ONE SENTENCE, TWO MEANINGS: SEMANTIC MULTIPLEXING FOR LLM-GENERATED TEXT STEGANOGRAPHY

Anonymous authors
Paper under double-blind review

ABSTRACT

Semantic steganography typically hides information by generating LLM text via mapping secret bits to a single semantic space. A natural text, however, can readily convey two distinct meanings rooted in different domains of knowledge through a single carrier. In this work, we attempt to transform this polysemy of natural language into a measurable semantic steganographic channel resource, called semantic multiplexing. We introduce Iphis, a steganographic protocol that embeds two independent secret bit streams in one carrier. For knowledge bases such as urban traffic and computer networks, there are homographs like "traffic" and "path", called pivot words. Iphis introduces joint semantic coordinates to encode the semantics from two different spaces, and uses pivot words to assist LLM in generating carrier sentences. Text generated by Iphis has two layers of semantics: the surface semantics can be naturally understood, while the inner ones can only be extracted when corresponding knowledge base is revealed. Our evaluation indicates that the second semantic stream doubles the matched payload per carrier, initially demonstrating potential of semantic multiplexing. The evaluation also shows that a warden without key cannot recover hidden meaning.

1 INTRODUCTION

Large language models (LLMs) have profoundly changed text steganography. Semantic steganography starts to map secret bits to semantic space encodings and control LLMs to generate stegotext (Yang et al., 2020b; Bai et al., 2024; Figueira, 2022). Receivers recover secret messages by classification, usually considering only a single semantic space. Natural language, however, is inherently polysemous: one sentence can carry two meanings. Since the dawn of human history, countless puns, metaphors, irony, and oracles have drawn on the richness of this ambiguity, yet text steganography has not exploited such redundancy. In this paper, we attempt to transform the polysemy of natural language into a measurable semantic steganographic channel resource, called *semantic multiplexing*.

We take inspiration from an ancient hoax: a political leader, in an effort to defame his opponent, concocted an insinuation: "A container flies up to the sky, and the moon shines over Chang'an (today's Xi'an)". Since the name of his opponent happens to mean "container" and "moon", a seemingly innocuous nursery rhyme could send a disturbing implication. Here, "moon" played the role of a *pivot word* that refers to two different knowledge bases: one is the astronomical phenomenon, and the other is the name of the political figure. The rhyme conveyed dual meanings with no encoding: the inner meaning was hidden from the majority of the public. Such hiding, though, is opportunistic: it relies on the background of the audience rather than on any keyed, reproducible mechanism. That is what a steganographic protocol has to supply.

Earlier generative text steganography hid bits in per-token choices (Kaptchuk et al., 2021; Ding et al., 2023; Zhang et al., 2021), an idea inherited from classical substitution coding (Chang & Clark, 2014), or in one semantic state per text (Yang et al., 2020b; Bai et al., 2024); each places a single message in one carrier. Semantic multiplexing instead uses one carrier to hold a second independently keyed stream rather than to carry more bits in the first (Section 2).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Towards this goal, we propose a no-training text steganography protocol that embeds two readings, A and B, in one carrier text. The two readings are drawn from different semantic domains and are linked by a shared pivot word, a homographic word readable in both domains (e.g., *flow* as city traffic or as network data). Reading A is what any reader recovers from the sentence as written; reading B is not a subjective association a reader happens to form, but a second, domain-specific reading fixed by the candidate space, the reception rule, and the key hypothesis. Therefore, A and B can be independently decoded, carrying different secret messages. We name this protocol *Iphis* after the figure in Ovid's *Metamorphoses*, whose latent possibility surfaces only when elicited, just as B does. Throughout, we evaluate Iphis against a warden that holds the protocol, the channel, and the knowledge base, but not the key, and whose only lever is the observable distribution of carriers.

Our key contributions are:

- We propose *semantic multiplexing*: a single natural carrier encodes a joint semantic coordinate (A, B) whose two keyed streams are independently decodable, one under plain reading and the other only under its hypothesis and key.
- We instantiate semantic multiplexing as *Iphis*, a no-retraining protocol with a public semantic channel, keyed rejection sampling, and a pivot renderer, and measure a matched net payload of 1.0 bit per carrier against 0.5 for the single-stream baseline.
- We validate the protocol end to end on a real LLM, recovering all the frames, and run a per-stream key-scrambling intervention that isolates the capacity gain to the second semantic coordinate rather than a shared surface artifact.

2 Related Work

Token-level and semantic steganography. Token-level schemes sample in bits per word or per token. RNN-Stega (Yang et al., 2018) and VAE-Stega (Yang et al., 2020a) reweight the next-word distribution; Graph-Stega (Yang et al., 2020b) adds a knowledge graph to keep semantic relevance as the rate rises, though the coded unit stays the word (1.23–2.49 bpw); ADG (Zhang et al., 2021) groups tokens adaptively to 5.147–5.650 bits per word. Semantic steganography moves the coding unit from the word to a meaning: Bai et al. (Bai et al., 2024) map a message onto a state of an ontology-entity semantic space and report resistance to text rendering and word blocking; Figueira (2022) surveys the area. Neither approach treats one carrier as holding two independently keyed readings; Iphis addresses both readings of a carrier jointly (Section 3.1).

Security notions and threat models. Steganographic security asks whether an adversary who knows the channel but not the key can tell a carrier from an ordinary text. Prior work answers that question in two ways. The schemes above argue security by measurement: they train a steganalyzer and report the result, so the number moves with the detector — VAE-Stega reports 70.7–72.0% detection accuracy for itself and 81.2–81.3% for RNN-Stega under one detector (Yang et al., 2020a) — and they assume the adversary does not hold the model. Provably secure constructions instead assume the adversary holds the model. Following the definition of security as closeness of the stego and cover distributions (Cachin, 2004), Meteor requires the carrier distribution to remain the channel's and calls a private model "unreasonable" (Kaptchuk et al., 2021); Discop invokes Kerckhoffs's principle and constructs an exact distribution copy, reporting 3.48–5.76 bits per token with detection error at chance (Ding et al., 2023); minimum-entropy coupling attacks the same problem from the coupling side (de Witt et al., 2023). Iphis draws from a public channel and labels draws by key, so the warden may hold the channel, the knowledge base, and the extractors (Section 3).

Adjacent lines. Whole-text hiding re-renders a secret text as an equal-length cover by rank-guided sampling (Norelli & Bronstein, 2026); it codes no bit payload and reports neither a per-text rate nor a steganalysis, so it is a different object, not a stronger baseline. Robust steganography makes rewording part of the threat model (Perry et al., 2025), which we leave out of scope (Section 5). Rejection-sampling carrier coding over LLM outputs is our closest constructive relative (Huang et al., 2025), while frontier models show emerging but unreliable steganographic ability when merely asked (Zolkowski et al., 2026); Iphis codes explicitly and measures what comes out. Constrained decoding is the machinery our renderer leans on (Dekoninck et al., 2024; Yang & Klein,

