the ideal algorithm is secure on paper, its finite-precision realization systematically leaks information through the absence of low-probability tokens.

Detecting this leakage presents a unique challenge. Traditional linguistic steganalysis methods, such as R-BiLSTM-C (Niu et al., 2019) and syntax-aware SeSy (Yang et al., 2022), rely on capturing semantic incoherence or syntactic anoma-

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 20262–20274

lies. These methods operate effectively against modification-based steganography but fail against LLM-based generative methods for two reasons. First, interval-based steganographic algorithms typically generate high-probability tokens, leaving almost no semantic artifacts for neural classifiers to exploit. Second, prior evaluations often adopt a heterogeneous setting, training classifiers to distinguish "human-written text" from "machine-generated stego text". Such a protocol mostly measures generation mismatch rather than steganographic leakage. We therefore study a strict homologous setting in which cover and stego are generated with the same model family, tokenizer, precision, and decoding policy. In this setting, semantic steganalyzers degrade to random guessing because they remain blind to the subtle distributional truncation caused by LPV.

To expose this implementation-level vulnerability, we propose a novel steganalysis framework: Representative Random Numbers based Hypothesis Testing (RRNs-HT). Unlike semantic analyzers that examine what is generated, RRNs-HT examines how it is generated. Our insight is based on the inverse probability integral transform: if a sampling process faithfully follows the model's distribution, mapping the generated tokens back to their cumulative distribution function (CDF) values should yield a sequence of Representative Random Numbers (RRNs) that follows a standard uniform distribution $\mathcal{U}[0, 1)$. We show that LPV introduces a structural void near the tail of this RRN distribution. By transforming the detection problem from semantic classification to statistical auditing of the sampling mechanism, RRNs-HT identifies LPV with high confidence and makes the implementation gap directly measurable.

Our main contributions are as follows:

1. **LPV as an Implementation-Security Vulnerability**: We formalize LPV as the systematic tail loss induced by finite-precision interval quantization and show that it makes exact distribution matching impossible for AC-style interval steganography in practice.

2. **The RRNs-HT Framework**: We introduce a non-learning statistical detector that maps generated tokens into cumulative-probability space and exposes the resulting "invisible tail" via hypothesis testing.

3. **Rigorous Validation and Scope Analysis**: In a strict homologous setting, RRNs-HT reliably detects AC and Meteor while state-of-the-art semantic steganalyzers fail.

2 Related Work

2.1 Generative Steganography and Provable Security

The evolution of linguistic steganography has shifted from modification-based approaches (Mielikainen, 2006; Filler et al., 2011) to generation-based paradigms (Yang et al., 2019a; Chen et al., 2022; Ziegler et al., 2019). Early generative methods utilized RNNs or LSTMs to embed information during text generation (Fang et al., 2017; Yang et al., 2019b). Recently, the focus has moved towards provably secure steganography, which theoretically guarantees that the Kullback-Leibler (KL) divergence between the cover and stego distributions approaches zero.

A cornerstone of this class is AC-based steganography (Ziegler et al., 2019; Shen et al., 2020), which maps messages to the cumulative probability space of an LLM. To address the "randomness reuse" vulnerability in AC, Meteor (Kaptchuk et al., 2021) was proposed, employing standard interval sampling with re-encryption. Further advancements like ADG (Zhang et al., 2021), Discop (Ding et al., 2023), and iMEC (de Witt et al., 2023) introduce dynamic grouping or minimum entropy coupling to maintain distributional consistency. SparSamp (Wang et al., 2025a) further optimizes efficiency via sparse sampling. Pan et al. (2025) proposed a quantization optimization strategy to significantly minimize the KL divergence caused by rounding errors. However, most of these methods predominantly rely on the assumption of infinite-precision arithmetic or perfect sampling, a theoretical ideal that faces inevitable constraints in finite-precision implementation.

Our work is complementary to this line of research. Rather than proposing another mitigation, we study the security consequence of finite-precision mismatch itself. In particular, we focus on interval-based schemes that encode through shared cumulative intervals and ask whether the resulting tail distortion can be detected under a strict homologous protocol. To our knowledge, prior work has not formulated LPV as a detection-oriented vulnerability or validated such auditing in matched-model conditions.

2.2 Linguistic Steganalysis

Parallel to steganography, steganalysis has evolved from handcrafted features (Taskiran et al., 2006; Xiang et al., 2014) to deep learning-based detectors. State-of-the-art (SOTA) methods primarily focus on capturing semantic incoherence or syntactic anomalies introduced by embedding. TS-RNN (Yang et al., 2019d) and R-BiLSTM-C (Niu et al., 2019) utilize recurrent neural networks and dense connections to extract long-range semantic dependencies. SeSy (Yang et al., 2022) further integrates syntactic structure features using graph attention networks. Approaches such as SANet (Xue et al., 2024) utilize adaptive domain-invariant feature extraction, and CADA (Yang et al., 2025) employs class-aware adversarial adaptation to mitigate domain shifts.

While effective against modification-based or early generative methods, these semantic-driven analyzers struggle against modern LLM-based steganography. Since these LLM-based steganographic algorithms sample from the high-probability region of the model, the generated text maintains high linguistic fluency, leaving minimal semantic artifacts. Furthermore, prior evaluations often adopt a heterogeneous setting (distinguishing machine-generated stego from human-written cover), where classifiers inadvertently detect the generative artifacts of the LLM rather than the steganographic signal itself. In a rigorous homologous setting, these semantic classifiers often degrade to random guessing.

3 The Implementation Gap: LPV in Finite-Precision Sampling

In this section, we provide a theoretical analysis of the conflict between the mathematical assumption of infinite precision in generative steganography and the physical constraints of floating-point arithmetic. We demonstrate that this conflict inevitably leads to the *Low-Probability Vanishing* (LPV) phenomenon, transforming a numerical limitation into a detectable statistical side-channel.

3.1 The Illusion of Perfect Sampling

Generative steganographic methods, such as AC-based (Ziegler et al., 2019; Shen et al., 2020) and Meteor (Kaptchuk et al., 2021), rely on the *Inverse Probability Integral Transform* to achieve provable security. Let $\mathcal{V}$ be the vocabulary, and $P_\theta(w_t \mid h)$ be the conditional probability distribution predicted by an LLM given context h. Theoretically, these algorithms partition the continuous interval $[0, 1)$ into disjoint sub-intervals based on the cumulative distribution function (CDF) of P_θ. To encode a message, the algorithm selects a token w_t whose interval contains the target message bits. Ideally, if the sampling is perfectly faithful to the model, the resulting steganographic distribution $P_\mathcal{S}$ aligns exactly with the natural distribution $P_\mathcal{C}$, implying:

$$D_{\mathrm{KL}}(P_\mathcal{C} \parallel P_\mathcal{S}) = 0. \tag{1}$$

This condition is the cornerstone of "provable security" (Cachin, 1998), guaranteeing that the stego-text is statistically indistinguishable from natural text.

