

Opacity Is Not Privacy: Information-Constrained Latent Communication for Multi-Agent LLM Systems

Yao Gong^{1,2*}, Long Xiang^{4,*}, Di Liang^{2,3,†}, Peiyang Liu¹,
Zekai Lin², Mengze Hong³, Xianjie Wu⁵, Minlong Peng^{2,3,†}

¹Peking University, ²Tencent Hunyuan, ³Tencent Yuanbao

⁴Sun Yat-sen University, ⁵Beijing Information Science&Technology University

{mryaogong, islongxiang}@gmail.com; wuxianjie@bistu.edu.cn

{liangd17, mlpeng16}@fudan.edu.cn

Abstract

Latent communication provides an efficient alternative to text-based exchange in multi-agent LLM systems by transmitting continuous hidden representations instead of natural-language messages. However, opacity is not privacy: latent messages may still carry private prompts, agent memory, or task-irrelevant sensitive information recoverable by downstream agents or adversaries. We study this risk through *communicated latent messages*, the edge-level representations that cross agent boundaries and define the privacy interface of latent multi-agent systems. We identify two coupled leakage mechanisms: *instantaneous leakage* from individual messages and *propagation leakage* accumulated across multi-round transcripts.

We propose **Information-Constrained Latent Communication (ICLC)**, a model-agnostic communication-layer framework that constrains latent information exposure without modifying backbone LLM parameters. ICLC combines a receiver-conditioned latent bottleneck, adversarial regularization over messages and transcripts, and topology-aware gates that suppress unnecessary communication edges. Across private-context question answering, collaborative reasoning, and tool-use planning, ICLC preserves the utility benefit of latent communication while reducing average leakage from 0.747 to 0.398 compared with raw latent communication. These results suggest that privacy in latent multi-agent systems should be treated as a dynamic information-flow problem over messages, edges, and interaction rounds.

1 Introduction

Large language models (LLMs) are increasingly deployed as multi-agent systems, where role-specialized agents coordinate through communica-

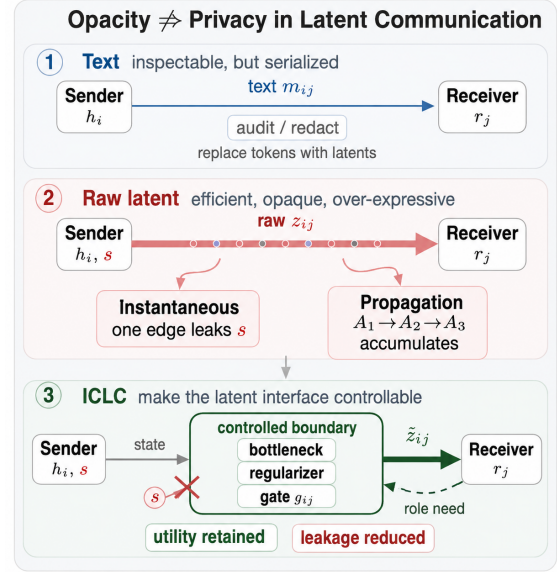


Figure 1: **Opacity is not privacy in multi-agent latent communication.** Text communication provides an inspectable but serialized boundary. Raw latent messages improve bandwidth but expose dense sender information, causing *instantaneous leakage* on individual edges and *propagation leakage* through multi-round interaction. ICLC inserts an information-constrained boundary that uses receiver-conditioned bottlenecks, leakage regularization, and topology-aware gating to preserve role-relevant utility while suppressing private signals.

tion, tool use, memory sharing, and multi-turn protocols. Systems such as AutoGen, MetaGPT, and ChatDev typically rely on natural-language messages to exchange plans, observations, critiques, and tool outputs (Li et al., 2023a; Chen et al., 2024; Wu et al., 2024; Hong et al., 2024; Qian et al., 2024). This text interface is convenient because messages are discrete, inspectable, and easy to log. Yet text also becomes a narrow serialization boundary: it converts high-dimensional internal states into token sequences, adding overhead and potentially discarding information useful for later coordination. Recent work has therefore begun to explore

¹* Equal Contribution. [†] Corresponding Author.

²This work was completed by Yao Gong and Long Xiang under Di Liang’s supervision.

latent communication, where agents exchange continuous representations such as output embeddings, hidden states, activations, or KV-cache states instead of decoded text (Pham et al., 2024; Ramesh and Li, 2025; Du et al., 2025; Zou et al., 2025; Zheng et al., 2025; Shi et al., 2026; Kriuk and Ng, 2025; Yu et al., 2026).

We focus on the object that actually crosses an agent-to-agent edge, which we call a *communicated latent message*: a continuous representation derived from a sender’s inference-time state and consumed by a receiver through a latent interface. While such messages can improve bandwidth, they shift the privacy boundary from an editable text channel to an opaque representation channel. A communicated latent message may encode private attributes, confidential context, memorized content, or task-irrelevant details that are difficult to inspect, redact, or audit. Thus, as illustrated in Figure 1, opacity alone does not provide privacy.

This creates two coupled leakage modes. *Instantaneous leakage* occurs when a single transmitted message reveals more about the sender’s private context than the receiver needs. *Propagation leakage* occurs when weak private signals are transformed, retransmitted, and accumulated across interaction rounds until they become recoverable from downstream agents or latent transcripts. These risks cannot be addressed by treating each vector independently: compressing individual messages may still allow multi-hop accumulation, while sparsifying edges alone does not ensure safe latent exchange.

We propose **Information-Constrained Latent Communication** (ICLC), a model-agnostic framework that regulates information flow at the latent communication boundary. ICLC does not modify backbone LLM parameters. Instead, it controls how a sender’s internal state is converted into an edge-specific latent message for a receiver. The framework combines three mechanisms: a receiver-conditioned latent bottleneck that limits message capacity, adversarial leakage regularization that suppresses sensitive information recoverable from messages and transcripts, and a topology-aware gate that reduces unnecessary exposure across the agent graph. Together, these components implement data minimization at the latent communication boundary.

We evaluate ICLC on private-context question answering, collaborative reasoning, and tool-use planning under attribute inference, private-context

membership inference, and secret reconstruction attacks. Experiments show that raw latent communication improves utility over text-based communication but substantially amplifies privacy leakage. ICLC improves this privacy–utility trade-off by reducing both instantaneous and propagation leakage while preserving the collaboration benefits of latent exchange. Our contributions are: (1) we define *communicated latent messages* as the privacy-relevant objects in multi-agent latent communication; (2) we identify instantaneous and propagation leakage as two coupled failure modes; (3) we propose ICLC, a communication-layer framework that improves privacy–utility trade-offs through bottlenecking, adversarial leakage regularization, and topology-aware exposure control.

2 Related Work

Multi-agent LLM systems. Recent LLM-based multi-agent systems decompose complex tasks into interacting roles, using language for planning, critique, tool use, and software development (Li et al., 2023a; Chen et al., 2024; Wu et al., 2024; Hong et al., 2024; Han et al., 2026; Qian et al., 2024). These systems motivate explicit inter-agent communication interfaces, but most assume text messages as the default boundary. ICLC studies the privacy implications of replacing this inspectable boundary with communicated latent messages.

Latent and cache-based communication. CIPHER enables multi-agent debate through output embeddings (Pham et al., 2024); activation communication directly combines intermediate states between language model agents (Ramesh and Li, 2025); and KVComm studies selective sharing of KV-cache states (Shi et al., 2026; Kriuk and Ng, 2025; Asif et al., 2026; Du et al., 2025; Lin et al., 2026; Zou et al., 2025; Liu et al., 2026a; Zheng et al., 2025; Yu et al., 2026). These works demonstrate that non-textual communication can be useful and efficient. ICLC is complementary: it asks what private information becomes exposed when these continuous objects cross agent boundaries, and how such exposure can be controlled.

Representation leakage and agent privacy. Dense representations are not inherently private: embedding inversion and internal-state inversion attacks can recover substantial information about original inputs (Morris et al., 2023; Li et al., 2023b; Dong et al., 2025). Recent work also shows that

LLM agents may leak information through memory, tools, or communication channels (Wang et al., 2025a; He et al., 2025; Wang et al., 2025b; Patil et al., 2025; Zharmagambetov et al., 2025; Li et al., 2025; Xue et al., 2026a; Wang et al., 2025c; Zeng et al., 2025; Liu et al., 2025a; El Yagoubi et al., 2026). ICLC targets a lower-level communication object than final outputs or text logs: the message exposed on each directed agent-to-agent edge.

3 Method

We present **Information-Constrained Latent Communication (ICLC)**, a communication-layer framework for controlling information exposure in multi-agent latent communication. ICLC operates at the boundary where an agent’s internal inference state is converted into an edge-specific communicated latent message. As shown in Figure 2, the framework combines three controls: a local substrate extractor, a receiver-conditioned bottleneck, and a topology-aware exposure gate. During training, leakage critics probe both individual messages and multi-round transcripts; at inference time, these critics are discarded.

3.1 Latent Communication and Leakage

We consider a multi-agent LLM system with agents $\mathcal{A} = \{A_1, \dots, A_N\}$ interacting over T rounds. Each task instance contains a shared instruction q , a target output y , and agent-specific private contexts $\mathcal{X} = \{x_i\}_{i=1}^N$. The private context of agent A_i may include sensitive targets s_i^r , where $r \in \mathcal{R}$ denotes a privacy risk such as private attributes, private-context membership, or confidential spans.

At round t , agents communicate over a directed graph $\mathcal{G}^{(t)} = (\mathcal{A}, \mathcal{E}^{(t)})$. Let

$$Z_i^{(t-1)} = \{z_{k \rightarrow i}^{(t-1)} : (k, i) \in \mathcal{E}^{(t-1)}\} \quad (1)$$

be the latent messages received by A_i in the previous round, with $Z_i^{(0)} = \emptyset$. The internal inference state of A_i is

$$H_i^{(t)} = F_i(x_i, q, \text{Fuse}_i(Z_i^{(t-1)})), \quad (2)$$

where F_i denotes the backbone agent module.

The privacy-relevant object is not the full internal state $H_i^{(t)}$, but the *communicated latent message* actually exposed from A_i to A_j . We represent this communication boundary as

$$z_{i \rightarrow j}^{(t)} = \mathcal{C}_\Theta(H_i^{(t)}, q, e_i, e_j, t), \quad (3)$$

where e_i and e_j are sender and receiver role embeddings. In general, $H_i^{(t)} \neq z_{i \rightarrow j}^{(t)}$: ICLC learns the boundary map $\mathcal{C}_\Theta$ while leaving the backbone state internal to the sender.

We consider honest-but-curious receivers or coalitions that follow the protocol but train post-hoc predictors on the latent messages delivered to them. For a coalition $\mathcal{C} \subseteq \mathcal{A}$, the observable transcript is

$$\mathcal{O}_\mathcal{C}^{1:T} = \bigcup_{t=1}^T \left\{ z_{i \rightarrow j}^{(t)} : (i, j) \in \mathcal{E}^{(t)}, j \in \mathcal{C} \right\}. \quad (4)$$

The transcript contains only communicated latent messages and excludes downstream hidden states. We target empirical leakage reduction under attribute inference, private-context membership inference, and secret reconstruction; formal differential privacy guarantees are outside the scope of this work.

Figure 3 illustrates why both notions are needed. A single communicated latent message may already reveal sensitive information locally, but sensitive information may also become recoverable only after repeated transmission and aggregation across rounds. The first failure mode is edge-local, whereas the second is interaction-level.