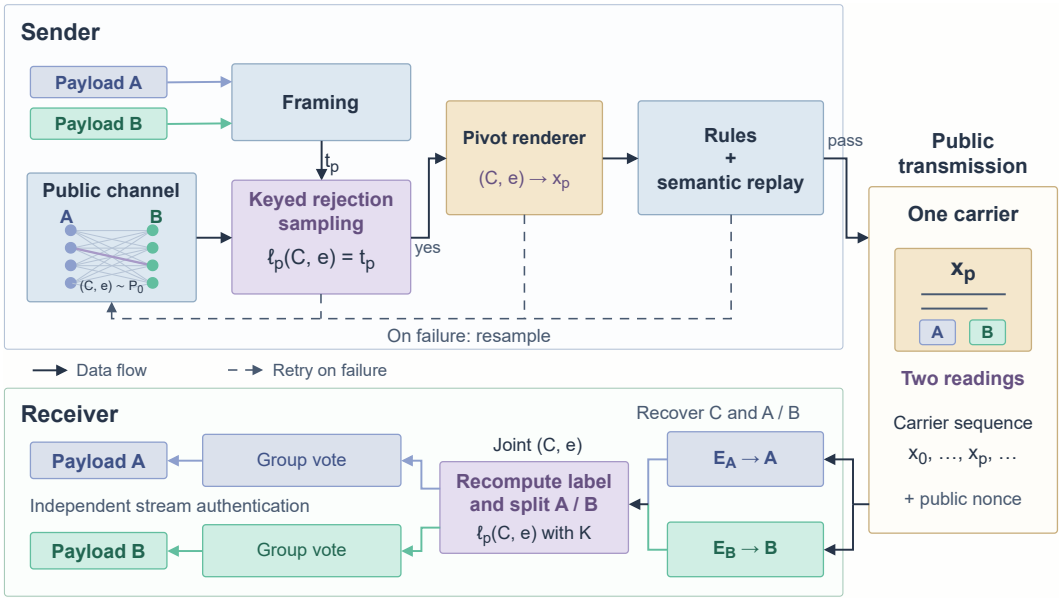

Figure 1: **The Iphis pipeline.** The sender (top) and the receiver (bottom) share one public channel; the carrier sequence crosses between them as public transmission. Symbols: context C; edge $e = (a, b)$ with readings $A = a$, $B = b$; state $z = (C, e)$; target t_p and keyed label $\ell_p(C, e)$ at position p; key K; draw budget $r_{\max}$.

Algorithm 1 The Iphis codec.

Require: key K, nonce, channel P_0, renderer R, extractors E_A, E_B, target $t_p \in \{0,1\}^q$

1: **Frame.** Build t_p from the length field, HMAC tag, keyed padding and r repetitions of each enabled stream.
2: **Draw.** Sample $z = (C, e) \sim P_0$; accept iff $\ell_p(C, e) = t_p$.
3: **Render.** Keep $x_p = \mathrm{R}(z)$ iff the rule checks pass and a replayed $\mathrm{Extract}(x_p) = z$; otherwise return to **Draw**, up to $r_{\max}$ attempts, then fail closed.
4: **Receive.** On $z_p = \mathrm{Extract}(x_p)$ (erasing on absent cues or competing candidates) recompute $\ell_p(z_p)$, vote within each group of r, unwhiten, and authenticate each stream separately.

2021; Liu et al., 2021; Yao et al., 2024): a pivot constrains one attribute of one sentence, while the protocol, not the decoder, chooses the state.

Puns and ambiguity. Pun and ambiguity research asks whether models can produce, recognize, or resolve a second reading (Xu et al., 2024; Chen et al., 2024; Liu et al., 2023); Iphis uses neither their data nor their methods. For them the second reading is an interpretation to be judged; for Iphis it is a state fixed by the candidate space and recovered under the key.

3 The Iphis Protocol

Iphis is one codec with three coupled stages. The public semantic channel defines legal states (Section 3.1); keyed rejection sampling assigns a state to each message symbol (Section 3.2); a pivot renderer realizes the state as text (Section 3.3). The receiver inverts the observation and verification steps on the same carrier sequence, and Figure 1 summarizes the pipeline.

A *frame* is one carrier sequence of N generated sentences, each a *physical position*. With repetition factor r, the N/r *logical positions* are each rendered r times and every copy is sampled independently, so the sender may try at most $r_{\max}$ draws per position; a stream is *enabled* when the current configuration carries its payload. Algorithm 1 states the sender's loop and the receiver's inversion at one position.

Drawing and rendering form a rejection loop: the key K decides which public draws are kept, the nonce travels with the sequence as public metadata, and a capacity failure may be retried under a keyed nonce variant. Appendix A.1 gives the frame layout and the per-stream authentication in full. We compare a single-stream configuration at $q = 1$ with the dual-stream configuration at $q = 2$, holding the public channel, renderer, and position budget fixed, so the extra bit addresses a second reading of the same sentence.

3.1 Semantic channel and joint coordinate

Given a public seed, nonce, and position, the channel returns a candidate draw $z = (C, e)$ with probability $P_0(C, e)$ and never inspects the payload or its target symbol. In this paper the channel is a hand-built semantic space over an urban and a computer-network domain. A bipartite graph has four clusters (*node*, *path*, *port*, *flow*); each contains four urban-side and four network-side units. Pairing one unit from each side gives an edge $e = (a, b)$, yielding 64 edges in total.

A draw names a public context C and a cross-domain edge $e = (a, b)$. Its two units are the carrier's two readings, $A = a$ on the urban side and $B = b$ on the network side. We call (A, B) the *joint coordinate* and (C, e) the *channel state*, writing the readings at position $p \in \{0, \dots, N-1\}$ as $A_p = a_p$ and $B_p = b_p$, and the state there as $z_p = (C_p, e_p)$. A state determines a coordinate, but not conversely: several public contexts may license the same pair of readings. The coordinate is thus a pair of readings, not the pivot word by itself.

Each cluster contributes three public contexts (twelve total), each fixing a genre and lead phrase, and fixes the pivot its renderer may use to connect the two readings. The renderer may not write explicit second-domain vocabulary. P_0 is uniform by default, with a Zipf-skewed variant changing only public edge weights. The graph is synthetic, and its seed and digest are fixed in the experimental manifest. We treat the protocol, the channel P_0, the semantic knowledge base, and the extraction rules as public, and keep the key private. The adversary therefore holds everything except the key, and her goal is to decide whether a carrier is engaged in covert communication at all. The security goal is that carriers stay indistinguishable from the public channel for a warden without the key. It is not that the second reading stays unreadable: a warden holding the knowledge base can read a coordinate, but cannot tell whether it was placed deliberately. A warden that additionally trains against this protocol is outside the present claim.

3.2 Keyed rejection sampling and framing

Framing fixes what each position must encode; the key then decides which public draws are kept, so the kept draw carries the target.

Framing. Each enabled stream carries a length field, an HMAC tag, HMAC-counter whitening, and keyed padding to the logical-position budget, and repetition factor r expands its logical symbols into physical targets. A stream whose repeated frame would exceed N is rejected as a capacity failure before any carrier is published. Framing runs before any position is sampled; at decoding the labels are recomputed, voted, unwhitened, and checked against the padding and the tag. A missing carrier or a group erasure invalidates the whole frame. Public draws at each position have budget $r_{\max}$, and exhaustion discards the current frame attempt. The tag is an HMAC, the same keyed primitive that supplies the position label at sampling.

Sampling and labelling. Let t_p be the q-bit target symbol obtained from the framed, padded, and repeated streams at physical position p. Here $q = q_A + q_B$, where q_A and q_B are the label widths contributed by the two streams. For candidate $z = (C, e)$, the encoder computes

$$\ell_p(C, e) = \mathrm{MSB}_q(\mathrm{HMAC}\,(K, \mathrm{enc}(\mathrm{nonce}, p, C, e)))\,,$$

where MSB_q denotes the q most significant bits of the digest. The function enc domain-separates the nonce, the position p, the context identifier C, and the edge identifier e. The encoder draws (C, e) from P_0 first and evaluates this label only afterwards: a match to t_p accepts the candidate, otherwise the next public draw is tried. The key therefore labels an already sampled channel rather than constructing static accepted and rejected buckets. With $q_A = q_B = 1$, the high and low bits label A and B; with $q_B = 0$, the same procedure is the matched single-stream baseline. The

renderer sees only accepted coordinates: rejected draws are never published and never enter the receiver input.

3.3 PIVOT RENDERING, RECEIVERS, AND REPETITION CODING

An accepted state is still an abstract pair of readings; this stage realizes it as a sentence, and reads the same pair back out of that sentence.

Rendering. Given $z_p = (C_p, e_p)$, a frozen LLM receives the context lead, semantic units, pivot recipe, and distractors, and generates one natural sentence x_p. The sentence must open with the lead, express A_p on its plain reading, and use the pivot to support B_p through designated B cues. The prompt forbids explicit second-domain vocabulary, and rule checks enforce context, length, and vocabulary constraints.