3.2 The Finite-Precision Constraint

However, practical implementations of these algorithms operate on von Neumann architectures constrained by finite-precision floating-point arithmetic. The precise interval of every token in $\mathcal{V}$ is computationally infeasible when the vocabulary size is large (e.g., $|\mathcal{V}| \approx 50$K for GPT-2).

Specifically, standard floating-point formats (e.g., FP32 or FP16) possess a machine epsilon ϵ_{mach}, representing the smallest difference between two representable values. Both AC and Meteor embed messages by matching shared prefixes of probability intervals, necessitating a mapping from the continuous probability space $[0, 1)$ to a discrete integer interval. Consequently, when the probability of a token $w \in \mathcal{V}$ falls below a certain threshold derived from this integer quantization, numerical underflow occurs. To prevent decoding errors and maintain the validity of the arithmetic coding intervals, steganographic algorithms are forced to perform tail truncation and renormalization. Notably, AC employs frequent interval scaling to maintain precision, which exacerbates the quantization error, making the LPV problem significantly more severe than in Meteor.

3.3 Formalizing LPV

We define LPV as the systematic removal of tokens located in the long tail of the probability distribution due to the quantization limits of interval scaling.

Let $\mathcal{I}(w)$ be the length of the interval assigned to token w in the cumulative probability space. In finite-precision arithmetic, any token whose scaled interval length is smaller than the minimum representable precision δ cannot be encoded. We define

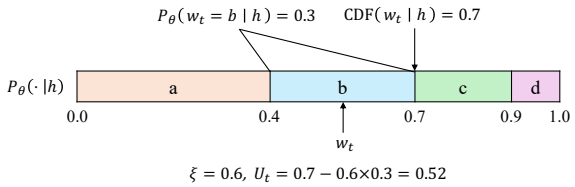


Figure 2: A calculation example of Representative Random Number (RRN). The U_t is sampled uniformly from the cumulative probability interval.

the set of vanishing tokens $\mathcal{V}_{\text{lpv}}$ as:

$$\mathcal{V}_{\text{lpv}} = \{w \in \mathcal{V} \mid P_\theta(w \mid h) < \delta\}, \tag{2}$$

where δ is determined by the implementation precision (e.g., 2^{-32} for standard AC implementations).

Consequently, the steganographic algorithm samples from a truncated and renormalized distribution $\hat{P}$, rather than the true distribution P_θ. The probability mass lost to truncation is:

$$\epsilon_{\text{loss}} = \sum_{w \in \mathcal{V}_{\text{lpv}}} P_\theta(w \mid h). \tag{3}$$

The effective steganographic distribution $\hat{P}$ is thus:

$$\hat{P}(w \mid h) = \begin{cases} \dfrac{P_\theta(w \mid h)}{1 - \epsilon_{\text{loss}}}, & \text{if } w \notin \mathcal{V}_{\text{lpv}}, \\ 0, & \text{if } w \in \mathcal{V}_{\text{lpv}}. \end{cases} \tag{4}$$

Crucially, since $\epsilon_{\text{loss}} > 0$ for any heavy-tailed distribution (which is characteristic of natural language), it follows that $D_{\text{KL}}(P_\theta \parallel \hat{P}) > 0$.

3.4 LPV as a Statistical Side-Channel

While recent works like Discop (Ding et al., 2023) and iMEC (de Witt et al., 2023) attempt to address distribution mismatches via distribution copies or minimum entropy coupling, and Pan et al. (2025) significantly reduce the statistical distance by calibrating quantization distortion, standard AC-based methods (Ziegler et al., 2019), interval-based methods (Kaptchuk et al., 2021), and their optimized variants inevitably suffer from the distribution shift described in Eq. (4).

Although the semantic impact of removing extremely low-probability tokens is negligible (hence fooling semantic analyzers (Yang et al., 2022, 2023; Wang et al., 2025b)), the *statistical* impact is permanent. The renormalization process creates a structural artifact: the cumulative distribution function (CDF) of the generated stegotext never reaches the true theoretical limit of 1.0 relative to the original model.

In the next section, we demonstrate how to exploit this "invisible tail" using a statistical transformation, turning this implementation flaw into a reliable detection fingerprint.

Algorithm 1: RRNs Sampling Process

Input: Sequence of tokens $S = \{w_1, \ldots, w_n\}$, Reference language model P_θ
Output: RRNs sequence $\mathcal{U}_S = \{U_1, U_2, \ldots, U_n\}$

$U \leftarrow []$;
for $i = 1$ **to** n **do**
  $P^{(i)} \leftarrow P_\theta(\cdot \mid w_{<i})$;
  Determine token ordering $\text{idx}(\cdot)$ by sorting $P^{(i)}$ descendingly;
  $\text{CDF}_i \leftarrow \sum_{v \in \mathcal{V}, \text{idx}(v) \le \text{idx}(w_i)} P_\theta(v \mid w_{<i})$;
  Sample $\xi_i \sim \mathcal{U}[0, 1)$;
  $U_i \leftarrow \text{CDF}_i - \xi_i \cdot P_\theta(w_i \mid w_{<i})$;
  Append U_i to $\mathcal{U}_S$;
return $\mathcal{U}_S$;

Algorithm 2: RRNs-HT Process

Input: RRNs sequence $\mathcal{U}_S = \{U_1, \ldots, U_n\}$, Sequence of discarded probability masses $\mathcal{E} = \{\epsilon_{\text{loss}}^{(1)}, \ldots, \epsilon_{\text{loss}}^{(n)}\}$
Output: Z-score Z

$Y \leftarrow \sum_{i=1}^{n} \mathbb{I}[U_i \ge 1 - \epsilon_{\text{loss}}^{(i)}]$;
$\mu_Y \leftarrow \sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)}$;
$\sigma_Y^2 \leftarrow \sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)}(1 - \epsilon_{\text{loss}}^{(i)})$;
if $\sigma_Y^2 > 0$ **then**
  $Z \leftarrow (\mu_Y - Y)/\sqrt{\sigma_Y^2}$;
else
  $Z \leftarrow 0$;
return Z;