We therefore define two leakage modes. *Instantaneous leakage* measures sensitive information recoverable from a single edge-level message:

$$\mathcal{I}_{\text{inst}}^r = \mathbb{E}_{t, (i, j)} I(s_i^r; z_{i \rightarrow j}^{(t)} | q, e_j). \quad (5)$$

Propagation leakage measures sensitive information recoverable from a multi-round transcript:

$$\mathcal{I}_{\text{prop}}^r(\mathcal{C}) = I(\{s_i^r\}_{i=1}^N; \mathcal{O}_\mathcal{C}^{1:T} | q). \quad (6)$$

These mutual-information quantities define conceptual leakage criteria rather than direct training losses. ICLC approximates them with capacity control, exposure control, and adversarial recoverability objectives.

3.2 Information-Constrained Message Passing

ICLC instantiates the communication function in Eq. 3 through local substrate extraction, stochastic bottlenecking, topology-aware gating, and receiver-side fusion.

Local substrate extraction. For each sender A_i , ICLC first extracts a local latent substrate from selected hidden layers:

$$u_i^{(t)} = \text{Extract}_\rho(H_{i, \mathcal{L}}^{(t)}, q) \in \mathbb{R}^{K \times d_c}, \quad (7)$$

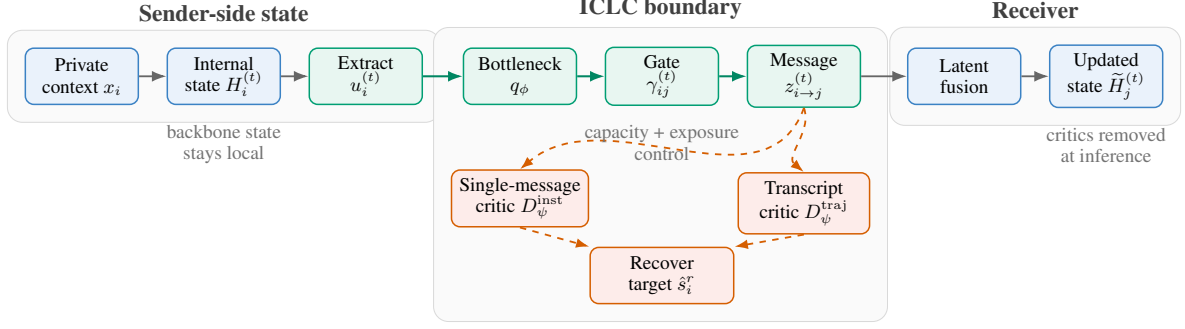


Figure 2: **Overview of Information-Constrained Latent Communication.** ICLC inserts a controlled communication boundary between a sender’s internal state and the edge-level latent message observed by a receiver. The extractor forms a local substrate, the bottleneck limits message capacity, and the gate controls which latent information is exposed. Dashed red arrows denote training-time leakage critics that probe message-level and transcript-level recoverability; they are discarded at inference time.

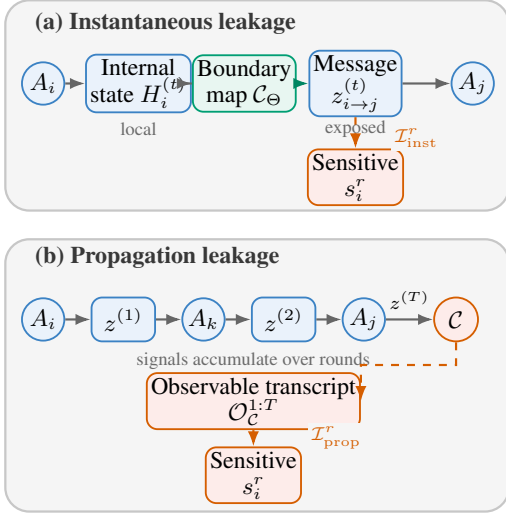


Figure 3: **Two leakage modes in latent communication.** Instantaneous leakage arises from one exposed edge-level message, whereas propagation leakage arises when weak signals accumulate across a multi-round transcript observed by a receiver or coalition.

where $H_{i,\mathcal{L}}^{(t)}$ denotes selected hidden layers, K is the number of latent slots, and d_C is the communication dimension. The substrate $u_i^{(t)}$ remains local and is never directly transmitted.

Task-conditioned bottleneck. Given $u_i^{(t)}$, ICLC produces a receiver-conditioned stochastic candidate message:

$$[\mu_{ij}^{(t)}, \log \sigma_{ij}^{(t)}] = B_\phi(u_i^{(t)}, q, e_i, e_j, t), \quad (8)$$

$$\tilde{z}_{i \rightarrow j}^{(t)} = \mu_{ij}^{(t)} + \sigma_{ij}^{(t)} \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \quad (9)$$

Let $q_{ij}^{(t)} = q_\phi(z | u_i^{(t)}, q, e_i, e_j, t)$ denote this posterior, and let $p_j = p_\eta(z | q, e_j)$ denote a receiver-

conditioned prior. The capacity penalty is

$$\mathcal{L}_{\text{cap}} = \mathbb{E}_{t,(i,j)} \left[D_{\text{KL}} \left(q_{ij}^{(t)} \| p_j \right) \right]. \quad (10)$$

This bottleneck encourages edge-specific data minimization: different receivers need not receive the same latent information.

Topology-aware exposure gate. Capacity control limits what may be encoded, but it does not decide whether an edge should be active. ICLC therefore learns an exposure gate:

$$\pi_{ij}^{(t)} = G_\omega(u_i^{(t)}, q, e_i, e_j, t), \quad (11)$$

$$\gamma_{ij}^{(t)} \sim \text{HardConcrete}(\pi_{ij}^{(t)}), \quad (12)$$

$$z_{i \rightarrow j}^{(t)} = \gamma_{ij}^{(t)} \odot \tilde{z}_{i \rightarrow j}^{(t)}. \quad (13)$$

If all slots are inactive, no message is exposed on that edge. We regularize the expected exposure surface:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t,(i,j)} \left[\|\gamma_{ij}^{(t)}\|_0 \right]. \quad (14)$$

Unlike standard communication sparsification, this gate is optimized as an exposure-control mechanism: inactive edges produce no communicated latent artifact for curious receivers or coalitions.

Receiver-side fusion. The receiver consumes active messages through a latent interface. Let $Z_j^{(t)}$ denote the active messages received by A_j at round t . The receiver update is

$$\tilde{H}_j^{(t)} = \text{Fuse}_\tau \left(H_j^{(t)}, \text{Agg}_\tau(Z_j^{(t)}) \right). \quad (15)$$

The updated state $\tilde{H}_j^{(t)}$ is then used for task prediction or subsequent message generation. Thus, the bottleneck controls *what* can be encoded, while the gate controls *where* latent information is exposed.

3.3 Leakage Regularization and Optimization

Capacity and exposure control reduce the opportunity for leakage, but they do not identify which information is sensitive. We therefore add training-time leakage critics that test whether sensitive targets remain recoverable from individual messages and multi-round transcripts. For compactness, define $\xi_{ij}^{(t)} = (z_{i \rightarrow j}^{(t)}, q, e_i, e_j, t)$ and $\chi_i^C = (\mathcal{O}_C^{1:T}, q, e_i)$.

For each target type r , the single-message and trajectory-level attacker utilities are

$$\mathcal{R}_{\text{inst}}^r = \mathbb{E}_{t,(i,j)} \log D_{\psi_r}^{\text{inst}}(s_i^r | \xi_{ij}^{(t)}), \quad (16)$$

$$\mathcal{R}_{\text{traj}}^r = \mathbb{E}_{\mathcal{C},i} \log D_{\psi_r}^{\text{traj}}(s_i^r | \chi_i^C). \quad (17)$$

The prediction head is instantiated as a classifier for attribute inference, a binary predictor for private-context membership inference, and an autoregressive decoder for secret reconstruction. We aggregate the leakage utility as

$$\mathcal{R}_{\text{leak}} = \sum_{r \in \mathcal{R}} (\mathcal{R}_{\text{inst}}^r + \mathcal{R}_{\text{traj}}^r). \quad (18)$$

These critics are used only as training-time regularizers and are discarded at inference time. In experiments, privacy is evaluated with held-out attackers trained after the communication modules are fixed.

Let $\Theta = \{\rho, \phi, \omega, \tau\}$ denote the trainable communication-layer parameters. Given the generated latent transcript $\mathcal{Z}^{1:T}$, the task loss is

$$\mathcal{L}_{\text{task}} = -\mathbb{E} \log p_{\Theta}(y | q, \mathcal{X}, \mathcal{Z}^{1:T}). \quad (19)$$

The overall objective is

$$\min_{\Theta} \max_{\psi} \mathcal{L}_{\text{task}} + \beta \mathcal{L}_{\text{cap}} + \alpha \mathcal{L}_{\text{flow}} + \lambda \mathcal{R}_{\text{leak}}. \quad (20)$$

The critics maximize $\mathcal{R}_{\text{leak}}$ to strengthen attacks, whereas the communication module minimizes it to reduce sensitive recoverability. This objective can be viewed as a Lagrangian relaxation of task optimization under constraints on message capacity, exposure surface, and empirical leakage.

At inference, ICLC uses no leakage critics or sensitive labels. It replaces sampling with deterministic gated posterior means:

$$z_{i \rightarrow j}^{(t)} = \mathbb{I}[\sigma(\pi_{ij}^{(t)}) > \delta_g] \odot \mu_{ij}^{(t)}. \quad (21)$$

The backbone agents remain frozen; ICLC adds only lightweight wrappers for extraction, bottlenecking, gating, and fusion.

4 Experiments

Evaluation questions. Our experiments are designed to answer four questions. **Q1:** Does raw latent communication expose more private information than text-based communication? **Q2:** Can ICLC reduce both message-level and transcript-level leakage while preserving the utility advantage of latent communication? **Q3:** Which component of ICLC accounts for each type of leakage reduction? **Q4:** How sensitive is ICLC to bottleneck capacity, adversarial strength, and exposure-control strength? Q1–Q2 are answered by the main experiments, while Q3–Q4 are analyzed in Section 5.

4.1 Experimental Setup

Tasks. We evaluate ICLC on three representative multi-agent settings: private-context question answering, collaborative reasoning, and tool-use planning. Private-context QA is built on HotpotQA (Yang et al., 2018), where supporting paragraphs are distributed across agents. Collaborative reasoning uses GSM8K (Cobbe et al., 2021), MuSiQue (Trivedi et al., 2022), and ProofWriter (Tafjord et al., 2021), where agents receive partial observations or complementary reasoning roles. Tool-use planning uses ToolBench (Qin et al., 2024), with planner, executor, verifier, and memory agents collaborating to compose tool calls. In all settings, protected information is placed in agent-local contexts but is not required for producing the final answer. Protected content includes synthetic canaries, private attributes, protected distractors, mock API keys, and confidential constraints.

Baselines and protocol. We compare all methods under the same agent roles, prompts, task inputs, communication topology, and backbone agents. The baselines cover three families of communication and sanitization strategies: **text-based communication**, including Text-MAS and Text-Redaction-MAS; **unstructured latent communication**, including Raw-Latent-MAS, Noise-Latent-MAS, and Calibrated-Noise-MAS; and **structured latent sanitization**, including Projection-MAS, Learned-Projection-MAS, and Adv-Sanitize-MAS. All baselines are tuned on validation data using the same privacy–utility selection criterion as ICLC, and no test examples are used for model or threshold selection. Final privacy results are computed with held-out attackers trained only after the communication modules are fixed. Implementation de-

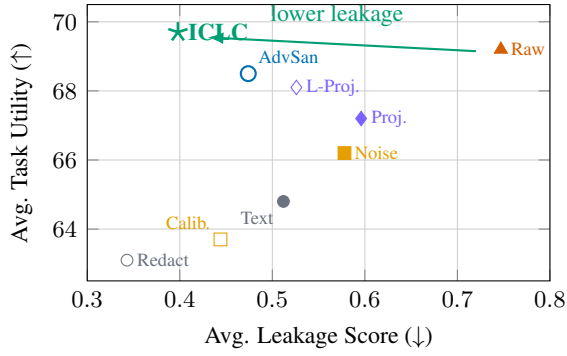


Figure 4: **Privacy-utility frontier.** ICLC preserves the utility of Raw-Latent-MAS while substantially reducing leakage.

tails, task construction, and full tuning protocols are provided in Appendix B.