Replay. The sender replays the receivers on x_p and keeps it only when the recovered context, A reading, and B reading match (C_p, A_p, B_p). A failed generation or replay returns to rejection, so a PRF match alone does not publish a carrier.

Observation. The observation step uses cluster-scoped candidate tables. E_A extracts the urban unit from the plain reading, while E_B, invoked under the B hypothesis, extracts the network unit from the pivot and its cues. Either receiver emits an erasure for absent cues, competing candidates, or an unknown context. Two agreeing votes settle a decision, a third breaks a tie, and three different answers erase it; the same rules govern replay and decoding (Appendix A.1).

Decoding. Repetition sits above this per-carrier step: each logical bit or label is repeated r times, and every physical copy is sampled and rendered independently, so group majority corrects isolated generation or extraction errors. Systematic confusions remain, and a tie is an erasure. The main experiments use $r = 3$. Each enabled stream is then unwhitened and authenticated independently, yielding zero, one, or two authenticated payload streams. A group erasure fails the frame, whereas a failed padding or tag check suppresses only its own stream and a successfully authenticated companion stream is still returned.

4 EXPERIMENTS

4.1 SETUP

Channel. The channel is a synthetic knowledge base of 32 units, 64 cross-domain edges, and 12 public contexts (four clusters with four units per domain). It is built for protocol validation and is not a claim about natural-language meaning.

Protocol matrix. The synthetic matrix runs 5 independent key and nonce seeds with 1000 frames per configuration, 256 fixed physical positions, retry budget $r_{\max} = 96$, symbol widths $q \in \{1, 2\}$, and both the uniform and the Zipf-skewed stress channel; a separate finite-alphabet boundary uses 5 seeds $\times$ 200 frames per configuration at $q \in \{3, 4\}$.

Closed loop. The real-LLM closed loop fixes a single model (`deepseek-chat`, reasoning off), generation temperature 0.3, receiver temperature 0.0, three-vote arbitration, one-byte payload per stream, repetition factor $r = 3$, and 10 requested frames; prompt v2 forbids a fixed set of template connectives and forces syntactic diversity across carriers, and is identified by the frozen configuration and the knowledge-base seed digest recorded in the run manifest. The two rendering recipes, coordinate and pivot, are compared against the single-stream baseline and against a matched generation pipeline that shares the generator, frozen prompts, and replay-verification gate but drops the keyed label test.

Metrics. We report correctness (frame recovery, erasure, authentication), capacity, distribution drift, and generation cost. R_{nominal} counts every label bit the channel can carry, R_{protocol} the payload a frame contracts for, and R_{net} that payload weighted by the observed frame success; TV

Table 1: Protocol capacity and correctness on the synthetic channel (L=128 bits per enabled stream; TV over 1.28M carriers).

Metric	Single-stream	Dual-stream
Frame success	$5000/5000$	$5000/5000$
R_{nominal} (bit/text)	1.0	2.0
R_{protocol} (bit/text)	0.5	1.0
R_{net} (bit/text)	0.5	1.0
Acceptance rate	0.498	0.246
Mean attempts / position	2.01	4.07
Attempts p95 / p99 / max	5 / 7 / 25	11 / 17 / 54
Marginal edge TV	0.0025	0.0036
Joint $(\text{context}, e)$ TV	0.0045	0.0066

is total variation from the public channel. S_A and S_B are the fractions of encoded frames whose A and B payloads are recovered in full; joint success requires both. Detection metrics are defined in Section 4.5. Ablations toggle one factor at a time for the sampling knobs (single versus dual stream, $r = 1$ versus $r = 3$) and for the prompt (v1 versus v2, Appendix A.4); the two rendering recipes are compared as run.

4.2 Protocol capacity and correctness

Table 1 reports the synthetic-channel matrix at 128 payload bits. At $L = 128$, both configurations round-trip all 5000 frames with zero failures, zero erasure, and full authentication, and the dual-stream configuration raises the matched net rate from 0.5 to 1.0 bit per text. Here $\Delta R_B = R_{\text{net}}(\text{dual}) - R_{\text{net}}(\text{single}) = 0.5$, identical over all five seeds and both channels, which reflects the deterministic protocol matrix; the zero-failure result gives a one-sided rule-of-three upper bound of 6×10^{-4} on the per-configuration failure rate. The same linear rate relation $R_{\text{net}}(L) = L/128$ was observed across the additional payload lengths and stress configurations listed in Appendix A.7.

Wider symbols require more proposals. Acceptance follows $\Pr[\text{accept}] \approx 2^{-q}$, so the dual-stream $q = 2$ configuration accepts 0.246 of draws and spends a mean of 4.07 attempts per position against 0.498 and 2.01 for the single-stream baseline. Increasing q exposes the sampling-cost boundary: retry exhaustion is 0.9–1.8% at $q = 3$ and 85.9–94.8% at $q = 4$ (per configuration of 1000 frames), while all target symbols remain reachable in the audited positions. The Zipf-skewed stress channel leaves the rate relation unchanged, affecting only public edge weights. Rejection sampling keeps the induced drift small: the empirical joint $(\text{context}, e)$ total variation over the finite 1.28M-carrier matrix is 0.0045 (single) and 0.0066 (dual), and the corresponding residual correlations satisfy $|r| \leq 0.006$. This is a codec-level figure: the protocol matrix's deterministic renderer never fails, so the accepted distribution equals the public channel exactly. Under the LLM renderer a candidate whose rendering or replay check fails is redrawn, which reweights the distribution, so the end-to-end drift exceeds this figure (Section 4.5).

4.3 Real-LLM closed loop and error correction

Figure 2 shows one published carrier and the two readings it holds. Table 2 reports the closed loop on the frozen `deepseek-chat` model for both rendering recipes. The coordinate recipe gates every carrier with a sender-side rule self-check plus replay verification. It encodes $9/10$ requested frames and fully recovers all 9 encoded frames ($720/720$ position-level semantic matches; end-to-end $9/10$); the tenth frame fails closed at position 47 after exhausting $r_{\max} = 48$ attempts.

The pivot recipe uses LLM replay arbitration in place of the sender-side rule self-check. Its plain form recovers $7/9$ encoded frames; the errors concentrate in single-position *systematic confusions*, where the receiver confidently maps a text about one B unit to an adjacent B unit on every vote without erasing, and generation-side B omission, which no extractor can recover. Position-level repetition absorbs the observed systematic-confusion errors: the full pivot matrix contains $2394/2400$ matching positions, and all six mismatches are corrected by their repetition groups.

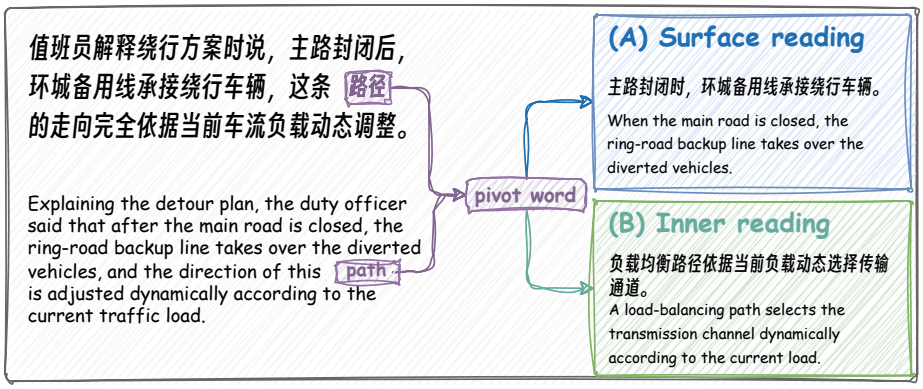

Figure 2: **An instance of stegotext.** A published carrier rendered under the *path* pivot.