4 RRNs-HT: A Statistical Side-Channel Attack via Inverse Sampling

Based on the theoretical analysis in Section 3, we propose a novel steganalysis framework: **Representative Random Numbers based Hypothesis Testing (RRNs-HT)**. Unlike prior linguistic steganalysis methods (Yang et al., 2019d; Niu et al.,

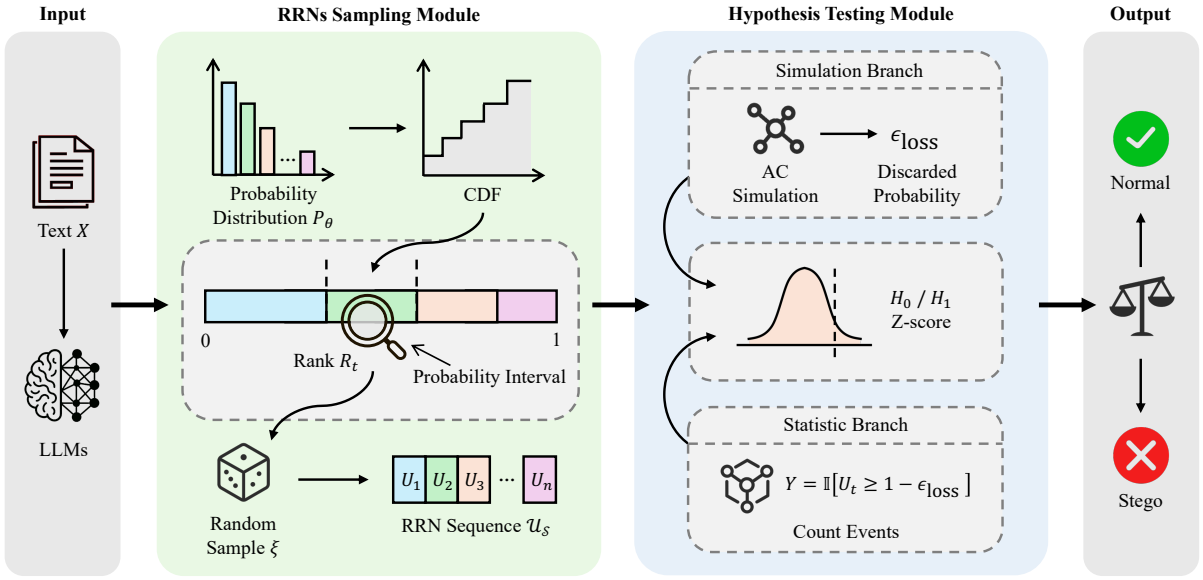


Figure 3: The overall framework of RRNs-HT. It consists of two stages: (1) the RRNs Sampling Module transforms discrete text into continuous signals, and (2) the Hypothesis Testing Module detects statistical anomalies (LPV) by calculating the Z-score deviation from the standard uniform distribution.

2019) that search for semantic incoherence in the generated text, RRNs-HT operates as a *statistical side-channel attack* against the steganography mechanism itself. It exploits the LPV vulnerability to distinguish between the theoretical distribution of the LLM and the actual truncated distribution used by steganography.

4.1 Probability Integral Transform

The core intuition of our approach relies on the universality of the Probability Integral Transform (PIT). Let X be a random variable with a continuous cumulative distribution function (CDF) F_X. It is a fundamental property of statistics that the random variable $U = F_X(X)$ follows a standard uniform distribution $U \sim \mathcal{U}[0, 1)$.

In the context of autoregressive text generation, for a given context h, the model outputs a discrete probability distribution over the vocabulary $\mathcal{V}$. If a token w_t is sampled faithfully according to the model's distribution $P_\theta(\cdot \mid h)$, we can map w_t back to the continuous probability space. We define the **Representative Random Number (RRN)**, denoted as U_t, for a token w_t as follows:

$$U_t = \text{CDF}(w_t \mid h) - \xi \cdot P_\theta(w_t \mid h), \quad (5)$$

where $\text{CDF}(w_t \mid h) = \sum_{v \in \mathcal{V}, \text{idx}(v) \leq \text{idx}(w_t)} P_\theta(v \mid h)$, and $\xi \sim \mathcal{U}[0, 1)$ is a random noise term introduced to smooth the distribution. Figure 2 shows a calculation example of RRN.

Proposition 1. *If a sequence of tokens $S = \{w_1, \ldots, w_n\}$ is generated by pure random sampling from the LLM P_θ (i.e., Cover), the corresponding sequence of RRNs $\mathcal{U}_S = \{U_1, \ldots, U_n\}$ converges to the standard uniform distribution $\mathcal{U}[0, 1)$ as $n \to \infty$.* The rigorous proof is provided in Appendix A.

4.2 The LPV Fingerprint in RRN Domain

While the semantic quality of steganographic text (Stego) generated by methods like AC (Ziegler et al., 2019) or Meteor (Kaptchuk et al., 2021) is indistinguishable from cover text, their statistical footprint in the RRN domain is distinct. As proven in Section 3, finite-precision arithmetic forces these algorithms to truncate the tail of the distribution. This truncation creates a structural "void" in the probability space.

Specifically, for a truncation threshold δ induced by floating-point precision, any token w with cumulative probability in the range $[1 - \epsilon_{\text{loss}}, 1)$ becomes unsampleable. Consequently, the distribution of RRNs for Stego text, denoted as $P_{\mathcal{S}}(u)$, exhibits a **hard cut-off** at the tail:

$$P_{\mathcal{S}}(u) = 0, \quad \forall u \in [1 - \epsilon_{\text{loss}}, 1). \quad (6)$$

This phenomenon, which we term the *Invisible Tail*, is illustrated in Figure 1. While semantic analyzers fail to notice the absence of these extremely rare tokens, RRNs-HT detects the statistical anomaly of the empty interval.

4.3 Hypothesis Testing Construction

We formulate the detection task as a statistical hypothesis testing problem. Given a suspect text S, we compute its RRN sequence $\mathcal{U}_S$ and test the following hypotheses:

- **Null Hypothesis (H_0):** $\mathcal{U}_S \sim \mathcal{U}[0,1)$. The text is natural (Cover).
- **Alternative Hypothesis (H_1):** $\mathcal{U}_S \not\sim \mathcal{U}[0,1)$ with significant tail vanishing. The text contains hidden information (Stego).