Attacks and metrics. We evaluate attribute inference, private-context membership inference, and secret reconstruction. Local attacks observe a single communicated latent message $z_{i \rightarrow j}^{(t)}$, while trajectory attacks observe the multi-round transcript $\mathcal{O}_C^{1:T}$ available to a curious receiver or coalition. We do not assume access to downstream hidden states. Privacy is reported as a lower-is-better leakage score normalized to $[0, 1]$. Task utility is measured by EM/F1, accuracy, planning success, or pass rate depending on the task.

4.2 Main Results: Privacy-Utility Trade-off

Table 1 reports the main privacy-utility comparison, and Figure 4 visualizes the corresponding frontier. The results support three observations.

Raw latent communication improves collaboration but amplifies leakage. Compared with Text-MAS, Raw-Latent-MAS improves average utility from 64.8 to 69.2, but also increases average leakage from 0.512 to 0.747. This gap supports our central claim that latent opacity should not be interpreted as privacy: continuous messages can preserve dense private signals that are absent from, or easier to filter in, text channels.

Stronger baselines expose the privacy-utility tension. Text-Redaction-MAS achieves the lowest leakage overall, but reduces average utility to 63.1, losing most of the benefit of latent exchange. Calibrated-Noise-MAS lowers leakage by degrading task performance. Learned-Projection-MAS and Adv-Sanitize-MAS are more competitive, but remain substantially leakier than ICLC because they mainly operate at the single-message level.

ICLC provides the strongest latent privacy-utility trade-off. Among latent communication

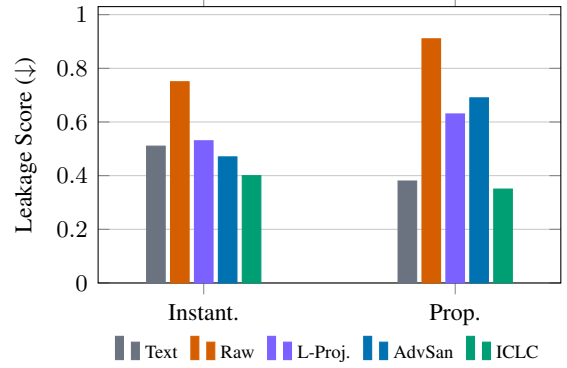


Figure 5: **Instantaneous and propagation leakage.** Instantaneous leakage is measured from a single communicated latent message. Propagation leakage is measured from $T = 8$ latent transcripts. ICLC reduces both local message leakage and multi-round transcript leakage.

methods, ICLC obtains the lowest average leakage of 0.398 while preserving the highest average utility of 69.7. Compared with Raw-Latent-MAS, it reduces leakage by 0.349 with comparable task utility. We do not interpret the slight utility gain over Raw-Latent-MAS as improved backbone capability, since all backbone agents and task prompts are unchanged. Rather, the bottleneck and gate may remove redundant or distracting latent signals, improving coordination in some interactions.

4.3 Leakage Decomposition

We next examine whether privacy leakage mainly comes from individual latent messages or from multi-round accumulation. Figure 5 decomposes leakage into instantaneous and propagation components for representative methods. Instantaneous leakage is the average single-message held-out attack score. Propagation leakage is the average transcript-level held-out attack score at $T = 8$ communication rounds. This separation is important because a method can reduce recoverability from an individual message while still allowing weak private signals to accumulate across the interaction trajectory.

Raw-Latent-MAS substantially increases both forms of leakage compared with Text-MAS. Learned-Projection-MAS reduces message-level leakage moderately, while Adv-Sanitize-MAS lowers single-message leakage but leaves high propagation leakage. This indicates that message-level sanitization alone cannot prevent sensitive signals from being retransmitted and amplified over multiple rounds.

ICLC reduces both leakage modes. Compared with Raw-Latent-MAS, ICLC lowers instantaneous

Method	Task Utility ($\uparrow$)				Privacy Leakage ($\downarrow$)			
	QA	Reason.	Plan	Avg.	Attr.	Mem.	Recon.	Avg.
Text-MAS	68.2	65.7	60.4	64.8	0.535	0.518	0.482	0.512
Text-Redaction-MAS	66.8	63.9	58.6	63.1	0.362	0.348	0.318	0.343
Raw-Latent-MAS	71.5	70.2	65.8	69.2	0.779	0.757	0.705	0.747
Noise-Latent-MAS	69.0	67.4	62.3	66.2	0.601	0.583	0.549	0.578
Calib.-Noise-MAS	66.4	64.8	59.9	63.7	0.462	0.448	0.421	0.444
Projection-MAS	69.8	68.3	63.5	67.2	0.617	0.604	0.568	0.596
Learned-Proj.-MAS	70.4	69.1	64.8	68.1	0.548	0.529	0.502	0.526
Adv-Sanitize-MAS	70.8	69.5	65.1	68.5	0.492	0.481	0.450	0.474
ICLC	71.9	70.9	66.3	69.7	<u>0.428</u>	<u>0.405</u>	<u>0.360</u>	<u>0.398</u>

Table 1: **Main privacy–utility results.** QA, Reason., and Plan denote HotpotQA exact match, average reasoning accuracy, and ToolBench planning success. Avg. is the arithmetic mean within each block; leakage is measured by held-out attackers. Bold marks the best value; underline marks the best latent-communication result.

Variant	Util.	Msg.	Traj.	
	$\uparrow$	$\downarrow$	$t=5$	$t=8$
Full ICLC	69.7	0.398	0.33	0.35
w/o Bottleneck	69.1	0.520	0.58	0.70
w/o Adv. Critic	68.7	0.570	0.61	0.73
w/o Topo. Gate	69.4	0.468	0.76	0.88
w/o All	69.2	0.747	0.84	0.91

Table 2: **Compact component ablation.** Utility averages QA, reasoning, and planning tasks. Msg. and Traj. report held-out single-message and transcript-level attack leakage, respectively. The row “w/o All” corresponds to Raw-Latent-MAS.

leakage from 0.75 to 0.40 and $T = 8$ propagation leakage from 0.91 to 0.35. The stronger reduction in trajectory leakage supports the need for topology-aware exposure control and transcript-level regularization.

5 Ablation Study

We ablate ICLC to identify how each information-flow control contributes to the privacy gains observed in Section 4. This section answers two questions: which components reduce message-level and transcript-level leakage, and how sensitive ICLC is to the strengths of its capacity, recoverability, and exposure controls. All ablations use the same backbone agents, task splits, prompts, communication schedule, and held-out attacker protocol as in Section 4. Unless otherwise stated, each variant is evaluated after the communication module is fixed, and leakage is measured by held-out attackers. Table 2 reports a compact average view; detailed per-task and per-attack breakdowns are provided in Appendix G.

5.1 Component Contribution

Table 2 shows that all three controls contribute to the full leakage reduction, but they affect different leakage pathways. Removing any component increases leakage while leaving average utility close to full ICLC. This indicates that the privacy gains are not obtained by simply disabling collaboration.

The bottleneck limits message capacity. Removing the receiver-conditioned bottleneck increases average message leakage from 0.398 to 0.520. It also increases $t=8$ trajectory leakage from 0.35 to 0.70. This shows that unconstrained latent capacity is itself a privacy risk: even when the gate and adversarial objective remain active, a high-capacity edge-level representation can still carry task-irrelevant private information. The trajectory increase further suggests that higher-capacity messages provide more material for multi-round accumulation.

The adversarial critic suppresses recoverable sensitive directions. Removing the leakage critic gives the largest increase in message-level leakage, from 0.398 to 0.570. This result shows that compression alone is not sufficient. A low-capacity message can still allocate its limited capacity to sensitive but task-irrelevant directions unless training explicitly penalizes recoverability. The critic therefore complements the bottleneck: the bottleneck controls how much information can cross the boundary, while the critic discourages which information should be recoverable.

The topology gate controls the exposure surface. Removing the topology-aware gate causes the strongest increase in transcript-level leakage. Although its message leakage is lower than the no-critic variant, its $t=8$ trajectory leakage rises

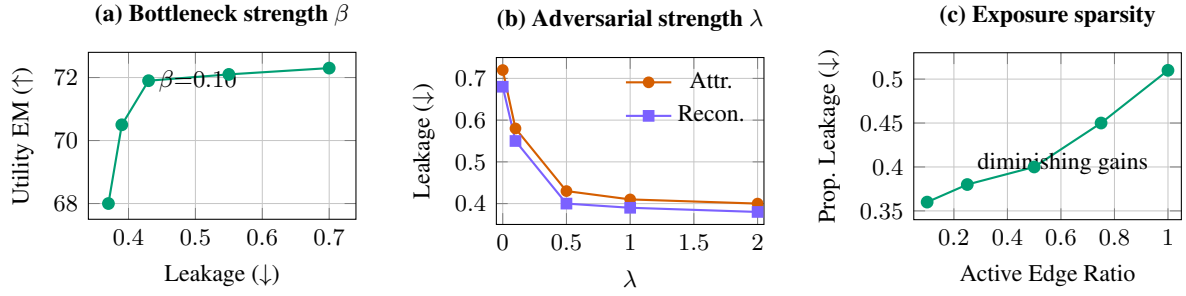


Figure 6: **Sensitivity to ICLC control strengths.** Panels sweep β on QA, λ on reasoning, and active-edge ratio on planning. Moderate control strengths yield the best privacy–utility balance.

to 0.88, close to Raw-Latent-MAS at 0.91. This asymmetry supports our central claim that propagation leakage is a system-level effect. A method can appear safer under isolated single-message attacks while still exposing enough repeated signals for a transcript-level attacker to recover private information. The gate reduces this risk by preventing unnecessary communicated latent artifacts from being created in the first place.

5.2 Sensitivity to Control Strengths

Figure 6 evaluates representative sweeps over the three ICLC control strengths. The panels should be read as trend analyses rather than direct cross-panel comparisons, since each sweep uses the task family where the corresponding control is most interpretable. Overall, ICLC has a broad useful operating region rather than a single brittle hyperparameter setting.

Increasing the bottleneck weight β initially reduces leakage with little utility loss, but overly strong bottlenecking eventually hurts task performance. This supports a data-minimization interpretation: the channel should suppress task-irrelevant private information without indiscriminately removing all coordination signal. Increasing the adversarial weight λ gives the largest leakage reduction at moderate values and then saturates, suggesting that the critic is most useful as a regularizer rather than as an overwhelming training objective. Finally, reducing the active edge ratio lowers propagation leakage, but the gains diminish at very sparse settings where communication becomes insufficient for multi-agent coordination. The corresponding utility drops and additional sensitivity tables are reported in Appendix H.

5.3 Takeaways

The ablation study supports three conclusions. First, bottlenecking, adversarial regularization, and topology gating are complementary: they control

capacity, recoverability, and exposure, respectively. Second, propagation leakage is not reducible to single-message leakage; it depends on the communication graph and the number of exposed latent artifacts over time. Third, ICLC preserves the utility benefit of latent communication because it constrains unnecessary information flow rather than eliminating collaboration. Appendix G provides full utility, attack, ablation, topology, round-scaling, and communication-efficiency results.