Table 2: Real-LLM closed loop on `deepseek-chat` (5 seeds $\times$ 2 frames, 240 physical positions $=$ 80 logical $\times 3$, retry budget $r_{\max} = 48$; pivot column uses prompt v2).

Metric	**Coordinate**	**Pivot ($r = 3$, v2)**
Frames encoded / requested	$9/10$	$\mathbf{10/10}$
Frames recovered / encoded	$9/9$	$\mathbf{10/10}$
End-to-end success / requested	$9/10$	$\mathbf{10/10}$
Position-level matches	$720/720$	$2394/2400$ (6 absorbed)
Reading success S_A / S_B / joint	1.0 / 1.0 / 1.0	1.0 / 1.0 / 1.0
Erasure rate	0	0
Amortized run calls / encoded frame	≈ 1139	≈ 4409
Cover renders accepted	$28/50$	$20/20$

Removing the code costs half the frames: at $r = 1$, with the same 80 logical positions, the identical pipeline encodes all 10 frames but recovers only 5 ($S_A = 0.5$, $S_B = 0.8$), since one uncorrected position error fails a stream. The per-position match rate barely moves (0.9938 vs 0.9975), so repetition buys error tolerance rather than cleaner positions, at four times the calls (44 090 vs 10 766). A matched control that changes only the prompt also reaches $10/10$, at fewer calls (32 496 vs 44 090); its effect appears on the detection side, where four of nine detectors exceed the pre-registered pass line on the v1 carriers (Appendix A.4). The reported configuration shows zero erasures and joint success 1.0. Ten frames carry little precision: under the one-sided convention used above, zero failures in ten bounds the frame failure rate by $1 - 0.05^{1/10} = 0.259$, so the closed loop supports a success rate of at least 74%.

The final nonce travels with the frame as public metadata, and in a sampled cross-process check the decoder recovers $2/2$ frames, including one nonce-variant frame. The bottleneck of the closed loop is therefore receiver consistency, not label verification, which succeeds on every accepted carrier.

The two recipes accept natural covers at different rates: the coordinate recipe accepts $28/50$ under its two-fact rule self-check, the pivot recipe $20/20$ under LLM replay arbitration.

Cross-model re-decoding provides a modularity check for the two coordinates. Under the same decoder, `Qwen3-30B-A3B` recovers all eight bits of the B stream in both sampled frames, while `DeepSeek-V3.2` recovers them in one of two frames. Transfer of the B stream across receivers supports its interpretation as a keyed protocol coordinate, whereas A remains tied to the surface realization produced by the renderer.

The pivot run consumed 44 090 provider calls (29 656 encode, 14 414 decode, 4407 per frame, plus 20 in-run cover renders) and the coordinate run 10 248 (5522 encode, 4321 decode, 355 on the failed tenth frame, plus 50 cover renders). Against the $240 \times 2 = 480$ label bits the frame nominally carries, it delivers 16.

4.4 DOES THE GAIN COME FROM THE SECOND STREAM?

A scramble decodes each side with an independent key. Scrambling one key destroys only its associated coordinate: under B scrambling, B has bit error rate 0.502 and fails authentication on all frames, while A is preserved bit-for-bit; A scrambling produces the symmetric result ($\mathrm{BER} = 0.490$). Together with the paired single-versus-dual matrix, these interventions support attribution of the capacity gain to an independently keyed second coordinate: scrambling only the B side leaves every A reading and the carrier surface untouched while destroying exactly the B payload. The frame authenticates the recovered payload streams.

Tamper collision. In a targeted integrity probe, replacing a carrier by a different same-context edge with the same PRF label succeeds on $11775/11775$ trials (Wilson CI lower bound 0.99967), separating payload authentication from carrier binding.

4.5 DETECTION AND NATURALNESS

We audit statistical separation and naturalness separately. Table 3(a) contrasts the 2400 published pivot carriers (prompt v2) against 500 covers from a matched generation pipeline, which shares the generator, frozen prompts, and replay-verification gate, removes only the keyed label test, and is not edge/context composition-matched. This cover set is itself replay-verified (500 accepted from 921 attempts) and serves both the detection and the naturalness contrasts below. Because a steganalyzer may exploit a score in either direction, we report the raw AUROC, its direction-aware form $\mathrm{Dir.} = \max(\mathrm{AUROC}, 1 - \mathrm{AUROC})$, and a one-sided 95% upper bound (Upper) from a cluster bootstrap over the 10 frames; the pre-registered pass line is ≤ 0.55. Five of nine detectors pass; the four that exceed the line do so in the reversed direction and track surface composition, since each is removed by deduplication, nearest-length matching, or context matching (Appendix A.3), and the carriers are shorter (47.2 vs 48.4, $p = 0.001$) and more repetitive (38%) than the matched covers, so the table characterizes this renderer's operating distribution.

A protocol-aware warden. These nine statistics are surface features, not the natural detector for a keyed coordinate. An adversary who knows the protocol, the channel, and the knowledge base can run the public extractors and read off a text's coordinate (C, e), so her statistic is the coordinate distribution itself, and the construction predicts that it is uninformative (Appendix A.5). Over the 192 legal coordinates and the same corpora as Table 3(a), a cross-fitted coordinate-only classifier is at chance (AUROC 0.511), with no significant cluster contrast (total variation 0.045, permutation $p = 0.21$). The detectors that do exceed the line therefore localize to the renderer, not to the key.

Naturalness (Table 3(b)) is broadly consistent with cover. Per-text lexical diversity and within-text repetition are close between stego and cover, with no significant difference; Self-BLEU is slightly higher for stego (0.808 vs 0.790), consistent with the duplication residual above; the blinded `GLM-4-9B-0414` judge reports similar mean scores across all three dimensions. The judge is smaller than the generator, so we read its scores as an instrument applied identically to both groups rather than as calibrated ground truth. The perplexity difference between the full samples (192.7 vs 181.5, $p = 0.042$) does not survive Holm correction (smallest adjusted $p = 0.167$) and reverses under nearest-length matching (192.7 vs 229.8), so surface length is the relevant covariate for this comparison.

4.6 POSITIONING AGAINST PRIOR STEGANOGRAPHY

On capacity we are far below this literature. A token-level method embedding one bit per word carries 50 to 250 times more per sentence than the 1.0 bit our dual configuration delivers, and our ceiling is set by the label width q rather than by tuning, since retry exhaustion already dominates at $q = 4$ (Section 4.2). What none of these methods offers is a second, independently keyed stream in the same carrier: each places one message per carrier. That is why we measure the gain at a matched position budget and attribute it by key scrambling. The six systems we position against report capacity in units that cannot be placed beside ours, and none of them reports a numeric extraction accuracy; their as-reported figures, and why they are not tabulated here, are in Appendix A.6.

Table 3: Detection and naturalness of published pivot carriers (prompt v2) against matched generation cover (2400 stego vs 500 cover; judge `GLM-4-9B-0414` on 100 vs 100; PPL under Qwen2.5-1.5B-Instruct).

(a) Detection.

Detector	AUROC	Dir.	Upper
char_entropy	0.453	0.547	0.569
rep4	0.500	0.500	0.500
punct_digit	0.496	0.504	0.528
sent_len_mean	0.454	0.546	0.570
sent_len_std	0.500	0.500	0.504
n_sentences	0.500	0.500	0.504
top5	0.527	0.527	0.548
len_chars	0.453	0.547	0.570
bigram_xent (cross-fitted)	0.447	0.554	0.577

(b) Naturalness.

Naturalness metric	Stego	Cover	MWU p
Distinct-1 (per text)	0.874	0.874	—
Distinct-2 (per text)	0.984	0.984	—
Distinct-3 (per text)	0.9997	0.9995	—
Self-BLEU	0.808	0.790	—
rep-2	0.033	0.032	—
GLM fluency (1–5)	4.54	4.44	0.16
GLM naturalness (1–5)	3.97	3.90	0.44
GLM coherence (1–5)	4.59	4.51	0.26
PPL (unadjusted)	192.7	181.5	0.042

Cluster-bootstrap intervals: Self-BLEU $[0.004, 0.033]$; every other row contains zero.