For each RRN u_i in the sequence (where $i = 1, \ldots, n$), let $\epsilon_{\text{loss}}^{(i)}$ denote the total discarded probability mass defined in Eq. (3). We define a Bernoulli random variable $X_i = \mathbb{I}[U_i \geq 1 - \epsilon_{\text{loss}}^{(i)}]$ to indicate whether the i-th RRN falls into the low-probability interval. Under H_0, since $U_S \sim \mathcal{U}[0,1)$, X_i follows a Bernoulli distribution with parameter $\epsilon_{\text{loss}}^{(i)}$. Its expectation is $\mathbb{E}[X_i] = \epsilon_{\text{loss}}^{(i)}$, and its variance is $\text{Var}(X_i) = \epsilon_{\text{loss}}^{(i)}(1 - \epsilon_{\text{loss}}^{(i)})$. For a sequence of length n, we count the total observed intrusions into the low-probability interval: $Y = \sum_{i=1}^{n} X_i$. By the Central Limit Theorem, for sufficiently large n, Y approximates a normal distribution with expectation and variance:

$$\mu_Y = \mathbb{E}[Y] = \sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)}, \tag{7}$$

$$\sigma_Y^2 = \text{Var}(Y) = \sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)}(1 - \epsilon_{\text{loss}}^{(i)}). \tag{8}$$

We construct the standardized Z-score as the final test statistic:

$$Z = \frac{\mu_Y - Y}{\sigma_Y} = \frac{\sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)} - Y}{\sqrt{\sum_{i=1}^{n} \epsilon_{\text{loss}}^{(i)}(1 - \epsilon_{\text{loss}}^{(i)})}}. \tag{9}$$

This Z-score quantifies the deviation of the observed count from the expected value under H_0. We reject the null hypothesis H_0 when $Z > 1.645$ (corresponding to a significance level of $\alpha = 0.05$). A rejection implies that the text sequence was likely generated by a method suffering from LPV. This targeted approach offers superior sensitivity compared to generic goodness-of-fit tests (e.g., Chi-square), enabling the precise detection of fine-grained statistical anomalies. The Z-score calculation process is summarized in Algorithm 2.

By relying on these statistical artifacts rather than learned semantic features, RRNs-HT functions as a zero-shot detector that does not require training on specific steganographic algorithms. In the current paper, we position it as a targeted white-box auditing tool for interval-based methods that exhibit LPV, rather than as a universal detector for every generative steganography setting.

5 Experimental Evaluation

5.1 Experimental Setup

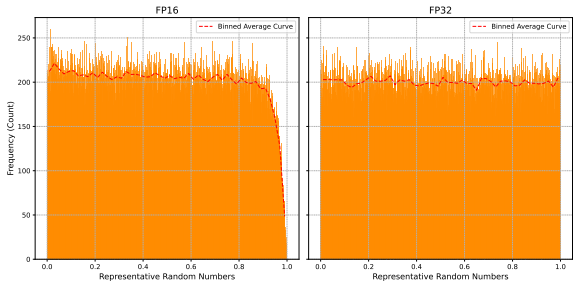
To validate our theoretical analysis, we designed a comprehensive set of verification experiments.

Models. We employ Qwen2.5 (Qwen Team, 2024) and Llama3.2 (Meta AI, 2024) as the backbone language models. Unless otherwise specified, we primarily report the results on Qwen2.5 in the main text to avoid redundancy, as Llama3.2 exhibits identical statistical trends.

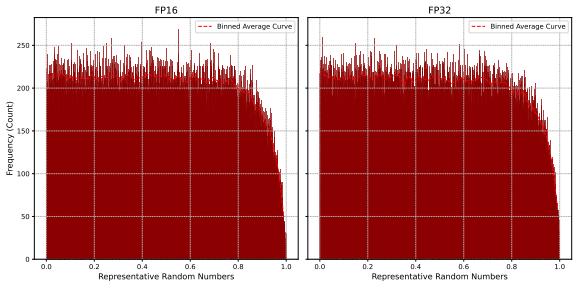
Datasets. Experimental contexts are sourced from three widely used datasets: Movie (Maas et al., 2011), Twitter (Go et al., 2009), and News (Thompson, 2017).

Baselines. We selected a range of representative text steganography methods for comparative analysis, including AC/AC-k (Ziegler et al., 2019), Meteor (Kaptchuk et al., 2021), ADG (Zhang et al., 2021), Discop (Ding et al., 2023), iMEC (de Witt et al., 2023), and SparSamp (Wang et al., 2025a). Specifically, to rigorously analyze the LPV problem, we distinguish between three configurations: AC-k follows the original implementation in AC (Ziegler et al., 2019) utilizing dynamic k truncation; AC is a modified version without applying explicit top-k truncation, allowing us to isolate the effects of numerical precision from the sampling strategy; and Normal serves as the ideal baseline employing standard autoregressive random sampling without truncation, theoretically yielding an RRNs distribution closest to uniform. In the main-text steganalysis comparison, we focus on AC and Meteor because they are the representative interval-based methods that most directly expose the LPV artifact targeted by RRNs-HT.

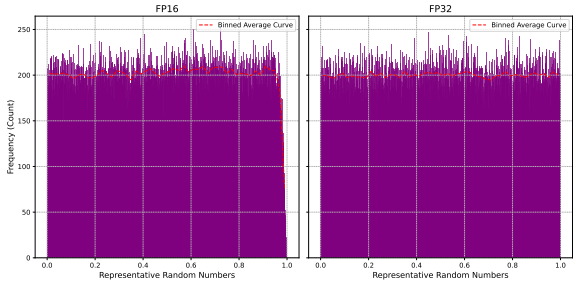
Configuration. The language model temperature was set to 1.0, and the candidate pool size for generation was fixed at top-$k = 300$ for method AC-k. We evaluated quantization precisions of



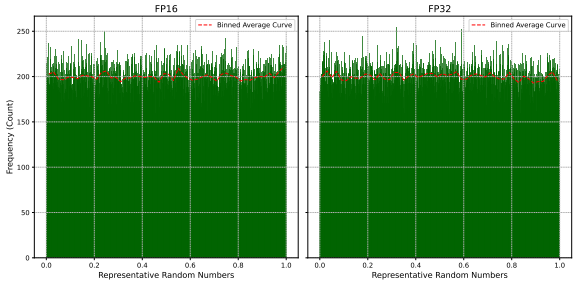
(a) AC: Sharp drop at tail due to precision limit.



(b) AC-k: Truncation caused by dynamic k.



(c) Meteor: Delayed tail drop.



(d) Normal: Strictly uniform distribution (Baseline).