6 Conclusion

We studied privacy leakage in multi-agent latent communication, where agents exchange continuous internal-state messages rather than inspectable natural language. Our results show that opacity is not privacy: raw latent communication can improve collaboration while leaking sensitive information through edge-level messages and multi-round transcripts. We therefore frame privacy in latent multi-agent systems as a dynamic information-flow problem over messages, receivers, edges, and rounds.

We introduced **Information-Constrained Latent Communication (ICLC)**, a communication-layer framework that constrains information exposure without modifying backbone language models. By combining receiver-conditioned bottlenecks, adversarial leakage regularization, and topology-aware exposure gates, ICLC reduces average leakage from 0.747 to 0.398 over raw latent communication while preserving latent-exchange utility. Ablations further show that these controls are complementary: bottlenecks limit message capacity, adversarial critics reduce recoverable sensitive directions, and topology-aware gates suppress multi-round propagation. Together, these findings suggest that privacy in latent multi-agent systems should be treated as a first-class protocol design objective, rather than as a byproduct of representation opacity or post-hoc output filtering.

Limitations

ICLC reduces empirical recoverability under the attacker models evaluated, but does not provide formal differential privacy or information-theoretic secrecy guarantees. Our results should therefore be viewed as evidence that communication bottlenecks, adversarial regularization, and exposure control improve the empirical privacy–utility trade-off, rather than proof that sensitive information is unrecoverable. A second limitation is that task-relevant and private information may be strongly entangled. When private information is necessary for the public task, useful communication may inevitably transmit correlated signals. ICLC can reduce unnecessary exposure, but cannot determine whether information is authorized for a given receiver or action. Practical deployments should therefore combine latent communication controls with access policies and audit mechanisms. Finally, held-out attackers may still underestimate future adaptive attackers with more data, architecture search, or side information. Future work should explore stronger adaptive evaluations, formal privacy guarantees, and dynamic privacy budgets for latent communication interfaces.

References

- Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. [Deep variational information bottleneck](#). In *ICLR*.
- Sadia Asif, Mohammad Mohammadi Amiri, Momin Abbas, Prasanna Sattigeri, and Karthikeyan Natesan Ramamurthy. 2026. [LCGuard: Latent communication guard for safe KV sharing in multi-agent systems](#). *arXiv preprint arXiv:2605.22786*.
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. [AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors](#). In *ICLR*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *arXiv preprint arXiv:2110.14168*.
- Chang Dai, Hongyu Shan, Mingyang Song, and Di Liang. 2025. [Hope: Hyperbolic rotary positional encoding for stable long-range dependency modeling in large language models](#). *arXiv preprint arXiv:2509.05218*.
- Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. 2019. [TarMAC: Targeted multi-agent communication](#). In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1538–1546. PMLR.
- Tian Dong, Yan Meng, Shaofeng Li, Guoxing Chen, Zhen Liu, and Haojin Zhu. 2025. [Depth gives a false sense of privacy: LLM internal states inversion](#). In *34th USENIX Security Symposium*, pages 1629–1648, Seattle, WA. USENIX Association.
- Zhuoyun Du, Runze Wang, Huiyu Bai, Zouying Cao, Xiaoyong Zhu, Yu Cheng, Bo Zheng, Wei Chen, and Haochao Ying. 2025. [Enabling agents to communicate entirely in latent space](#). *arXiv preprint arXiv:2511.09149*. Accepted to ACL 2026.
- Faouzi El Yagoubi, Godwin Badu-Marfo, and Ranwa Al Mallah. 2026. [AgentLeak: A full-stack benchmark for privacy leakage in multi-agent LLM systems](#). *arXiv preprint arXiv:2602.11510*.
- Yanai Elazar and Yoav Goldberg. 2018. [Adversarial removal of demographic attributes from text data](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, Brussels, Belgium. Association for Computational Linguistics.
- Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuan-Jing Huang. 2022. [Cqg: A simple and effective controlled generation framework for multi-hop question generation](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906.
- Ziyuan Gao, Di Liang, Xianjie Wu, Philippe Morel, and Minlong Peng. 2026. [Decorl: Decoupling reasoning chains via parallel sub-step generation and cascaded reinforcement for interpretable and scalable rlhf](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30789–30797.
- Tao Gui, Qi Zhang, Jingjing Gong, Minlong Peng, Di Liang, Keyu Ding, and Xuan-Jing Huang. 2018. [Transferring from formal newswire domain with hypernet for twitter pos tagging](#). In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2540–2549.
- Xiaosong Han, Ke Chen, Xindi Dai, Di Liang, Minlong Peng, Wei Pang, Fausto Giunchiglia, Xiaoyue Feng, Yonghao Liu, and Renchu Guan. 2026. [Trace: Discovering task-specific parameter via adaptation-aware probing for continual fine-tuning](#). In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 1530–1541.
- Pengfei He, Yuping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. 2025. [Red-teaming LLM multi-agent systems via communication attacks](#). In *Findings of*

- the Association for Computational Linguistics: ACL 2025, pages 6726–6747, Vienna, Austria. Association for Computational Linguistics.
- Mengze Hong, Xia Zeng, Zeyang Lei, Sheng Wang, Chen Jason Zhang, Di Jiang, Taiming Fu, Jinfeng Huang, Mengqiao Liu, Qinghe Chang, and 1 others. 2026. Uxbench: Benchmarking user experience in ai assistants. *arXiv preprint arXiv:2606.09570*.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. [MetaGPT: Meta programming for a multi-agent collaborative framework](#). In *Proceedings of the Twelfth International Conference on Learning Representations*.
- Zhanghao Hu, Qinglin Zhu, Runcong Zhao, Di Liang, Hanqi Yan, Yulan He, and Lin Gui. 2026. Beyond rag for agent memory: Retrieval by decoupling and aggregation. *arXiv preprint arXiv:2602.02007*.
- Boris Kriuk and Logic Ng. 2025. [Q-KVComm: Efficient multi-agent communication via adaptive KV cache compression](#). *arXiv preprint arXiv:2512.17914*.
- Bo Li, Di Liang, and Zixin Zhang. 2024a. Comate-former: Combined attention transformer for semantic sentence matching. In *ECAI 2024: 27th European Conference on Artificial Intelligence, 19–24 October 2024, Santiago de Compostela, Spain—including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)*, pages 3677–3684. SAGE Publications Pvt. Ltd 1 Oliver’s Yard, 55 City Road, London, EC1Y 1SP.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023a. [CAMEL: Communicative agents for “mind” exploration of large language model society](#). In *Advances in Neural Information Processing Systems*, volume 36.
- Haoran Li, Mingshi Xu, and Yangqiu Song. 2023b. [Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14022–14040, Toronto, Canada. Association for Computational Linguistics.
- Junchen Li, Chao Qi, Rongzheng Wang, Qizhi Chen, Liang Xu, Di Liang, Bob Simons, and Shuang Liang. 2026. When safety becomes a vulnerability: Exploiting llm alignment homogeneity for transferable blocking in rag. *arXiv preprint arXiv:2603.03919*.
- Liang Li, Qisheng Liao, Meiting Lai, Di Liang, and Shangsong Liang. 2024b. Local and global: Text matching via syntax graph calibration. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11571–11575. IEEE.
- Wenkai Li, Liwen Sun, Zhenxiang Guan, Xuhui Zhou, and Maarten Sap. 2025. [1-2-3 check: Enhancing contextual privacy in LLM via multi-agent reasoning](#). In *Proceedings of the The First Workshop on LLM Security (LLMSEC)*, pages 115–128, Vienna, Austria. Association for Computational Linguistics.
- Di Liang, Fubao Zhang, Qi Zhang, and Xuan-Jing Huang. 2019a. Asynchronous deep interaction network for natural language inference. 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 2692–2700.
- Di Liang, Fubao Zhang, Weidong Zhang, Qi Zhang, Jinlan Fu, Minlong Peng, Tao Gui, and Xuanjing Huang. 2019b. Adaptive multi-attention network incorporating answer information for duplicate question detection. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 95–104.
- Zekai Lin, Chao Xue, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Lei Jiang, Yu Lu, Bob Simons, Shuang Liang, and 1 others. 2026. Parameter importance is not static: Evolving parameter isolation for supervised fine-tuning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32846–32864.
- Jinbo Liu, Defu Cao, Yifei Wei, Tianyao Su, Yuan Liang, Yushun Dong, Yan Liu, Yue Zhao, and Xiyang Hu. 2025a. [Topology matters: Measuring memory leakage in multi-agent LLMs](#). *arXiv preprint arXiv:2512.04668*.
- Peiyang Liu, Zhirui Chen, Xi Wang, Di Liang, Youru Li, Zhi Cai, and Wei Ye. 2026a. Learning from contrasts: Synthesizing reasoning paths from diverse search trajectories. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10947–10969.
- Peiyang Liu, Ziqiang Cui, Di Liang, and Wei Ye. 2025b. Who stole your data? a method for detecting unauthorized rag theft. *arXiv preprint arXiv:2510.07728*.
- Peiyang Liu, Ziqiang Cui, Xi Wang, Di Liang, and Wei Ye. 2026b. Chain of evidence: Pixel-level visual attribution for iterative retrieval-augmented generation. *arXiv preprint arXiv:2605.01284*.
- Peiyang Liu, Qiang Yan, Ziqiang Cui, Di Liang, Xi Wang, and Wei Ye. 2026c. Beyond semantic relevance: Counterfactual risk minimization for robust retrieval-augmented generation. *arXiv preprint arXiv:2605.01302*.
- Xiaoyu Liu, Xiaoyu Guan, Di Liang, and Xianjie Wu. 2026d. Dpi: Exploiting parameter heterogeneity for interference-free fine-tuning. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 19337–19341. IEEE.