5 Discussion, Limitations, and Conclusion

5.1 Operating Point and Design Rule

Iphis establishes a practical operating point for semantic multiplexing. On the fixed 64-edge channel, $q = 2$ doubles the matched net rate from 0.5 to 1.0 bit per text while preserving the public channel and physical position budget. The closed loop reaches $10/10$ frames once sender replay verification, cue-aware erasure, three-vote arbitration, repetition coding, and nonce-variant retry are stacked. At wider symbols, retry exhaustion appears at $q = 3$ and dominates at $q = 4$, identifying a sampling boundary rather than a semantic-capacity gap. The result is a clear design rule: add a semantic coordinate when its readings remain separable under a shared hypothesis and when the renderer can expose receiver-verifiable cues.

5.2 Broader discussion

Capacity per carrier is governed by three coupled quantities: the number of legal joint coordinates, agreement among the receivers that observe them, and the cost of rendering an accepted coordinate. We vary payload length L and label width q and hold the rest at the operating point, where the binding constraint is the acceptance probability 2^{-q} rather than the size of the semantic alphabet (Section 4.2).

5.3 Limitations and future work

The reported rates belong to a synthetic graph with hand-annotated pivots, rendered by one model in one language; rebuilding the channel on an external semantic resource, automating pivot curation, and replacing rejection with a cheaper renderer would ease those bounds. Two limits remain open, and extending the protocol would have to address both: we do not test a warden adapted to this protocol, and payload authentication holds independently of the carrier, since repetition repairs isolated label errors and not paraphrase.

5.4 Conclusion

Iphis embeds two independently addressed semantic streams in one natural sentence: the surface reading is available to any reader, and the inner reading appears only under its hypothesis and key. Addressing both readings jointly doubles the matched payload per carrier at a fixed position budget, so polysemy becomes an addressable resource rather than noise to be controlled. The pattern extends to other semantic contrasts that a shared pivot can hold apart.

REPRODUCIBILITY STATEMENT

All experiments in this work were run under frozen pipeline configurations, including public channel seed, PRF domain separation, prompt version, model and sampling, and extraction voting rule. Every numeric result can be recomputed from the accompanying report manifests. A minimal code release is available at `https://anonymous.4open.science/r/iphis_iclr2027-CC7E/`: it contains the protocol implementation, the frozen synthetic knowledge base, and the evaluation scripts, without corpora. The full release (source code, pipeline configuration, corpora, and instructions for regenerating carriers and replaying the closed-loop evaluation) will accompany publication. The codebase only depends on the Python standard library and is covered by unit tests. Formal statements and underlying assumptions are deferred to the appendix.

ETHICS STATEMENT

The semantic knowledge bases used in this work are synthetic corpora constructed by the authors, with Chinese homographic-pivot units built from benign public-domain facts. This study involves no human subjects or personal data. The self-built corpora contain no material from third-party or proprietary sources and can be shared without conflicting with any third-party licensing terms. We plan to release the seed data and experiment code for research reuse, and we welcome further public construction of open-domain pivot semantic knowledge bases.

AI USE STATEMENT

We used generative AI tools to source information, generate the synthetic knowledge bases, assist with code implementation, give feedback on experiments, and polish the manuscript. We did not use them to develop the conceptual framework, write proofs, clean or reformat data, analyse results, or interpret them. We have reviewed all AI-assisted work and take responsibility for the final content, including text, claims, and artifacts produced with its aid.

REFERENCES

Minhao Bai, Jinshuai Yang, Kaiyi Pang, Yongfeng Huang, and Yue Gao. Semantic steganography: A framework for robust and high-capacity information hiding using large language models. *arXiv preprint arXiv:2412.11043*, 2024.

Christian Cachin. An information-theoretic model for steganography. *Information and Computation*, 192(1):41–56, 2004.

Ching-Yun Chang and Stephen Clark. Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method. *Computational Linguistics*, 40(2):403–448, 2014. doi: 10.1162/COLI_a_00176.

Yang Chen, Chong Yang, Tu Hu, Xinhao Chen, Man Lan, Li Cai, Xinlin Zhuang, Xuan Lin, Xin Lu, and Aimin Zhou. Are U a joke master? pun generation via multi-stage curriculum learning towards a humor LLM. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 878–890, Bangkok, Thailand and virtual meeting, aug 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.51. URL `https://aclanthology.org/2024.findings-acl.51`.

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

Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin Vechev. Controlled text generation via language model arithmetic. In *International Conference on Learning Representations (ICLR)*, 2024.

Jinyang Ding, Kejiang Chen, Yaofei Wang, Na Zhao, Weiming Zhang, and Nenghai Yu. Discop: Provably secure steganography in practice based on distribution copies. In *2023 IEEE Symposium on Security and Privacy (S&P)*, 2023.

João Figueira. A survey on semantic steganography systems, 2022.

Yu-Shin Huang, Peter Just, Hanyun Yin, Krishna Narayanan, Ruihong Huang, and Chao Tian. OD-Stega: LLM-based relatively secure steganography via optimized distributions, 2025. Closest constructive baseline; title, author list, and identifier follow arXiv:2410.04328v2.

Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. Dexperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Alisa Liu, Zhaofeng Wu, Julian Michael, Alane Suhr, Peter West, Alexander Koller, Swabha Swayamdipta, Noah Smith, and Yejin Choi. We're afraid language models aren't modeling ambiguity. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 790–807, Singapore, dec 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.51. URL https://aclanthology.org/2023.emnlp-main.51/.

Antonio Norelli and Michael M. Bronstein. Llms can hide text in other text of the same length. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=VbTLgEUocp. Poster; Calgacus, same-length generative text hiding.

Neil Perry, Sanket Gupte, Nishant Pitta, and Lior Rotem. Robust steganography from large language models. *arXiv preprint arXiv:2504.08977*, 2025.

Zhijun Xu, Siyu Yuan, Lingjie Chen, and Deqing Yang. "a good pun is its own reword": Can large language models understand puns? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11766–11782, Miami, Florida, USA, nov 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.657. URL https://aclanthology.org/2024.emnlp-main.657/.

Kevin Yang and Dan Klein. Fudge: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.

Zhong-Liang Yang, Xiao-Qing Guo, Zi-Ming Chen, Yong-Feng Huang, and Yu-Jin Zhang. Rnn-stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295, 2018.

Zhong-Liang Yang, Si-Yu Zhang, Yu-Ting Hu, Zhi-Wen Hu, and Yong-Feng Huang. Vae-stega: Linguistic steganography based on variational auto-encoder. *IEEE Transactions on Information Forensics and Security*, 16:880–895, 2020a.

Zhongliang Yang, Baitao Gong, Yamin Li, Jinshuai Yang, Zhiwen Hu, and Yongfeng Huang. Graph-stega: Semantic controllable steganographic text generation guided by knowledge graph, 2020b.

Shunyu Yao, Howard Chen, Austin W. Yu, Yingbo Zhou, and Karthik Narasimhan. Collie: Systematic construction of constrained text generation tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

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

Artur Zolkowski, Kei Nishimura-Gasparian, Robert McCarthy, Roland S. Zimmermann, and David Lindner. Early signs of steganographic capabilities in frontier llms. In *International Conference on Learning Representations (ICLR)*, 2026. Poster.

A Appendix

A.1 Complete algorithm

We give the protocol-level pseudocode; the end-to-end flow is illustrated in Figure 3. The deterministic "stand-in" renderer is used only for protocol matrices; all reported LLM numbers use the pivot renderer of Appendix A.2.

Algorithm 2 Protocol-level Iphis sender: framing and the rejection loop.