Figure 4: Frequency histograms of RRNs across different methods. The x-axis represents the RRN value range $[0, 1)$, and the y-axis represents frequency. (a-c) The AC-based and Meteor methods exhibit significant tail truncation (refer to the sharp drop at the right end) caused by the LPV effect. (d) In contrast, the Normal baseline maintains a consistent uniform distribution, serving as the ground truth.

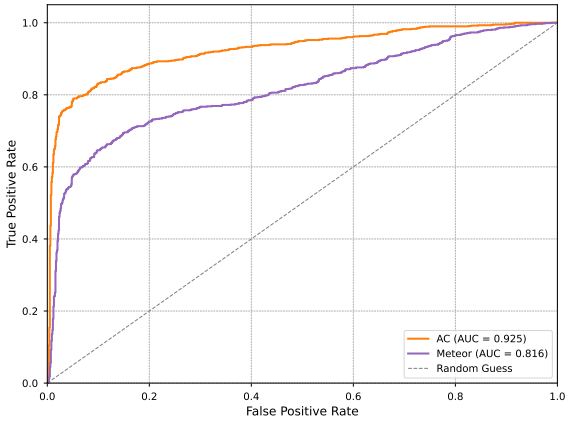


Figure 5: ROC curves of the proposed RRNs-HT framework detecting various steganographic methods. The results demonstrate high detection performance against AC (AUC=0.925) and Meteor (AUC=0.816).

$p = \{\text{FP16}, \text{FP32}\}$. Robustness under varying top-k and top-p values is summarized in Table 3.

Homologous audit protocol. To isolate implementation artifacts from generation mismatch, cover and stego text are produced under identical model family, tokenizer, floating-point precision, and global decoding policy. When we study top-k or top-p, the same setting is applied to both Normal and Stego. Consequently, any deviation detected by RRNs-HT is attributable to the additional loss induced by steganographic interval scaling rather than to a different generator or decoding configuration.

Metrics. We employ two primary metrics: (1) a qualitative assessment via RRNs frequency distribution histograms, and (2) a quantitative evaluation measuring the deviation of the RRNs distribution from the standard uniform distribution using statistical hypothesis testing.

5.2 Comparison of RRNs Distributions

Qualitative Assessment. We first analyze the RRNs frequency histograms. As shown in Figure 4d, the Normal method exhibits a flat distribution across $[0, 1)$, confirming that unbiased generation yields a uniform RRNs sequence. In contrast, AC-k displays abrupt truncation across all precisions (Figure 4b). This stems from the combined effect of explicit k-truncation and precision thresholds, creating a substantial "unreachable interval" that constitutes a significant statistical anomaly. For AC at low precision (Figure 4a), the histogram reveals a sharp tail drop solely attributable to precision limitations and probability interval scaling. However, at high precision, the reduced probability mass of discarded tokens makes the macroscopic

FP	l_m	Movie			Twitter			News		
		Normal	AC	Meteor	Normal	AC	Meteor	Normal	AC	Meteor
16	50	0.4182	1.2894	0.2339	0.4151	1.7902	0.3250	0.3937	1.4953	0.1733
	500	0.4103	4.0032	1.9782	0.9327	4.2897	1.2561	0.8245	4.0745	0.6406
	2000	0.3826	8.2148	2.2223	0.8381	7.3403	1.1473	0.2930	8.7280	0.7620
	5000	0.4430	12.7333	3.5805	0.8340	12.8933	1.5848	0.6828	13.5449	1.9922
	8000	0.5687	16.3242	3.4825	0.8272	16.5185	1.3051	0.5202	18.1691	2.1975
32	50K	0.1935	1.2034	0.3757	0.9720	0.9315	0.2921	0.1910	1.2143	0.9431
	100K	0.5175	1.7710	0.5100	0.4104	1.2605	0.4651	0.4949	1.6745	0.9452
	150K	0.4130	2.0259	0.6081	0.5143	1.5647	0.9623	0.4018	2.1394	1.0451
	200K	0.1386	2.2395	1.6713	0.5356	1.6865	1.5283	0.5146	2.3843	1.0944
	250K	0.7471	2.3749	2.2836	0.8810	1.9196	1.5826	0.5176	2.6544	1.1873

Table 1: Z-scores of statistical testing across different floating-point precisions (FP) and sequence lengths (l_m). Highlighted values indicate scores exceeding the significance threshold ($Z > 1.645$ at $\alpha = 0.05$), highlighting significant LPV artifacts detected by RRNs-HT.

Method	AC	Meteor
FCN	51.38%	50.83%
R-BiLSTM-C	49.90%	49.70%
BiLSTM-Dense	50.15%	50.50%
BERT-LSTM-ATT	49.72%	50.65%
SeSy	50.52%	50.40%
RRNs-HT (ours)	86.95%	77.40%

Table 2: Comparison of detection accuracy between RRNs-HT and steganalysis baselines on AC and Meteor. Values are averaged across three datasets. RRNs-HT is highly effective on these representative interval-based methods with LPV.

Impact of top-k					
top-k	300	1000	3000	10000	30000
AC (FP16)	0.9708	1.3159	4.3214	4.9120	5.1539
AC (FP32)	0.0167	0.0520	0.0379	0.1866	0.2604
Normal (FP16)	0.6603	0.8922	0.5472	0.4781	0.5244
Normal (FP32)	0.0191	0.0280	0.0567	0.1782	0.2831
Impact of top-p					
top-p	0.80	0.90	0.95	0.98	0.99
AC (FP16)	0.5680	1.4093	2.4103	3.7554	4.3940
AC (FP32)	0.5065	0.4886	0.4946	0.4848	0.5241
Normal (FP16)	0.2500	0.4535	0.7269	0.8306	0.6130
Normal (FP32)	0.4537	0.4854	0.5123	0.4607	0.4788

Table 3: Z-scores of RRNs-HT under different global decoding policies. Legitimate top-k/top-p truncation alone does not trigger large deviations on the Normal baseline, while AC remains detectable when LPV is present.

distribution appear quasi-uniform, masking this microscopic artifact. Finally, Meteor (Figure 4c) exhibits a delayed downward trend in the tail due to the absence of interval scaling. For qualitative examples of the generated text and their corresponding Z-scores, please refer to Table 5 in Appendix C.