- Xiaoyu Liu, Di Liang, Hongyu Shan, Peiyang Liu, Yonghao Liu, Muling Wu, Yuntao Li, Xianjie Wu, Li Miao, Jiangrong Shen, and 1 others. 2025c. Structural reward model: Enhancing interpretability, efficiency, and scalability in reward modeling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 672–685.
- Yonghao Liu, Mengyu Li, Di Liang, Ximing Li, Fausto Giunchiglia, Lan Huang, Xiaoyue Feng, and Renchu Guan. 2024. Resolving word vagueness with scenario-guided adapter for natural language inference. *arXiv preprint arXiv:2405.12434*.
- Yonghao Liu, Di Liang, Fang Fang, Sirui Wang, Wei Wu, and Rui Jiang. 2023a. Time-aware multiway adaptive fusion network for temporal knowledge graph question answering. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Yonghao Liu, Di Liang, Li Mengyu, Fausto Giunchiglia, Li Ximing, Sirui Wang, Wu Wei, Lan Huang, Xiaoyue Feng, Renchu Guan, and 1 others. 2023b. Local and global: Temporal question answering via information fusion. In *IJCAI*, volume 2023, pages 5141–5149. International Joint Conferences on Artificial Intelligence.
- Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. [Learning sparse neural networks through \$L_0\$ regularization](#). In *Proceedings of the International Conference on Learning Representations*.
- Ruotian Ma, Yiding Tan, Xin Zhou, Xuanning Chen, Di Liang, Sirui Wang, Wei Wu, Tao Gui, and Qi Zhang. 2022. Searching for optimal subword tokenization in cross-domain ner. *arXiv preprint arXiv:2206.03352*.
- John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. 2023. [Text embeddings reveal \(almost\) as much as text](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12448–12460, Singapore. Association for Computational Linguistics.
- Vaidehi Patil, Elias Stengel-Eskin, and Mohit Bansal. 2025. [The sum leaks more than its parts: Compositional privacy risks and mitigations in multi-agent collaboration](#). *arXiv preprint arXiv:2509.14284*.
- Chau Pham, Boyi Liu, Yingxiang Yang, Zhengyu Chen, Tianyi Liu, Jianbo Yuan, Bryan A. Plummer, Zhaoran Wang, and Hongxia Yang. 2024. [Let models speak ciphers: Multiagent debate through embeddings](#). In *Proceedings of the Twelfth International Conference on Learning Representations*.
- Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. [ChatDev: Communicative agents for software development](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, Bangkok, Thailand. Association for Computational Linguistics.
- Qi Qian, Muling Wu, Zisu Huang, Wenhao Liu, Changze Lv, Xiaohua Wang, Zhenghua Wang, Zhengkang Guo, Zhibo Xu, Lina Chen, and 1 others. Adaptive curriculum strategies: Stabilizing reinforcement learning for large language models.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. [ToolLLM: Facilitating large language models to master 16000+ real-world apis](#). In *Proceedings of the Twelfth International Conference on Learning Representations*.
- Vignav Ramesh and Kenneth Li. 2025. [Communicating activations between language model agents](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 51094–51116. PMLR.
- Hongyu Shan, Mingyang Song, Chang Dai, Di Liang, and Han Chen. 2025. R-capsule: Compressing high-level plans for efficient large language model reasoning. *arXiv preprint arXiv:2509.22131*.
- Xiangyu Shi, Marco Chiesa, Gerald Q. Maguire, Jr., and Dejan Kostic. 2026. [KVComm: Enabling efficient LLM communication through selective KV sharing](#). In *Proceedings of the Fourteenth International Conference on Learning Representations*.
- Jian Song, Di Liang, Rumei Li, Yuntao Li, Sirui Wang, Minlong Peng, Wei Wu, and Yongxin Yu. 2022. Improving semantic matching through dependency-enhanced pre-trained model with adaptive fusion. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 45–57.
- Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. 2016. [Learning multiagent communication with backpropagation](#). In *Advances in Neural Information Processing Systems*, volume 29, pages 2244–2252.
- Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. [ProofWriter: Generating implications, proofs, and abductive statements over natural language](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.
- Naftali Tishby, Fernando C. Pereira, and William Bialek. 2000. [The information bottleneck method](#). *arXiv preprint physics/0004057*. Originally presented at the 37th Annual Allerton Conference on Communication, Control, and Computing, 1999.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. [MuSiQue: Multi-hop questions via single-hop question composition](#). *Transactions of the Association for Computational Linguistics*, 10:539–554.

- Bo Wang, Weiye He, Shenglai Zeng, Zhen Xiang, Yue Xing, Jiliang Tang, and Pengfei He. 2025a. [Unveiling privacy risks in LLM agent memory](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25241–25260, Vienna, Austria. Association for Computational Linguistics.
- Rongzheng Wang, Yihong Huang, Muquan Li, Jiakai Li, Di Liang, Bob Simons, Pei Ke, Shuang Liang, and Ke Qin. 2026. Rethinking llm-driven heuristic design: Generating efficient and specialized solvers via dynamics-aware optimization. *arXiv preprint arXiv:2601.20868*.
- Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chongye Guo, Kun Wang, and Yang Wang. 2025b. G-Safeguard: A topology-guided security lens and treatment on LLM-based multi-agent systems. In *ACL2025*.
- Shouju Wang, Fenglin Yu, Xirui Liu, Xiaoting Qin, Jue Zhang, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. 2025c. Privacy in action: Towards realistic privacy mitigation and evaluation for LLM-powered agents. In *EMNLP 2025*.
- Sirui Wang, Di Liang, Jian Song, Yuntao Li, and Wei Wu. 2022. Dabert: Dual attention enhanced bert for semantic matching. In *Proceedings of the 29th international conference on computational linguistics*, pages 1645–1654.
- Yao Wang, Di Liang, and Minlong Peng. 2025d. Not all parameters are created equal: Smart isolation boosts fine-tuning performance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9882–9896.
- Muling Wu, Qi Qian, Wenhao Liu, Xiaohua Wang, Zisu Huang, Di Liang, LI Miao, Shihan Dou, Changze Lv, Zhenghua Wang, and 1 others. 2025a. Progressive mastery: Customized curriculum learning with guided prompting for mathematical reasoning. *arXiv preprint arXiv:2506.04065*.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2024. [AutoGen: Enabling next-gen LLM applications via multi-agent conversations](#). In *Proceedings of the First Conference on Language Modeling*.
- Xianjie Wu, Di Liang, Jian Yang, Xianfu Cheng, LinZheng Chai, Tongliang Li, Liqun Yang, and Zhoujun Li. 2025b. Breaking size barrier: Enhancing reasoning for large-size table question answering. In *International Conference on Database Systems for Advanced Applications*, pages 241–256. Springer.
- Xianjie Wu, Xiaohang Xu, Tingyu Jiang, Jian Yang, Di Liang, Xianfu Cheng, Zhenhe Wu, Linzheng Chai, Wei Zhang, Jiaheng Liu, and 1 others. 2026. Mmtablebench: A multi-level multimodal benchmark for reasoning and layout complexity in table qa. In *Proceedings of the ACM Web Conference 2026*, pages 3881–3892.
- Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, and 1 others. 2025c. Tablebench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506.
- Xianjie Wu, Jian Yang, Tongliang Li, Shiwei Zhang, Yiyang Du, LinZheng Chai, Di Liang, and Zhoujun Li. 2025d. Unleashing potential of evidence in knowledge-intensive dialogue generation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Chao Xue, Di Liang, Pengfei Wang, and Jing Zhang. 2024. Question calibration and multi-hop modeling for temporal question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19332–19340.
- Chao Xue, Di Liang, Sirui Wang, Jing Zhang, and Wei Wu. 2023. Dual path modeling for semantic matching by perceiving subtle conflicts. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, and 1 others. 2026a. Reason only when needed: Efficient generative reward modeling via model-internal uncertainty. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 23302–23319.
- Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, and 1 others. 2026b. Why supervised fine-tuning fails to learn: A systematic study of incomplete learning in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30186–30213.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Ye Yu, Heming Liu, Haibo Jin, Xiaopeng Yuan, Peng Kuang, and Haohan Wang. 2026. [Learning to communicate: Toward end-to-end optimization of multi-agent language systems](#). *arXiv preprint arXiv:2604.21794*.
- Ziqian Zeng, Jianwei Wang, Junyao Yang, Zhengdong Lu, Haoran Li, Huiping Zhuang, and Cen Chen. 2025.

[PrivacyRestore: Privacy-preserving inference in large language models via privacy removal and restoration](#). In *ACL*, pages 10821–10855, Vienna, Austria. Association for Computational Linguistics.

Arman Zharmagambetov, Chuan Guo, Ivan Evtimov, Maya Pavlova, Ruslan Salakhutdinov, and Kamalika Chaudhuri. 2025. [AgentDAM: Privacy leakage evaluation for autonomous web agents](#). In *NeurIPS*.

Rui Zheng, Bao Rong, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuan-Jing Huang. 2022. Robust lottery tickets for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2211–2224.

Yujia Zheng, Zhuokai Zhao, Zijian Li, Yaqi Xie, Mingze Gao, Lizhu Zhang, and Kun Zhang. 2025. [Thought communication in multiagent collaboration](#). *arXiv preprint arXiv:2510.20733*. NeurIPS 2025 Spotlight.

Jiaru Zou, Xiyuan Yang, Ruizhong Qiu, Gaotang Li, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. 2025. [Latent collaboration in multi-agent systems](#). *arXiv preprint arXiv:2511.20639*.

A Extended Related Work

This appendix provides an extended discussion of related work. The main paper focuses on the problem formulation and empirical findings; here we give a broader taxonomy of prior work and clarify the position of ICLC.

A.1 Multi-Agent Language Model Systems

Large language models are increasingly organized into multi-agent systems, where multiple role-specialized agents coordinate through explicit communication, tool use, verification, planning, or memory sharing. Representative systems such as AutoGen, MetaGPT, and ChatDev demonstrate that agent collaboration can improve problem solving by decomposing complex tasks into structured interaction protocols (Li et al., 2023a; Chen et al., 2024; Wu et al., 2024; Hong et al., 2024; Qian et al.; Xue et al., 2026b; Qian et al., 2024). Most of these systems use natural language as the communication medium, which makes messages human-inspectable and compatible with standard prompting, logging, and moderation interfaces. However, text communication also serializes high-dimensional internal model states into token sequences, creating a bandwidth bottleneck and potentially discarding information useful for collaboration. ICLC studies the privacy implications of moving beyond this text boundary.

A.2 Latent Communication and Continuous Inter-Agent Reasoning

A growing line of work explores communication beyond natural language. CIPHER lets models communicate through raw transformer output embeddings rather than sampled tokens, enabling multi-agent debate through embedding-level signals (Pham et al., 2024). Communicating Activations Between Language Model Agents studies direct activation-level communication by combining intermediate activations from one language model into another model’s forward pass (Ramesh and Li, 2025). Other recent work explores inter-agent communication entirely in latent space, including latent thoughts, latent working memory, and private/shared latent variables in multi-agent collaboration (Du et al., 2025; Zou et al., 2025; Liu et al., 2026c; Wu et al., 2026; Liu et al., 2026d; Zheng et al., 2025; Hong et al., 2026; Yu et al., 2026).

These works establish that non-textual representations can serve as effective communication media. However, their primary focus is utility, efficiency, or feasibility. ICLC instead asks what private information becomes exposed when such continuous internal representations cross agent boundaries, and how this exposure can be constrained.

A.3 KV-Cache and State-Sharing Communication

Another closely related direction uses transformer cache states as communication artifacts. KVComm enables efficient communication between LLMs through selective sharing of key-value pairs, using attention-based selection to identify informative cache entries (Shi et al., 2026; Li et al., 2024a; Liu et al., 2024; Li et al., 2024b; Gui et al., 2018; Liu et al., 2026b; Xue et al., 2024; Kriuk and Ng, 2025). Such cache-derived communication is attractive because KV states preserve rich contextual and reasoning information, but this also makes them privacy-sensitive. LCGuard directly studies safe KV sharing in multi-agent systems and introduces adversarial transformations to reduce reconstruction-based leakage from shared KV artifacts (Asif et al., 2026; Liu et al., 2023a; Xue et al., 2023; Liu et al., 2023b).

ICLC differs from KV-specific defenses in two ways. First, our communicated latent message is defined as a general edge-level object $z_{i \rightarrow j}^{(t)}$, which may be instantiated by hidden-state-derived slots, activation-derived messages, soft prefixes, or cache-

derived representations. Second, our leakage formulation explicitly separates instantaneous leakage from trajectory-level propagation leakage over multi-round transcripts. This lets ICLC control not only reconstruction from a single shared representation, but also the accumulation of sensitive signals across the communication graph.

A.4 Privacy Leakage from Dense Representations

Dense representations should not be treated as private merely because they are not human-readable. Embedding inversion studies show that text embeddings can reveal substantial information about their original inputs. Vec2Text demonstrates that dense text embeddings can be inverted to recover much of the original text, including sensitive personal information (Morris et al., 2023; Gao et al., 2026; Liang et al., 2019b; Li et al., 2026). GEIA similarly shows that sentence embeddings can be used to reconstruct coherent input sequences through generative inversion (Li et al., 2023b). Recent internal-state inversion work further suggests that deep internal states of LLMs may still preserve recoverable prompt information (Dong et al., 2025).

These results motivate our threat model: communicated latent messages may encode sensitive attributes, confidential spans, membership signals, or memorized content even when no natural-language disclosure occurs. ICLC therefore evaluates leakage with held-out attackers rather than assuming that vector opacity implies privacy.