Require: $q = q_A + q_B$ with $q_A, q_B \in \{0, 1\}$; N physical positions with $r \mid N$; shared key K; label $\ell_p(C, e) = \mathrm{MSB}_q(\mathrm{HMAC}(K, \mathrm{enc}(\mathrm{nonce}, p, C, e)))$

for each permitted nonce variant, initial nonce first **do**
  for each enabled stream $S \in \{A, B\}$ **do**
    $f_S \leftarrow [\mathrm{version} \mid \mathrm{length} \mid \mathrm{payload}_S \mid \mathrm{HMAC\ tag}]$
    if $r|f_S| > N$ **then**
      return CAPACITY_FAILURE
    end if
    $w_S \leftarrow \mathrm{whiten}(f_S, K, \mathrm{nonce}, S)$
    $t_S \leftarrow \mathrm{repeat}(\mathrm{keyed_pad}(w_S, N/r, K, \mathrm{nonce}, S), r)$
  end for
  $X \leftarrow [\,], p \leftarrow 0$
  while $p < N$ **do**
    $t_p \leftarrow (t_A[p], t_B[p])$ over the enabled sides; $x \leftarrow$ NONE
    for $a = 1$ **to** $r_{\max}$ **do**
      $(C, e) \sim P_0$ ▷ draw first, label afterwards
      if $\ell_p(C, e) = t_p$ **and** $\mathrm{R}(C, e)$ succeeds **and** $\mathrm{Extract}(\mathrm{R}(C, e)) = (C, e)$ **then**
        $x \leftarrow \mathrm{R}(C, e)$; **break** ▷ sender replay
      end if
    end for
    if $x =$ NONE **then**
      break ▷ discard X; next nonce restarts framing
    end if
    $X \leftarrow X \| x; p \leftarrow p + 1$
  end while
  if $p = N$ **then**
    return (X, nonce) ▷ complete frame only
  end if
end for
return CAPACITY_FAILURE ▷ no partial publication

A.2 Frozen configuration and prompts

Frozen configuration. Knowledge base: 4 clusters × (4 A + 4 B) units, 64 edges, 12 contexts; seed digest `073f77edba791f272dcf836bbe9d9cfca637309acdcf3740ca2bae4f7f38e432`. Protocol matrix: 5 key/nonce seeds, 1000 frames/config, $N = 256$, $r_{\max} = 96$, $q \in \{1, 2\}$ for the full matrix (plus a $q \in \{1..4\}$ label-support audit), payload lengths $\{0, 1, 5, 7, 8, 16, 32, 64, 128\}$; stress channel Zipf-skewed weights (validated: coverage, finiteness, positivity, min normalized probability). LLM loop: `deepseek-chat` (reasoning off), generation temperature 0.3 / top-p 0.95 / 512 output tokens, receiver temperature 0.0, three votes (agree / tiebreak / split-eraser), 240 physical positions (80 logical × $r = 3$), 1-byte payload/stream, 4-byte tag, repetition factor $r = 3$. Nonce uniqueness is caller-guaranteed. The final nonce after a nonce-variant retry is recorded as public frame metadata; a separate decoder process re-derives it and re-decodes the published carriers ($2/2$ frames verified, including one nonce-variant frame).

Generation prompt (coordinate recipe, English gloss). "You are a semantic carrier generator. Given a public context, a surface fact A and an inner fact B, produce one natural, fluent Chinese sentence such that: (1) it opens with the given context lead; (2) under a plain first reading it clearly

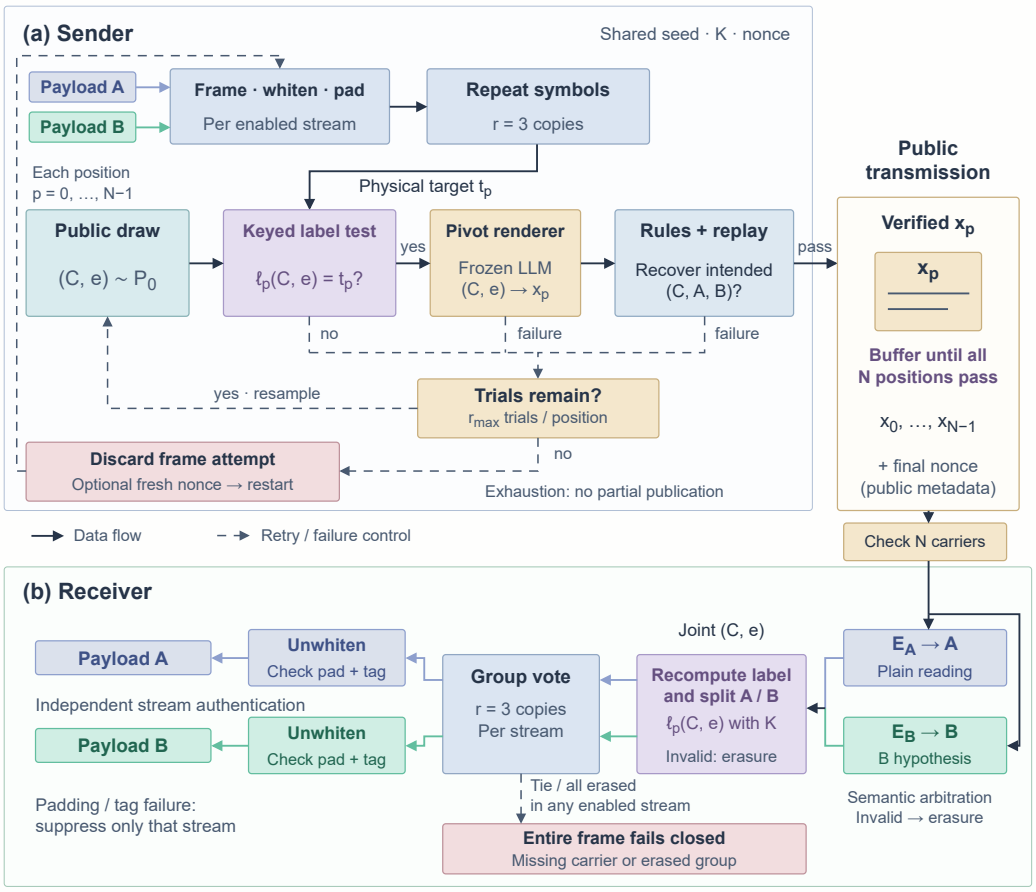

Figure 3: **End-to-end Iphis flow and failure handling.** **(a)** Each draw is labelled before rendering and kept only if replaying the receivers recovers the same context–edge pair; either gate failing returns the position to sampling, and exhausting the per-position $r_{\max}$ budget discards the attempt, so no partial frame is published. **(b)** At reception the same observation procedure runs and the encoding is inverted; an erased or tied group fails the frame, whereas a padding or tag check suppresses only its own stream.

Algorithm 3 Protocol-level Iphis: extraction and reception.

function EXTRACT(x) ▷ same observation procedure for replay and reception
  identify the cluster; arbitrate $E_A(x)$ against $E_B(x \mid B \text{ hypothesis})$
  recover the context C and the edge $e = (A, B)$ from the public tables
  return (C, e), or ERASURE if either reading or the context is invalid
end function

function RECEIVER(X, nonce)
  if $|X| \neq N$ **then**
    return FRAME_FAILURE
  end if
  for $p = 0$ **to** $N - 1$ **do**
    $z \leftarrow \mathrm{Extract}(X[p])$; on ERASURE record an erasure for both enabled sides, else recompute $\ell_p(z)$ and split it into A/B label bits
  end for
  for each enabled stream S **do**
    for each consecutive group of r physical positions **do**
      vote over the non-erased label bits; **if** tied or all erased **return** FRAME_FAILURE
      append the majority bit to w_S
    end for
    unwhiten w_S; parse the frame and check its HMAC tag and keyed padding
    release $\mathrm{payload}_S$ iff all checks pass, otherwise suppress S
  end for
end function

expresses fact A; (3) the same sentence also naturally refers to fact B (coordinated addition, elaboration, or follow-up), so that both facts are independently identifiable; B must be worded with B's own distinctive cue vocabulary (words absent from all other candidate B facts in the distractor list); (4) do not paste or enumerate fact sources verbatim; paraphrase naturally while preserving key semantics; (5) total length within $[\mathrm{min_chars}, \mathrm{max_chars}]$ (check with `len(text)`); (6) no forbidden terms. Output only JSON: {`"text": "..."`}."