Quantitative Assessment. Table 1 details the Z-scores across three diverse datasets (Movie, Twitter, and News). Overall, the detection accuracy exhibits a clear positive correlation with the sequence length (l_m); as the generation length increases, the cumulative statistical deviation becomes more significant, leading to higher Z-scores. We highlight the high practicality of our method at FP16 precision, where robust detection is achievable with extremely limited data. Specifically, AC triggers the significance threshold ($\alpha = 0.05, Z > 1.645$) with only 50 tokens in the Twitter domain (1.7902). The detection at FP32 is more challenging due to the finer granularity, typically necessitating significantly longer sequences. We observed consistent phenomena on Llama3.2, confirming that the LPV vulnerability is model-agnostic (detailed results are provided in Appendix B).

5.3 Robustness to Top-p and Top-k Decoding

An important practical question is whether RRNs-HT simply reacts to legitimate decoding truncation. Table 3, where each entry uses a 100-token sequence, shows that it does not. Under matched global top-k or top-p settings, the Normal baseline remains consistently below the detection threshold, whereas AC under FP16 becomes increasingly de-

tectable as more tail mass is exposed. For example, under global top-k, the AC score rises from 0.9708 at $k = 300$ to 5.1539 at $k = 30000$, and under global top-p, it exceeds the significance threshold from $p = 0.95$ onward. By contrast, the FP32 rows remain low in this fixed-length table because shorter sequences accumulate less evidence, consistent with the longer-text requirement already observed in Table 1. This confirms that RRNs-HT is not triggered by intended truncation itself; it responds to the additional tail loss created during steganographic interval scaling.

5.4 Computational cost and deployment scope

RRNs-HT requires one teacher-forced forward pass over the suspect sequence to recover the per-position reference distributions, followed by probability-space post-processing to compute RRNs and the corresponding test statistic. This is more expensive than a lightweight text classifier, but it differs from autoregressive generation in one crucial respect: the forward pass is parallel over sequence positions and does not require retraining. We therefore view RRNs-HT as an offline or batch auditing tool in white-box settings. Under model or framework mismatch, practitioners should calibrate thresholds on cover-only samples rather than transferring a universal cutoff across systems.

5.5 Comparison with Steganalysis Models

To benchmark our approach against state-of-the-art baselines, we evaluated a diverse set of steganalysis models, specifically FCN (Yang et al., 2019c), R-BiLSTM-C (Niu et al., 2019), BiLSTM-Dense (Yang et al., 2020), BERT-LSTM-ATT (Zou et al., 2021), and SeSy (Yang et al., 2022). Table 2 presents the comparative detection accuracy on AC and Meteor. Observation reveals that baseline methods achieve accuracies approximating 50% across all testing schemes, performing no better than random guessing. In stark contrast, RRNs-HT demonstrates remarkable effectiveness on AC and Meteor, attaining detection accuracies of 86.95% and 77.40%, respectively. The ROC curves in Figure 5 further corroborate the performance of RRNs-HT.

6 Conclusion

This paper studies the implementation-level security of interval-based LLM steganography. We formulate Low-Probability Vanishing (LPV) as the tail loss created by finite-precision interval quantization and show that RRNs-HT converts this leakage into a measurable statistical fingerprint. Experiments in strict homologous settings demonstrate strong detection on AC and Meteor, while semantic steganalyzers remain near chance, underscoring that RRNs-HT is a targeted audit rather than a generic text classifier. These results narrow the gap between theoretical distribution matching and deployed security by showing that finite precision can invalidate the practical indistinguishability promised by idealized analysis.

Limitations

We intentionally scope this paper to strict homologous auditing, where the detector has access to the matched reference distribution and the same tokenizer, precision, and decoding policy as the generator. This white-box assumption enables a clean study of implementation leakage, but it also means RRNs-HT cannot be transferred directly to arbitrary deployment systems. In mismatched settings, practitioners would need system-specific treatment of tokenizer mismatch, policy uncertainty, and threshold calibration rather than reusing a universal threshold from the current paper.

RRNs-HT is designed to detect LPV caused by interval-based steganographic sampling. Accordingly, it is highly effective on AC-style methods but intentionally shows limited sensitivity on methods that do not induce the same tail artifact. This specificity is a feature for auditing LPV, but it also means the framework should not be interpreted as a universal detector for all linguistic steganography.

Finally, detection strength depends on both precision and sample size. FP16 exposes LPV with short texts, whereas FP32 typically requires longer sequences to accumulate sufficient evidence. Together with the need for a teacher-forced forward pass, this makes the current framework better suited to offline auditing than to real-time filtering.

Ethical Considerations

This paper studies both the failure modes of linguistic steganography and a corresponding detection mechanism. As with most security research, the contribution is dual-use: the analysis may help defenders audit deployed systems, but it may also inform steganography designers about where current implementations leak information. We therefore present RRNs-HT as an auditing framework for controlled research and defensive evaluation, rather than as a tool for indiscriminate surveillance.

Acknowledgments

The authors thank the reviewers for their valuable comments. This work was supported in part by the Natural Science Foundation of China under Grant 62302146.

References

Christian Cachin. 1998. An information-theoretic model for steganography. *Information and Computation*, 192:41–56.

Kejiang Chen, Hang Zhou, Hanqing Zhao, Dongdong Chen, Weiming Zhang, and Nenghai Yu. 2022. Distribution-preserving steganography based on text-to-speech generative models. *IEEE Transactions on Dependable and Secure Computing*, 19(5):3343–3356.

Christian Schroeder de Witt, Samuel Sokota, J Zico Kolter, Jakob Nicolaus Foerster, and Martin Strohmeier. 2023. Perfectly secure steganography using minimum entropy coupling. In *The Eleventh International Conference on Learning Representations*.

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)*, pages 2238–2255.

Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017. Generating steganographic text with LSTMs. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106, Vancouver, Canada. Association for Computational Linguistics.

Tomáš Filler, Jan Judas, and Jessica Fridrich. 2011. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Transactions on Information Forensics and Security*, 6(3):920–935.

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

Yu-Shin Huang, Peter Just, Krishna Narayanan, and Chao Tian. 2024. Od-stega: Llm-based near-imperceptible steganography via optimized distributions. *arXiv preprint arXiv:2410.04328*.

Yu-Shin Huang, Chao Tian, Krishna Narayanan, and Lizhong Zheng. 2025. Relatively-secure llm-based steganography via constrained markov decision processes. In *2025 IEEE International Symposium on Information Theory (ISIT)*, pages 1–6.

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*, CCS '21, page 1529–1548, New York, NY, USA. Association for Computing Machinery.

Andrew L. 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, Portland, Oregon, USA. Association for Computational Linguistics.