A.5 Privacy and Security in Multi-Agent LLMs

Recent work has begun to study privacy and security risks specific to LLM agents. AgentLeak argues that privacy leakage in multi-agent LLM systems can be missed by output-only auditing, because sensitive information may flow through internal channels such as inter-agent messages, shared memory, and tool arguments (El Yagoubi et al., 2026; Wang et al., 2026; Liu et al., 2025c; Wu et al., 2025c; Wang et al., 2022; Song et al., 2022; Wang et al., 2025d). Other work studies privacy risks in agent memory, autonomous web agents, and communication attacks against multi-agent systems (Wang et al., 2025a; He et al., 2025; Wang et al., 2025b; Zharmagambetov et al., 2025; Li et al., 2025; Liang et al., 2019a; Wu et al., 2025d; Hu et al., 2026; Wang et al., 2025c; Zeng et al., 2025; Liu et al., 2025a; Wu et al., 2025b; Patil

et al., 2025). These studies support a broader view that privacy should be evaluated inside the agent pipeline, not only in final responses.

ICLC is complementary to these benchmarks and audits. Rather than measuring text-level or action-level leakage alone, we focus on latent communication channels. Our method treats the communication boundary itself as the locus of privacy control.

A.6 Information Bottlenecks, Adversarial Removal, and Sparse Communication

ICLC is also related to information bottleneck methods, adversarial representation learning, and sparse communication (Tishby et al., 2000; Alemi et al., 2017; Elazar and Goldberg, 2018; Zheng et al., 2022; Louizos et al., 2018; Fei et al., 2022; Liu et al., 2025b; Ma et al., 2022; Shan et al., 2025; Dai et al., 2025; Wu et al., 2025a; Sukhbaatar et al., 2016; Das et al., 2019). Information bottlenecks constrain representation capacity and have been used to encourage task-relevant compression. Adversarial representation learning attempts to reduce the recoverability of protected attributes, but prior work has cautioned that adversarial removal does not provide formal privacy guarantees and should be evaluated with independent attackers. Sparse communication and learned topology methods reduce unnecessary message passing in multi-agent systems.

ICLC combines these ideas at the latent communication boundary. The bottleneck limits the raw capacity of each edge-level channel, the leakage critics test whether sensitive information remains recoverable, and the topology-aware gate reduces the exposure surface over the communication graph. The key distinction is that ICLC treats privacy leakage as a dynamic information-flow problem rather than a static representation-cleaning problem.

B Implementation Details of ICLC

B.1 Latent Substrate Extraction

In the main paper, ICLC extracts a local substrate

$$u_i^{(t)} = \text{Extract}_\rho(H_{i,\mathcal{L}}^{(t)}, q) \in \mathbb{R}^{K \times d_c}.$$

Here $H_{i,\mathcal{L}}^{(t)}$ denotes hidden states from selected transformer layers, K is the number of latent slots, and d_c is the communication dimension. The extractor can be instantiated with last-token pooling, mean pooling, learned query pooling, or attention

Benchmark	Train	Valid.	Test	Protected Fields
HotpotQA	4,000	500	1,000	Synthetic canaries, attributes, confidential spans
GSM8K	5,000	500	1,000	Task-irrelevant user attributes and distractors
MuSiQue	4,000	500	1,000	Protected records and entity-level distractors
ProofWriter	3,200	400	800	Private rules and irrelevant protected facts
ToolBench	3,000	500	800	Mock API keys, preferences, confidential constraints

Table 3: **Reference dataset statistics.** Protected fields are injected into local agent contexts but are not needed for the public answer or final plan.

pooling. Unless otherwise stated, we use learned query pooling followed by a linear projection to the shared communication dimension.

A typical implementation is:

$$\text{Extract}_\rho(H) = \text{AttnPool}(Q_\rho, H)W_\rho,$$

where $Q_\rho \in \mathbb{R}^{K \times d_q}$ are learned slot queries and $W_\rho \in \mathbb{R}^{d_h \times d_c}$ projects pooled states into the communication space. The extracted substrate is local to the sender and is not transmitted directly. This distinction is important: ICLC controls the boundary map from $u_i^{(t)}$ to $z_{i \rightarrow j}^{(t)}$, rather than exposing the full internal state.

B.2 Task-Conditioned Bottleneck

For each candidate communication edge $i \rightarrow j$, ICLC uses a task-conditioned bottleneck encoder to produce a stochastic latent message. Given the sender state $u_i^{(t)}$, task instruction q , sender embedding e_i , receiver embedding e_j , and round index t , the encoder parameterizes a diagonal Gaussian posterior:

$$q_{ij}^{(t)}(z) = \mathcal{N}\left(z; \mu_{ij}^{(t)}, \text{diag}\left((\sigma_{ij}^{(t)})^2\right)\right). \quad (22)$$

Equivalently, the posterior parameters are produced by

$$(\mu_{ij}^{(t)}, \sigma_{ij}^{(t)}) = q_\phi\left(u_i^{(t)}, q, e_i, e_j, t\right).$$

This conditioning makes the transmitted representation depend not only on the sender state, but also on the intended receiver and the current task.

During training, we sample a candidate message with the reparameterization trick:

$$\tilde{z}_{i \rightarrow j}^{(t)} = \mu_{ij}^{(t)} + \sigma_{ij}^{(t)} \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \quad (23)$$

where $\odot$ denotes element-wise multiplication. At inference time, we use the posterior mean $\mu_{ij}^{(t)}$ as the deterministic message.

The capacity regularizer constrains the amount of information carried by each candidate message:

$$\mathcal{L}_{\text{cap}} = \mathbb{E}_{t, (i, j)} \left[D_{\text{KL}}\left(q_{ij}^{(t)} \| p_j\right) \right], \quad (24)$$

where the prior is conditioned on the receiver and the task:

$$p_j(z) = p_\eta(z | q, e_j). \quad (25)$$

When no learned prior is used, p_j reduces to the standard Gaussian $\mathcal{N}(0, I)$.

For diagonal Gaussians $q = \mathcal{N}(\mu_q, \text{diag}(\sigma_q^2))$ and $p = \mathcal{N}(\mu_p, \text{diag}(\sigma_p^2))$, the KL divergence can be written in a compact vector form:

$$D_{\text{KL}}(q \| p) = \frac{1}{2} \left(\|(\mu_q - \mu_p) \odot \sigma_p\|_2^2 + \|\sigma_q \odot \sigma_p\|_2^2 - d_z + 2 \sum_{k=1}^{d_z} \log \frac{\sigma_{p,k}}{\sigma_{q,k}} \right), \quad (26)$$

where d_z is the latent dimensionality and $\odot$ denotes element-wise division. This form is algebraically equivalent to the usual coordinate-wise KL expression, but avoids an overly wide display equation.

The receiver-conditioned prior is useful because different receivers may require different task-relevant information. A verifier may only need high-level consistency signals, whereas an executor may need concrete constraints such as tool arguments, action preconditions, or output-format requirements. Conditioning the prior on e_j therefore encourages the bottleneck to preserve information useful for the target receiver while suppressing unnecessary latent details.

B.3 HardConcrete Exposure Gate

ICLC uses a differentiable hard-concrete gate to determine which slots are exposed on each edge. Given a gate logit $\pi_{ij}^{(t)}$, we sample $u \sim \text{Uniform}(0, 1)$ and compute:

$$s = \sigma \left(\frac{\log u - \log(1 - u) + \pi_{ij}^{(t)}}{\tau_g} \right),$$

where τ_g is the gate temperature. Following the hard-concrete relaxation, the stretched and clipped gate is:

$$\bar{s} = s(r - l) + l, \quad g_{ij}^{(t)} = \min(1, \max(0, \bar{s})),$$

where $l < 0$ and $r > 1$ are the lower and upper stretch constants. We use slot-wise gates by default, so $g_{ij}^{(t)} \in [0, 1]^K$. The flow regularizer approximates the expected number of active latent slots:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t,(i,j)} \left[\|g_{ij}^{(t)}\|_0 \right].$$

At inference time, we use a deterministic threshold:

$$\hat{g}_{ij}^{(t)} = \mathbb{I}[\sigma(\pi_{ij}^{(t)}) > \delta_g].$$

If all slots on an edge are inactive, no communicated latent artifact is produced for that edge.

B.4 Receiver-Side Latent Fusion

The receiver aggregates active incoming messages and incorporates them through a latent interface:

$$\tilde{H}_j^{(t)} = \text{Fuse}_\tau \left(H_j^{(t)}, \text{Agg}_\tau(Z_j^{(t)}) \right).$$

We consider three fusion variants:

- **Soft-prefix fusion:** incoming messages are projected into prefix-like continuous embeddings and prepended to the receiver computation.
- **Cross-attention fusion:** receiver hidden states attend over incoming latent messages.
- **Adapter fusion:** messages are projected through a small adapter and added to selected hidden layers.

Unless otherwise specified, we use soft-prefix fusion because it is simple, model-agnostic, and does not require modifying the backbone attention implementation. All backbone LLM parameters remain frozen.

C Threat Models and Leakage Metrics

C.1 Attacker Observation Levels

We evaluate two main attacker observation levels. A **single-message attacker** observes one edge-level communicated latent message:

$$z_{i \rightarrow j}^{(t)}.$$

A **trajectory attacker** observes the transcript available to a curious receiver or coalition:

$$\mathcal{O}_C^{1:T} = \bigcup_{t=1}^T \{z_{i \rightarrow j}^{(t)} : (i, j) \in \mathcal{E}^{(t)}, j \in \mathcal{C}\}.$$

This is the main transcript-level threat model used in the paper. We do not assume access to downstream hidden states in the main evaluation.

For completeness, one can also evaluate a stronger **compromised-receiver hidden-state attacker** that observes downstream receiver states. This setting is stricter than our main threat model and should be interpreted as a stress test rather than the default privacy definition.

C.2 Leakage Score Normalization

All privacy results are reported as lower-is-better leakage scores normalized to $[0, 1]$. For attribute inference, leakage is computed from held-out attack performance and normalized by the task-specific chance baseline. For binary private-context membership inference, we report attack advantage rather than raw accuracy:

$$\text{Adv} = 2 |\text{AUC} - 0.5|.$$

This avoids ambiguity when raw accuracy falls below chance because of label inversion. For secret reconstruction, we report normalized reconstruction success or token-level recovery. The average leakage score in the main tables is the arithmetic mean of attribute, membership, and reconstruction leakage.

C.3 Attack Architectures

We use held-out attackers for final privacy evaluation. For attribute inference, the attacker is an MLP classifier or regressor over the observed latent message or transcript embedding. For private-context membership inference, the attacker is a binary classifier trained to predict whether a protected record or span appears in the sender’s private context. For secret reconstruction, the attacker is an autoregressive decoder trained to reconstruct a confidential span:

$$\mathcal{R}_{\text{sec}} = \mathbb{E} \left[\sum_{\ell=1}^{|s_i^{\text{sec}}|} \log p_\psi \left(s_{i,\ell}^{\text{sec}} \mid s_{i,<\ell}^{\text{sec}}, \xi_i \right) \right],$$

where ξ_i is either a single-message observation or a trajectory observation. All final attackers are trained after the communication modules are fixed.

C.4 Held-Out Attacker Protocol

The leakage critics used during ICLC training are not reused for final privacy evaluation. After training the communication modules, we freeze all ICLC parameters and train fresh held-out attackers on a separate split. This prevents the reported leakage reduction from merely reflecting overfitting to

weak training-time critics. All reported privacy results use these held-out attackers unless otherwise specified.

D Task Construction and Dataset Details

D.1 Dataset Statistics

Table 3 summarizes the reference dataset sizes and task construction. The protected fields are inserted into agent-local contexts and are not required for the public task answer. We use these fields only to evaluate whether communicated latent messages expose task-irrelevant private content.