Generation prompt (pivot recipe v2, English gloss). Same preamble with a shared pivot word: "(1) open with the context lead; (2) the pivot appears as the shared fulcrum of the A-reading and the B-reading, ideally once; (3) first reading: the pivot is read in the A-domain and the sentence naturally, unambiguously expresses fact A; (4) inner reading: re-reading the same pivot in the B-domain context implies fact B; B must be worded with B-domain distinctive cues (absent from distractors); (5) no forbidden terms (they would make the B-reading explicit); no verbatim pasting of fact sources; key semantics preserved; (6) total length within bounds; (7) phrasing must vary: do not reuse template skeletons such as "..., whereas ... then...", "..., just as .....", "both ... and ..."; vary word order, clause structure, and lexicon each time so the sentence reads like a natural sentence written by different people. Output only JSON: {`"text": "..."`}."

Receiver prompts (English gloss). E_A (plain reading): select the *unique* A fact from the candidate list that the text most directly expresses; consider only listed `semantic_ids`; if no candidate clearly matches or two are equally strong, output `null`. E_B (guided, coordinate): select the unique B fact implied by the same sentence; prefer cue vocabulary; *mandatory erasure* when no candidate's cues occur or cues split across candidates; JSON with `null` for erasure. E_B (pivot): the pivot word alone is *not* evidence of B (it is the shared fulcrum); without a non-pivot B-domain clue, or with clues pointing to several candidates, output `null`.

Naturalness judge prompt. Blinded LLM judge rates each text on three dimensions (1–5 integer each; no source/domain mention in the prompt). The judge is the open-weight `GLM-4-9B-0414`, served locally so the audit can be reproduced from released weights. The judge prompt is blinded to source domain and steganographic condition, and the exact version is archived with the experiment manifest.

A.3 DETECTOR DEFINITIONS AND PRE-REGISTERED AUDIT

Detector set (9 statistics). Let x be a carrier string and $n = |x|$ its character length. (1) *character entropy* $H_1 = -\sum_c \frac{\text{count}(c)}{n} \log_2 \frac{\text{count}(c)}{n}$; (2) *bigram cross-entropy* under an add-α (α=0.1, smoothing 5000) character-bigram model trained by 2-fold *symmetric* cross-fit (stego and cover are both scored under the opposite fold's model; the asymmetric self-fit cover variant inflates AUROC and is *not* used; folds are split by file order over the 500 covers with comparable per-fold edge composition); (3) *repetition ratio* rep4: fraction of character 4-grams that occur more than once within x; (4) *punctuation/digit ratio*: fraction of non-CJK, non-letter characters; (5–6) *sentence-length mean and standard deviation* over the segments split by sentence-ending punctuation (Chinese full stop / exclamation / question / semicolon, and their ASCII counterparts); (7) *top*-5 *concentration*: fraction of x covered by its five most frequent characters; (8) *length* n; (9) *sentence count*: number of sentence separators.

Direction-aware convention. A steganalyzer may exploit a score in either direction, so every detector is reported as $\max(\text{AUROC}, 1 - \text{AUROC})$ with AUROC under average-rank tie handling. Uncertainty is a one-sided 95% upper bound from 2000 resamples under a per-text bootstrap and a cluster bootstrap that resamples the 10 published frames (2400 stego texts) and the 500 matched covers.

Pre-registered criterion and ds3 result. The preregistered matched-contrast gate is a direction-aware upper bound of $\leq$ 0.55. Five of nine detectors pass (rep4, punct_digit, sent_len_std, n_sentences, and top5 at upper 0.548); four exceed the line in the reversed direction: char_entropy (0.569), sent_len_mean (0.570), len_chars (0.570), and the cross-fitted bigram (0.577), all of the length/entropy family.

Sensitivity of the four over-threshold detectors. Table 4 shows that each over-threshold detector is removed by an intervention orthogonal to the key. Deduplicating both groups (unique/unique) drops the bigram to 0.506 (upper 0.542), because the 38% duplicate rate reflects the small synthetic knowledge base and the low sampling temperature required for reliable recovery. Nearest-length matching of covers drops sentence length and character count to 0.50 (upper 0.517); context-stratified matching drops character entropy to 0.532 (upper 0.545). The matched conditions use a per-text bootstrap because the matching breaks the frame structure; full and dedup use the cluster bootstrap.

Table 4: Sensitivity of the four over-threshold detectors to deduplication, nearest-length matching, and context-stratified matching (direction-aware AUROC; bold marks the intervention that brings the detector inside the 0.55 line; upper bounds are given in the text).

Detector	Full	Dedup	Length	Context
bigram_xent (cross-fitted)	0.554	**0.506**	—	0.551
sent_len_mean	0.546	0.531	**0.501**	0.531
len_chars	0.547	0.531	**0.500**	0.533
char_entropy	0.547	**0.516**	0.546	0.532

Scramble and collision audits. Scramble: per-side key override in decoding recomputes one side's PRF labels and unframing key only (Section 4.4); 5 seeds $\times$ 10 frames each direction. Collision: for every encoded position, enumerate all same-context alternative edges whose PRF label equals the original and re-decode with the substituted carrier; 5 seeds $\times$ 5 frames, 11775 substitution trials.

A.4 PROMPT CONTROL UNDER THE DETECTION SUITE

The reported carriers use prompt v2. To separate the prompt from the rest of the pipeline, we re-ran the same closed loop with prompt v1, sharing every flag (frames, seeds, positions, repetition, retry budget, output-token budget, cover count) and differing only in the prompt; that arm also encoded and recovered all 10 frames, at 32 496 provider calls against 44 090. Both corpora were scored by the same nine detectors against the same 500 covers.

Table 5: **Key isolation under a decoder-side key intervention.** Five master-key seeds $\times$ ten matched frames (192 physical positions, $q_A = q_B = 1$, $r = 1$), carriers held fixed while one side's label and unframing key are overridden. BER (9600 bits per intervention, averaged over seeds, including framing and padding) is over the affected stream's PRF labels; chance is 0.5. Counts are of 50 frames; "preserved" requires bit-for-bit equality with normal decoding.

Intervention	Affected-stream BER	A payload	B payload
A-side scrambled	0.490	0/50	50/50 *preserved*
B-side scrambled	0.502	50/50 *preserved*	0/50

Table 6: Prompt control. One-sided 95% upper bounds, 2400 carriers per arm against the same 500 covers; the pre-registered pass line is ≤ 0.55.

Detector	Prompt v1	Prompt v2
char_entropy	0.612	0.476
rep4	0.525	0.500
punct_digit	0.349	0.519
sent_len_mean	0.693	0.479
sent_len_std	0.526	0.503
n_sentences	0.527	0.503
top5	0.534	0.549
len_chars	0.732	0.478
bigram_xent (cross-fitted)	0.563	0.470

Carriers under prompt v1 are longer (53.9 vs 47.2 characters; covers 48.4). Four of the nine detectors exceed the line under prompt v1 (length, sentence length, character entropy, and bigram cross-entropy); all nine stay within it under prompt v2.

A.5 Coordinate-level warden

The natural detector for a keyed coordinate is not a surface statistic but the coordinate distribution itself: an adversary who knows the protocol, the channel P_0, and the knowledge base can run the public extractors on any text and read off its coordinate (C, e). The construction predicts that this statistic is uninformative. For a position with target symbol t_p the encoder accepts a draw exactly when the keyed label matches t_p, and $\Pr[\text{accept} \mid (C, e)] = 2^{-q} p_{\text{render}}(C, e)$ does not depend on t_p, so the accepted coordinate distribution equals the public channel up to the rendering gate.