Meta AI. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

J. Mielikainen. 2006. Lsb matching revisited. *IEEE Signal Processing Letters*, 13(5):285–287.

Yan Niu, Juan Wen, Ping Zhong, and Yiming Xue. 2019. A hybrid r-bilstm-c neural network based text steganalysis. *IEEE Signal Processing Letters*, 26(12):1907–1911.

Chao Pan, Donghui Hu, Yaofei Wang, Kejiang Chen, Yinyin Peng, Xianjin Rong, Chen Gu, and Meng Li. 2025. Rethinking prefix-based steganography for enhanced security and efficiency. *IEEE Transactions on Information Forensics and Security*, 20:3287–3301.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Jiaming Shen, Heng Ji, and Jiawei Han. 2020. Near-imperceptible neural linguistic steganography via self-adjusting arithmetic coding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 303–313, Online. Association for Computational Linguistics.

CM Taskiran, U Topkara, M Topkara, and EJ Delp. 2006. Attacks on lexical natural language steganography systems. In *Security, Steganography, and Watermarking of Multimedia Contents VIII*, volume 6072, pages 97–105. Proceedings of the SPIE.

Andrew Thompson. 2017. All the News. Kaggle dataset.

Rong Wang, Lingyun Xiang, Yangfan Liu, and Chunfang Yang. 2023. Png-stega: Progressive non-autoregressive generative linguistic steganography. *IEEE Signal Processing Letters*, 30:528–532.

Yaofei Wang, Gang Pei, Kejiang Chen, Jinyang Ding, Chao Pan, Weilong Pang, Donghui Hu, and Weiming Zhang. 2025a. Sparsamp: Efficient provably secure steganography based on sparse sampling. In *34th USENIX Security Symposium (USENIX Security 25)*.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):25461–25469.

Lingyun Xiang, Xingming Sun, Gang Luo, and Bin Xia. 2014. Linguistic steganalysis using the features derived from synonym frequency. *Multimedia Tools and Applications*, 71(3):1893–1911.

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*, IH&MMSec '20, page 5–10, New York, NY, USA. Association for Computing Machinery.

Jinshuai Yang, Zhongliang Yang, Siyu Zhang, Haoqin Tu, and Yongfeng Huang. 2022. Sesy: Linguistic steganalysis framework integrating semantic and syntactic features. *IEEE Signal Processing Letters*, 29:31–35.

Jinshuai Yang, Zhongliang Yang, Jiajun Zou, Haoqin Tu, and Yongfeng Huang. 2023. Linguistic steganalysis toward social network. *IEEE Transactions on Information Forensics and Security*, 18:859–871.

Kuan Yang, Kejiang Chen, Weiming Zhang, and Nenghai Yu. 2019a. Provably secure generative steganography based on autoregressive model. In *Digital Forensics and Watermarking*, pages 55–68, Cham. Springer International Publishing.

Zhen Yang, Yufei Luo, Jinshuai Yang, Xin Xu, Ru Zhang, and Yongfeng Huang. 2025. Class-aware adversarial unsupervised domain adaptation for linguistic steganalysis. *IEEE Transactions on Information Forensics and Security*, 20:5181–5194.

Zhong-Liang Yang, Xiao-Qing Guo, Zi-Ming Chen, Yong-Feng Huang, and Yu-Jin Zhang. 2019b. Rnn-stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295.

Zhongliang Yang, Yongfeng Huang, and Yu-Jin Zhang. 2019c. A fast and efficient text steganalysis method. *IEEE Signal Processing Letters*, 26(4):627–631.

Zhongliang Yang, Ke Wang, Jian Li, Yongfeng Huang, and Yu-Jin Zhang. 2019d. Ts-rnn: Text steganalysis based on recurrent neural networks. *IEEE Signal Processing Letters*, 26(12):1743–1747.

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, Online. Association for Computational Linguistics.

Zachary Ziegler, Yuntian Deng, and Alexander Rush. 2019. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215, Hong Kong, China. Association for Computational Linguistics.

Jiajun Zou, Zhongliang Yang, Siyu Zhang, Sadaqat ur Rehman, and Yongfeng Huang. 2021. High-performance linguistic steganalysis, capacity estimation and steganographic positioning. In *Digital Forensics and Watermarking*, pages 80–93, Cham. Springer International Publishing.

A Theoretical Proof of RRNs Distribution

In this section, we provide the rigorous derivation of the probability density function (PDF) of the Representative Random Number U_i, proving the necessary and sufficient condition for its uniformity.

Derivation. Recall the definition of U_i in Eq. (5). Let $P_{\mathcal{S}}^{(i)}$ denote the actual sampling distribution of the steganographic algorithm (Stego distribution), and $P_{\mathcal{C}}^{(i)}$ denote the reference distribution derived from the language model $P_\theta(\cdot \mid x_{<i})$ (Cover distribution).

Let $\mathrm{CDF}_{(k)}^{(i)} = \sum_{j=1}^{k} P_{\mathcal{C},(j)}^{(i)}$ be the cumulative probability sum up to rank k (with $\mathrm{CDF}_{(0)}^{(i)} \triangleq 0$), where $P_{\mathcal{C},(j)}^{(i)}$ is the probability of the j-th token in the descendingly sorted vocabulary of the cover distribution. Based on the construction $U_i = \mathrm{CDF}_{(R_i)}^{(i)} - \xi_i \cdot P_{\mathcal{C},(R_i)}^{(i)}$, the variable U_i falls within the interval $(\mathrm{CDF}_{(R_i-1)}^{(i)}, \mathrm{CDF}_{(R_i)}^{(i)}]$, where R_i is the rank of the chosen token.

The probability density function $f_{U_i}(u)$ for $u \in [0, 1)$ can be derived using the law of total probability. Since U_i is conditionally uniform given R_i, its density is the weighted sum of uniform densities on disjoint intervals:

$$f_{U_i}(u) = \sum_{k \in \mathcal{V}} P_{\mathcal{S}}^{(i)}(k) \cdot f_{U_i \mid R_i = k}(u). \tag{10}$$

Given that $f_{U_i|R_i=k}(u)$ is the density of $\mathcal{U}(\mathrm{CDF}_{(k-1)}^{(i)}, \mathrm{CDF}_{(k)}^{(i)}]$, its value is $\frac{1}{P_{\mathcal{C},(k)}^{(i)}}$ if u falls in the interval, and 0 otherwise. Thus, the marginal density becomes:

FP	l_m	Movie			Twitter			News		
		Normal	AC	Meteor	Normal	AC	Meteor	Normal	AC	Meteor
16	50	0.3833	1.1689	0.4161	0.4118	1.8401	0.3449	0.3340	1.5688	0.3607
	500	0.7630	4.8949	1.1202	0.7854	5.8657	1.6263	0.9412	4.9313	1.3980
	2000	0.7054	9.3933	1.2139	0.5167	10.2380	1.2581	0.8465	9.2893	1.3193
	5000	0.8703	15.2051	1.2267	0.7664	15.3215	1.3377	0.8373	15.5240	1.3841
	8000	0.8342	19.2277	1.4774	0.7343	21.5453	1.3491	0.8661	19.1908	1.6534
32	50K	0.3521	1.3592	0.4296	0.4321	1.3394	0.4205	0.4261	1.4540	0.4428
	100K	0.6202	1.9590	0.6156	0.5964	1.9817	0.5829	0.6367	1.9329	0.6296
	150K	0.7538	2.4542	0.7801	0.6954	2.4047	0.6835	0.7733	2.3743	0.7321
	200K	0.8619	2.8613	0.8623	0.8402	2.8713	0.8492	0.6085	2.8548	0.8435
	250K	0.7023	3.1839	0.9124	0.8741	3.2099	0.7922	0.7733	3.3215	1.0014

Table 4: Z-scores under different methods and datasets on Llama3.2. Higher Z-scores (> 1.645) indicate significant deviation from the uniform distribution, revealing the presence of LPV artifacts.

$$f_{U_i}(u) = \sum_{k\geq 1} \frac{P^{(i)}_{\mathcal{S},(k)}}{P^{(i)}_{\mathcal{C},(k)}} \cdot \mathbb{I}\left[u \in (\mathrm{CDF}^{(i)}_{(k-1)}, \mathrm{CDF}^{(i)}_{(k)}]\right], \tag{11}$$

where $\mathbb{I}[\cdot]$ is the indicator function. Eq. (11) shows that $f_{U_i}(u)$ is a piecewise constant function (a step function).

Proof of Uniformity. We aim to prove the condition for $U_i \sim \mathcal{U}[0,1)$. The standard uniform distribution requires $f_{U_i}(u) = 1$ for all $u \in [0,1)$. From Eq. (11), this equality holds if and only if the ratio $\frac{P^{(i)}_{\mathcal{S},(k)}}{P^{(i)}_{\mathcal{C},(k)}} = 1$ for all k. This leads to the fundamental condition:

$$U_i \sim \mathcal{U}[0,1) \iff P^{(i)}_{\mathcal{S}} = P^{(i)}_{\mathcal{C}}. \tag{12}$$

This concludes the proof that any deviation in the generation distribution ($P^{(i)}_{\mathcal{S}} \neq P^{(i)}_{\mathcal{C}}$), such as the tail truncation caused by LPV, directly results in the non-uniformity of U_i.

B Experimental Results on Llama3.2

In the main text, we primarily reported the results on Qwen2.5 to validate the effectiveness of our proposed RRNs-HT framework. To further demonstrate the universality and robustness of the Low-Probability Vanishing (LPV) vulnerability, we extended our evaluation to the Llama3.2 family of models.

This supplementary analysis verifies that the detected finite-precision artifacts are not specific to a single model architecture but are intrinsic to the floating-point arithmetic operations used in steganographic sampling. Table 4 presents the detailed quantitative results across three diverse datasets (Movie, Twitter, and News). The results reveal a consistent trend where the AC method on Llama3.2 exhibits significantly high Z-scores across all datasets, frequently exceeding the critical threshold of 1.645 ($\alpha = 0.05$) even at short sequence lengths. In stark contrast, the Normal sampling baseline maintains consistently low Z-scores, validating that our RRNs-HT method correctly identifies naturally generated text as benign with a low false-positive rate. Furthermore, we observe that the Z-scores for steganographic text tend to increase with the sequence length, aligning with our theoretical expectation that the cumulative evidence of the "missing tail" becomes more statistically pronounced in longer texts. These findings strongly support the conclusion that LPV is a fundamental implementation flaw in current "provably secure" steganography methods regardless of the underlying language model.

C Examples of Generated Text

To provide a more intuitive understanding of the LPV phenomenon, Table 5 illustrates the qualitative and statistical differences between text generated by standard random sampling (Normal) and the Arithmetic Coding (AC) steganography method.

Visually and semantically, both text samples appear highly fluent and coherent, rendering them virtually indistinguishable to human readers and

Prompt	I highly recommend Malcolm Gladwell's "The Tipping Point." My next audiobook will probably be one of his as well.	Z-score
Normal	I've read all of his non-fiction books but this is the first of his audiobook titles I've heard. So, I listened to excerpts to gauge the quality of narration. And it's first rate. It's even a secondary reason to listen to other project of Gladwell's (such as Outliers or Blink). If you have any interest in hearing Gladwell talk about concepts such as "The Law of the Few", "Outliers" or "The Stickiness Factor", then I highly recommend you give [...continues]	0.33
AC	Speaking strictly in terms of best-selling fiction, a deft reader can make an interesting, hard-to-classify, obtuse novel come alive. So, what does a reader do if he wants to make the words in a novel come alive? Which of Gladwell's tips from the book would be the most relevant to this endeavor? Certainly! To make the words in a novel come alive involves engaging all the senses and bringing the characters and setting to life for the listener. Malcolm Gladwell's ideas about [...continues]	1.67
Meteor	Thanks for sharing. This review expresses appreciation for Malcolm Gladwell's book "The Tipping Point" and expresses enthusiasm about reading more of his work in audio format. An implicit insight from the review is Gladwell's wide appeal either due to clever storytelling and his ability to explain complex ideas simply and clearly. The reviewer finds it useful as a reference and credit to another user for suggesting Gladwell's work. The recommendation given to others provides potential readers with an idea about a book's value, thus illustrating the [...continues]	0.99

Table 5: Examples of generated text. Green highlights indicate tokens with RRNs in the interval $[0.9, 1)$.

capable of fooling state-of-the-art semantic steganalyzers. However, the statistical side-channel analysis performed by RRNs-HT reveals a distinct structural anomaly. In the Normal sample, we observe a natural distribution of Representative Random Numbers (RRNs) across the entire $[0, 1)$ interval, including the high-probability tail region. Conversely, the AC-generated text systematically lacks tokens falling into this specific tail interval (e.g., $[0.9, 1)$) due to the finite-precision constraints discussed in Section 3. This structural "invisible tail" in the AC sample directly leads to the significantly elevated Z-score (e.g., 1.67 vs. 0.33), demonstrating how RRNs-HT bypasses the semantic layer to effectively detect steganography at the arithmetic level.