D.2 Private-Context QA

Private-context QA is built on HotpotQA (Yang et al., 2018). For each question, supporting paragraphs are distributed across agents. Synthetic private attributes, canaries, or confidential spans are inserted into local contexts. The public question is constructed so that these protected fields are not required for the final answer. This setup tests whether agents can collaborate without unnecessarily exposing private local context.

D.3 Collaborative Reasoning

Collaborative reasoning uses GSM8K (Cobbe et al., 2021), MuSiQue (Trivedi et al., 2022), and ProofWriter (Tafjord et al., 2021). Agents receive partial observations, intermediate reasoning roles, or complementary subproblems. Sensitive distractors such as user attributes or protected records are inserted into agent-local context. These protected fields are task-irrelevant and are used only to measure leakage.

D.4 Tool-Use Planning

Tool-use planning uses ToolBench (Qin et al., 2024). Agents are assigned roles such as planner, executor, verifier, and memory manager. Protected fields include mock API keys, private preferences, and confidential constraints. The final plan can be evaluated without revealing these protected fields to all agents.

D.5 Prompt Templates

We use fixed prompts for each agent role and each task family. The prompt templates specify the public task objective, the agent’s local context, and the allowed communication interface. Text-MAS agents output natural-language messages. Latent communication agents do not output natural-

language inter-agent messages; instead, communicated latent messages are generated by the communication module.

Planner prompt.

You are the planner agent. Use your local context and messages from other agents to propose the next step. Do not reveal private fields unless they are necessary for solving the public task.

Verifier prompt.

You are the verifier agent. Check whether the proposed answer or plan satisfies the public task. Use incoming information only for verification.

Memory agent prompt.

You manage local memory. Retrieve only task-relevant information. Protected records should not be exposed to other agents unless explicitly required.

Executor prompt.

You are the executor agent. Convert the approved plan into tool calls or intermediate actions. Use only task-relevant constraints and avoid exposing private fields to other agents.

E Baselines and Communication Settings

E.1 Text and Raw Latent Communication

Text-MAS uses natural-language messages as the communication medium. Messages are logged and passed to receiver agents as text. The token budget is kept fixed across methods that use text communication.

Text-Redaction-MAS applies a text-side redaction filter before message delivery. The filter removes spans classified as private attributes, canaries, credentials, or confidential constraints. This baseline is expected to reduce leakage, but it can also remove useful context and reduce task utility.

Raw-Latent-MAS directly transmits hidden-state-derived representations without privacy control. This baseline corresponds to exposing a raw latent artifact or its projection across the communication edge. It is used only as an unprotected baseline, not as the ICLC message definition.

E.2 Noise and Projection Baselines

Noise-Latent-MAS adds Gaussian noise to transmitted latent messages:

$$z' = z + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_n^2 I).$$

Calibrated-Noise-MAS tunes the noise scale σ_n on the validation set using the same privacy–utility selection rule as ICLC.

Projection-MAS applies a fixed low-dimensional projection before transmission. The projection is random and is kept fixed after initialization.

Learned-Projection-MAS learns a task-preserving projection adapter on the validation-selected training split. It is stronger than fixed projection because it can preserve task-useful directions while reducing some irrelevant dimensions.

Adv-Sanitize-MAS trains a message-level adversarial sanitizer against single-message sensitive-attribute recovery. It does not use trajectory-level critics or topology-aware exposure control.

E.3 ICLC Ablations

We evaluate three main ablations:

- **w/o Bottleneck:** removes the KL capacity regularizer and directly uses deterministic projected messages.
- **w/o Adv. Critic:** removes leakage critics and trains only with task, capacity, and flow losses.
- **w/o Topology Gate:** keeps the bottleneck and adversarial regularization, but exposes all candidate edges.

The row **w/o All Components** corresponds to Raw-Latent-MAS.

E.4 Communication Topologies

We evaluate three communication topologies:

- **Fully connected:** every agent may communicate with every other agent.
- **Chain:** information propagates sequentially through agents.
- **Star:** a central coordinator aggregates and redistributes messages.

Fully connected graphs create the largest exposure surface, while chain graphs create fewer propagation paths.

E.5 Baseline Tuning Protocol

All baselines and ICLC variants are selected on a validation split before test evaluation. For each method, we sweep the method-specific control parameters available to that method. The same validation privacy–utility selection criterion is applied to all methods. No test-set privacy or utility results are used for hyperparameter selection.

Hyperparameter	Value
Number of agents N	4
Default communication rounds T	3
Round-scaling maximum T	8
Latent slots K	8
Communication dimension d_c	256
Selected layers $\mathcal{L}$	middle + final
Bottleneck weight β	0.10
Flow weight α	0.05
Leakage weight λ	0.50
Gate threshold δ_g	0.50
HardConcrete temperature τ_g	0.67
Critic update steps	2
Communication LR	1×10^{-4}
Critic LR	2×10^{-4}
Batch size	16
Optimizer	AdamW
Random seeds	3

Table 4: **Hyperparameters used for ICLC training and inference.** Unless otherwise stated, all experiments use this configuration.

F Training Details and Hyperparameters

F.1 Optimization Procedure

ICLC is trained with alternating updates. At each iteration, leakage critics are first updated to maximize sensitive target recoverability. The communication module is then updated to minimize the task loss, capacity loss, exposure loss, and adversarial leakage objective. All backbone LLM parameters remain frozen throughout training.

F.2 Hyperparameters

Table 4 reports the reference hyperparameters used in our experiments. Unless otherwise stated, the same configuration is used across task families and baselines are tuned using the validation protocol described in Appendix E.5.

F.3 Compute and Parameter Count

We freeze the backbone LLMs and train only lightweight communication modules and leakage critics. The inference-time overhead comes from the extractor, bottleneck mean network, topology gate, and receiver-side fusion module. Leakage critics are discarded at inference time. Table 5 reports the parameter count and compute budget for the reference setting.

G Additional Experimental Results

This section provides additional experimental evidence supporting the main privacy–utility compari-

Item	Reference Value
Trainable communication parameters	18.6M
Trainable critic parameters	24.1M
Inference-time added parameters	18.6M
Backbone parameters updated	0
Training GPUs	$4 \times$ A100 80GB
Training time per seed	8.5 hours
Inference latency overhead	+6.8%

Table 5: **Compute and parameter count.** Leakage critics are used only during training and are removed at inference time.

son and the ablation conclusions. We organize the results into five parts: full per-benchmark utility, full held-out attack results, detailed component ablation, topology and round-scaling analysis, and communication efficiency. All results use the same task splits, backbone agents, communication protocol, and held-out attacker evaluation described in Section 4.

G.1 Full Per-Benchmark Utility Results

Table 6 reports the full per-benchmark utility breakdown. The reasoning score in Table 1 is the arithmetic mean of GSM8K, MuSiQue, and ProofWriter. The average utility in Table 1 is the arithmetic mean of QA, Reason., and Plan. The numbers therefore align directly with the main results.

ICLC obtains the strongest utility on all five benchmarks while retaining the efficiency and coordination benefits of latent communication. The comparison with Text-Redaction-MAS shows that aggressive text-side filtering can reduce leakage but also removes information useful for coordination. The comparison with Raw-Latent-MAS shows that privacy control does not require giving up the utility advantage of latent exchange.

G.2 Full Held-Out Attack Results

Table 7 reports leakage under held-out attackers trained after the communication modules are fixed. The single-message averages match the privacy columns in Table 1. Trajectory leakage is evaluated on $T = 8$ multi-round transcripts and corresponds to the system-level threat model used in this paper: curious receivers or coalitions observe communicated latent artifacts over multiple rounds, but not hidden states internal to downstream agents.

The gap between single-message and trajectory leakage is largest for Raw-Latent-MAS, whose av-

erage leakage increases from 0.747 to 0.910 when the attacker observes a multi-round transcript. This confirms that latent privacy risk is amplified by interaction history. Text-Redaction-MAS achieves the lowest leakage overall, but Table 6 shows that this comes with a substantial utility loss. Among latent-communication methods, ICLC reduces both forms of leakage most effectively, lowering average single-message leakage to 0.398 and $T = 8$ trajectory leakage to 0.350.

G.3 Detailed Component Ablation

Table 8 provides the full component ablation underlying Table 2. The task utility columns show that removing individual privacy controls does not substantially change task performance. The leakage columns show that the components target different failure modes: the bottleneck and adversarial critic are most important for suppressing single-message recoverability, while the topology gate is most important for suppressing propagation over long interaction horizons.

G.4 Topology and Round-Scaling Results

Table 9 evaluates $T = 8$ trajectory leakage under different communication topologies. Fully connected graphs expose the largest number of sender-receiver paths and therefore produce the highest leakage for all latent methods. Chain graphs reduce the exposure surface, but they can still propagate private signals sequentially. ICLC obtains the lowest leakage across all three graph families, showing that its gains are not tied to a single topology.

Figure 7 shows leakage growth as the number of communication rounds increases. Raw-Latent-MAS accumulates leakage rapidly, rising from 0.42 at $t = 1$ to 0.91 at $t = 8$. Text-MAS grows more slowly because text messages are lower-bandwidth and more inspectable. ICLC has the flattest curve, indicating that the bottleneck, critic, and gate jointly reduce the amount of private signal available for repeated retransmission.

G.5 Communication Efficiency

Table 10 reports communication efficiency. ICLC reduces the active edge ratio by 37% relative to fully connected raw latent communication and uses the lowest bandwidth per round among latent-communication baselines shown here. The active edge ratio reported here is the validation-selected operating point of full ICLC, while Appendix H.1 reports controlled sparsity sweeps. Fewer active

Method	HotpotQA	GSM8K	MuSiQue	ProofWriter	ToolBench
Text-MAS	68.2	70.1	63.2	63.8	60.4
Text-Redaction-MAS	66.8	68.3	62.9	60.5	58.6
Raw-Latent-MAS	71.5	75.0	68.2	67.4	65.8
Noise-Latent-MAS	69.0	72.1	66.1	64.0	62.3
Calibrated-Noise-MAS	66.4	69.1	63.8	61.5	59.9
Projection-MAS	69.8	72.9	67.0	65.0	63.5
Learned-Projection-MAS	70.4	73.7	67.8	65.8	64.8
Adv-Sanitize-MAS	70.8	74.0	68.3	66.2	65.1
ICLC	71.9	75.8	69.2	67.7	66.3

Table 6: **Full per-benchmark task utility results.** HotpotQA reports exact match, GSM8K/MuSiQue/ProofWriter report accuracy, and ToolBench reports planning success.

Method	Single-Message Leakage ($\downarrow$)				$T = 8$ Trajectory Leakage ($\downarrow$)			
	Attr.	Mem.	Recon.	Avg.	Attr.	Mem.	Recon.	Avg.
Text-MAS	0.535	0.518	0.482	0.512	0.402	0.385	0.353	0.380
Text-Redaction-MAS	0.362	0.348	0.318	0.343	0.300	0.289	0.260	0.283
Raw-Latent-MAS	0.779	0.757	0.705	0.747	0.940	0.916	0.874	0.910
Noise-Latent-MAS	0.601	0.583	0.549	0.578	0.700	0.672	0.636	0.669
Calibrated-Noise-MAS	0.462	0.448	0.421	0.444	0.560	0.531	0.500	0.530
Projection-MAS	0.617	0.604	0.568	0.596	0.690	0.670	0.626	0.662
Learned-Projection-MAS	0.548	0.529	0.502	0.526	0.660	0.638	0.594	0.631
Adv-Sanitize-MAS	0.492	0.481	0.450	0.474	0.724	0.694	0.652	0.690
ICLC	<u>0.428</u>	<u>0.405</u>	<u>0.360</u>	<u>0.398</u>	<u>0.372</u>	<u>0.349</u>	<u>0.329</u>	<u>0.350</u>

Table 7: **Full held-out attack results.** Single-message leakage measures recoverability from one communicated latent message, and trajectory leakage measures recoverability from $T = 8$ multi-round latent transcripts. Lower is better. Bold marks the lowest leakage among all methods, while underlining marks the best result among latent-communication methods.

edges are not only cheaper, but also reduce the number of communicated latent artifacts available to curious receivers.