Legal coordinate space. A context is licensed only inside its own cluster, so the 64 edges and 12 contexts do not form 768 admissible pairs. The legal space is 4 clusters $\times$ 16 edges $\times$ 3 contexts $=$ 192 cells. Coordinates are read from the sender-side record of the published carriers (`enc_edge`/`enc_context`, $n = 2400$) and from the cover corpus used by Table 3 ($n = 500$).

Warden measurements. A cross-fitted coordinate-only classifier, scored by the log-likelihood ratio of a cell under the two empirical distributions with add-one smoothing (0.5 = chance), reaches AUROC 0.5114 on the joint coordinate, 0.4941 on the cluster alone, and 0.4964 on the edge alone. The cluster marginals are 0.207/0.194/0.305/0.294 for carriers against 0.216/0.230/0.282/0.272 for covers (total variation 0.0448, permutation $p = 0.209$ over 3000 resamples). No per-cluster contrast is significant, and the per-cluster total variations are far below what the sample sizes would resolve.

Table 7: Per-cluster coordinate contrast (48 legal cells per cluster; permutation test on total variation, 3000 resamples, seed 20260922).

Cluster	Carriers	Covers	Total variation	Permutation p
flow	497	108	0.302	0.164
node	466	115	0.289	0.095
path	732	141	0.217	0.777
port	705	136	0.261	0.200

Extractor behaviour. The warden observes the same extraction the receiver performs: of 2400 positions, 2394 decode to the encoded edge and none erases. Reproducing these numbers requires the cover corpus asserted by `experiments/run_coordinate_warden.py`; the repository holds several pivot cover sets and only one of them is the corpus Table 3 consumes.

A.6 AS-REPORTED BASELINE NUMBERS

Prior generative steganography reports capacity per token or per word, on English corpora and models that differ from ours: 3–7 bits per word for RNN-Stega, whose steganalysis runs at 3–5 (Yang et al., 2018), 1.23–2.49 bpw for Graph-Stega, a unit its paper never expands (Yang et al., 2020b), 2.17–2.60 bits per word for VAE-Stega, the operating points of its detection study (Yang et al., 2020a), 3.09 bits per token for Meteor (Kaptchuk et al., 2021), 3.48–5.76 bits per token for Discop (Ding et al., 2023), and 5.147–5.650 bits per word for ADG (Zhang et al., 2021). These cannot be placed in a column beside ours, and not merely because the units differ: the same method re-run under a different model shifts the reported rate substantially. ADG reports 5.147 bits per word, while a re-run of the same method gives 3.07–5.39 bits per token (Ding et al., 2023); Meteor reports 3.09, and the same re-run gives 2.91–5.19. None of the six reports a numeric extraction accuracy, claiming correctness by construction instead.

Detectability is reported in the other direction but varies too widely to tabulate: Discop reports a detection error at chance (49.50–50.45%) and ADG an accuracy of 0.496–0.548, while VAE-Stega reports 70.7–72.0% for itself and 81.2–81.3% for RNN-Stega (Yang et al., 2020a). Those figures come from different detectors on different reference texts and are not ours, so we read them as context for our own near-chance result (Section 4.5), not as a comparison.

Table 8: Capacity as reported by the original authors, under their own units, languages, and models. The rows are not commensurable with one another or with this work, and none is used in a comparison.

System	Capacity (as reported)	Language and model
RNN-Stega	3–7 bpw (capacity); 3–5 (steganalysis)	English, LSTM
Graph-Stega	1.23–2.49 bpw	English, GRU encoder + LSTM decoder + KG
VAE-Stega	2.17–2.60 bpw (detection operating points)	English, variational auto-encoder
Meteor	3.09 bits/token	English, GPT-2
Discop	3.48–5.76 bits/token	English, GPT-2
ADG	5.147–5.650 bits/word	English, 2-layer LSTM

A.7 PROTOCOL MATRIX, STRESS CHANNEL, AND DIAGNOSTICS

Per-configuration success counts and net rate. Table 9 reports the success counts and denominators for every configuration of the synthetic channel. The main matrix runs 5 seeds $\times$ 1000 frames (5000 per configuration) at each payload length $L \in \{0, 1, 5, 7, 8, 16, 32, 64, 128\}$, for both the single-stream (q=1) and dual-stream (q=2) configurations and both the uniform and Zipf-skewed stress channels; every configuration round-trips all frames with zero failures and zero erasure. At these operating points the net rate is $R_{\text{net}} = L/256$ (single) and $L/128$ (dual), so the dual-stream gain $\Delta R_B = L/256$ is identical over all five seeds and both channels. The finite-alphabet boundary runs 5 seeds $\times$ 200 frames (1000 per configuration) at $L = 8$ for $q \in \{3, 4\}$ and reports the retry-exhaustion counts.

Zipf-stress channel details. Skewed weights per edge; validation guarantees (coverage, finiteness, positivity, minimum normalized probability) applied before each run; stress runs use the same PRF/nonce domain separation as uniform runs.

Efficiency. The pivot recipe uses 240 physical positions (80 logical positions with $r = 3$) and three receiver votes per position. It consumes 44,070 calls across ten encoded frames, or approximately 4,407 calls per frame; the coordinate recipe consumes approximately 1,139 calls per encoded frame. These totals include the sender, receiver, and replay calls reported in Table 2.

Table 9: Frame success counts per configuration on the synthetic channel. Main matrix: 5 seeds $\times$ 1000 frames $=$ 5000 per cell at every payload length $L \in \{0, 1, 5, 7, 8, 16, 32, 64, 128\}$ bits (uniform and Zipf-stress identical; zero failures, zero erasure). Finite-alphabet boundary: 5 seeds $\times$ 200 frames $=$ 1000 per cell at $L = 8$.

Configuration	**Uniform**	**Zipf-stress**
Single-stream (q=1)	5000/5000 per L	5000/5000 per L
Dual-stream (q=2)	5000/5000 per L	5000/5000 per L

Configuration (L=8)	**Channel**	**Success**	**Failure rate**
q=3	uniform	991/1000	0.9%
q=3	stress	982/1000	1.8%
q=4	uniform	141/1000	85.9%
q=4	stress	52/1000	94.8%

A.8 Dataset provenance and ambiguity-resource boundary

Knowledge base provenance. The dual-meaning seed is the author-constructed `synthetic_closed_world_v0.1` (SHA-256 `818e6ad2...3a3c703720bb928`). Structure: A = city-function units, B = computer-network units, 4 clusters (`node`/`path`/ `port`/`flow`), each with 4 A + 4 B units, full Cartesian product of 16 candidate edges per cluster (64 in total), 12 public contexts. A standalone parser validates the seed, expands units, enumerates edge/context tasks, and generates test fixtures. It is a prototype-curated micro-library, *not* automatically extracted from an external corpus.

Ambiguity-resource boundary (ChinesePun / PunEval / AmbiEnt). The mentioned resources are cited only as linguistic background for homographic ambiguity and dual-reader verification:

> "ChinesePun introduced a Chinese pun-generation dataset containing homophonic and homographic puns annotated with pun-word pairs. It provides a useful linguistic precedent for homographic ambiguity, but targets humorous pun generation rather than semantic communication. Iphis therefore uses an independently constructed synthetic graph of A/B factual units, contexts, and candidate edges; ChinesePun examples and annotations are not used as the protocol substrate."

ChinesePun examples and annotations are not used as source, training, or evaluation data.

A.9 Annotation protocol and reproducibility manifests

Annotation protocol. The naturalness audit uses a blinded LLM judge on 100 stego and 100 cover texts, with three integer ratings per text (fluency, naturalness, coherence). The prompt contains no source-domain or steganography labels; sample construction and scores are recorded in the run manifest.

Manifest excerpts. The reproducibility manifest records the source digest, frozen prompt identifiers, model names, temperatures, seeds, nonce metadata, and command lines used for each reported table. Generated text is produced for this study from the synthetic channel; the manuscript's AI-use and ethics statements cover model assistance, synthetic data, and licensing.