H Additional Analysis

This section expands the sensitivity and robustness analysis behind Section 5. The goal is not to tune ICLC by hand for a particular metric, but to understand how leakage changes when we vary capacity, adversarial pressure, exposure sparsity, latent slots, layer selection, and attacker coalition size.

H.1 Sensitivity to Loss Weights and Sparsity

Table 11 summarizes three controlled sensitivity studies. Increasing the bottleneck weight β reduces single-message leakage but eventually harms task utility. Increasing the adversarial weight λ gives most of its gain at moderate regularization strength, after which leakage gains saturate while utility drop continues to grow. Reducing the active edge ratio lowers propagation leakage, but overly sparse communication sharply degrades ToolBench planning performance.

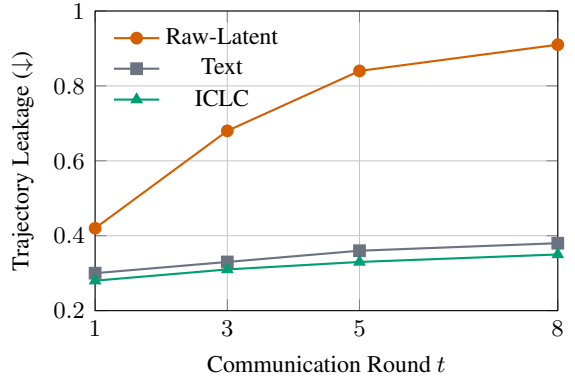


Figure 7: **Trajectory leakage growth over communication rounds.** Raw latent communication exhibits rapid accumulation, while ICLC keeps leakage stable across long-horizon interactions.

The three trends support the same design principle: privacy improves when capacity and exposure are constrained, but overly strong constraints remove task-relevant information. The validation-selected ICLC configuration therefore uses moderate bottlenecking, moderate adversarial pressure,

Variant	Task Utility ($\uparrow$)			Message Leakage ($\downarrow$)			Trajectory Leakage ($\downarrow$)		
	QA	Reason.	Plan	Attr.	Mem.	Recon.	$t = 3$	$t = 5$	$t = 8$
Full ICLC	71.9	70.9	66.3	0.428	0.405	0.360	0.31	0.33	0.35
w/o Bottleneck	71.4	70.2	65.8	0.552	0.526	0.483	0.44	0.58	0.70
w/o Adv. Critic	71.1	69.7	65.4	0.605	0.573	0.531	0.46	0.61	0.73
w/o Topology Gate	71.7	70.5	66.0	0.501	0.472	0.430	0.62	0.76	0.88
w/o All Components	71.5	70.2	65.8	0.779	0.757	0.705	0.68	0.84	0.91

Table 8: **Full component ablation.** The bottleneck and adversarial critic reduce message-level leakage, while the topology-aware gate is most important for suppressing trajectory leakage at later rounds. The row “w/o All Components” corresponds to Raw-Latent-MAS.

Topology	Raw	L-Proj.	AdvSan	ICLC
Fully connected	0.91	0.66	0.69	0.35
Chain	0.74	0.56	0.63	0.31
Star	0.79	0.59	0.68	0.33

Table 9: **Trajectory leakage under different communication topologies.** All values report normalized $T = 8$ trajectory leakage. ICLC gives the strongest reduction across graph structures.

and a learned topology-aware gate rather than a fixed sparse graph.

H.2 Sensitivity to Latent Slots and Layers

Table 12 varies the number of latent slots K . Larger slot budgets improve utility at first because agents can exchange more task-relevant information. However, leakage increases steadily as K grows. The marginal utility gain from $K = 16$ to $K = 32$ is small, while leakage rises substantially. This supports the default choice of a moderate slot budget.

Table 13 varies the layer set $\mathcal{L}$. Early layers have lower utility because they are less aligned with task-level reasoning. Final layers improve utility but expose more private information. Middle and middle-plus-final configurations provide the best trade-off in our reference setting, suggesting that useful communication should expose task abstractions without directly transmitting the most sensitive final-state information.

H.3 Coalition Size Analysis

Table 14 evaluates trajectory attackers with different coalition sizes $|\mathcal{C}|$. Larger coalitions observe more incoming edges and therefore obtain stronger attack signals. Raw-Latent-MAS is highly sensitive to coalition size, rising from 0.62 for a single curious receiver to 0.91 when four receivers collude. ICLC keeps leakage much flatter, increas-

ing only from 0.29 to 0.35. This result supports the system-level interpretation of ICLC: constraining both message capacity and exposure surface reduces the marginal value of compromising additional receivers.

H.4 Learned Communication Graphs

Table 15 summarizes the communication graphs learned by the topology-aware gate. The learned edge ratios differ across task families. Collaborative reasoning uses denser communication because multiple agents need to exchange intermediate constraints. Tool-use planning uses sparser communication because planner-verifier and planner-executor interactions are often sufficient. Private-context QA falls between these settings. This adaptive behavior is important: a fixed sparse graph would reduce leakage but could undercommunicate on reasoning tasks, while a fixed dense graph would unnecessarily expose private context on tasks that need only limited coordination.

H.5 Takeaways from Additional Analysis

The additional analyses lead to four conclusions. First, privacy improves when latent capacity is constrained, but excessive compression harms task utility. Second, adversarial regularization is most useful at moderate strength; beyond that range, leakage gains saturate while utility loss grows. Third, propagation leakage is governed by the exposure surface: topology, edge ratio, communication rounds, and coalition size all change how much private signal can accumulate. Fourth, ICLC’s learned graph is task-adaptive, which explains why it can reduce leakage without eliminating useful collaboration. These results support the main design choice of combining bottlenecking, adversarial leakage regularization, and topology-aware gating in a single communication-layer framework.

Method	Active Edge Ratio	Generated Tokens	Bandwidth / Round	Latency	Avg. Utility
Text-MAS	1.00	620	6.8 KB	1.00×	64.8
Raw-Latent-MAS	1.00	0	48.0 KB	0.82×	69.2
Learned-Projection-MAS	1.00	0	16.0 KB	0.86×	68.1
Adv-Sanitize-MAS	1.00	0	18.4 KB	0.92×	68.5
ICLC	0.63	0	<u>13.1 KB</u>	0.89×	69.7

Table 10: **Communication efficiency.** ICLC reduces active latent exchanges and has the lowest bandwidth among latent-communication methods shown here, while preserving the utility benefit of latent communication. Latency is normalized to Text-MAS. Bold marks the best overall value where directly comparable, and underlining marks the best value among latent-communication methods.

β	EM	Leak.	λ	Attr.	Recon.	Drop	Edge	Plan	Inst.	Prop.
0.01	72.3	0.70	0.0	0.72	0.68	0.0	1.00	66.3	0.43	0.51
0.05	72.1	0.55	0.1	0.58	0.55	0.2	0.75	66.2	0.39	0.45
0.10	71.9	0.43	0.5	0.43	0.40	0.6	0.50	66.0	0.37	0.40
0.50	70.5	0.39	1.0	0.41	0.39	0.9	0.25	64.8	0.35	0.38
1.00	68.0	0.37	2.0	0.40	0.38	1.3	0.10	61.7	0.34	0.36

Table 11: **Sensitivity to bottleneck strength, adversarial strength, and communication sparsity.** Left: effect of β on HotpotQA exact match and average single-message leakage. Middle: effect of λ on the reasoning mixture, where Drop denotes utility drop. Right: effect of fixed active edge ratio on ToolBench planning utility and leakage.

K	Utility ($\uparrow$)	Leakage ($\downarrow$)	Active Slots
2	67.8	0.34	1.3
4	69.0	0.37	2.1
8	69.7	0.40	3.4
16	70.0	0.46	5.8
32	70.1	0.55	9.7

Table 12: **Sensitivity to the number of latent slots.** Larger slot budgets increase capacity and may increase leakage.

Layer Set	Utility ($\uparrow$)	Leakage ($\downarrow$)
Early layers	67.4	0.43
Middle layers	68.9	0.40
Final layers	70.3	0.49
Middle + final	69.7	0.40
All selected layers	69.9	0.45

Table 13: **Sensitivity to layer selection.** Final-layer states improve task utility but can expose more task-irrelevant private information.

$ \mathcal{C} $	Text	Raw-Latent	ICLC
1	0.31	0.62	0.29
2	0.34	0.76	0.32
3	0.36	0.86	0.34
4	0.38	0.91	0.35

Table 14: **Coalition size analysis.** Trajectory leakage increases with coalition size, but ICLC keeps leakage growth substantially flatter than Raw-Latent-MAS.

Task Family	Edge Ratio	Active Slots	Prop. Leak
Private-context QA	0.61	3.1	0.34
Collaborative reasoning	0.70	3.8	0.36
Tool-use planning	0.58	2.9	0.35

Table 15: **Learned communication graph statistics.** The topology-aware gate adapts exposure to the task family.

used only to test reconstruction leakage.

I Qualitative Case Studies and Failure Cases

I.1 Sanitized Reconstruction Example

Table 16 shows a sanitized reconstruction example. Unlike the aggregate leakage scores in the main experiments, this example illustrates what a reconstruction attacker observes in a concrete tool-use planning instance. All secrets are synthetic and are

I.2 Failure Cases

ICLC reduces empirical recoverability under specified attackers, but it does not provide formal privacy guarantees. The method can still fail when task-relevant information is strongly entangled with sensitive information. For example, if a private attribute is necessary for selecting a correct tool or answering the public query, any useful communication protocol may need to expose some related

Field	Content
Public task	Produce a valid tool-use plan for a mock workflow. The final plan does not require exposing the credential string to other agents.
Private local context	The memory agent stores a synthetic credential <code>sk_test_42FAKE_MONARCH</code> for a mock tool.
Raw-Latent attacker output	Recovers a near-complete string containing 42FAKE and MONARCH.
ICLC attacker output	Produces unrelated high-frequency tool tokens and fails to recover the credential substring.
Interpretation	Raw latent messages can carry task-irrelevant confidential substrings, while ICLC suppresses this unnecessary reconstruction channel in this example.

Table 16: **Sanitized secret reconstruction example.** The example uses a synthetic mock credential and contains no real secret.

signal. Similarly, stronger adaptive attackers may recover more information than the training-time critics. These limitations motivate future work on stronger formal guarantees and adaptive privacy budgets for latent communication.

J Reproducibility Details

This appendix summarizes the protocol used to produce the reported results. All methods are evaluated with the same task splits, agent roles, prompts, communication topology, number of rounds, and frozen backbone agents. The backbone LLM parameters are never updated; only communication-layer modules and training-time leakage critics are optimized. Unless otherwise stated, experiments use four agents, $T = 3$ communication rounds, and three random seeds. Long-horizon propagation experiments use up to $T = 8$ rounds.

ICLC implementation. ICLC extracts latent substrates from middle-plus-final hidden layers. The default setting uses $K = 8$ latent slots and communication dimension $d_c = 256$. Messages are generated by a receiver-conditioned bottleneck, filtered by a hard-concrete exposure gate, and consumed through soft-prefix fusion. At inference time, ICLC uses deterministic posterior means and a fixed gate threshold $\delta_g = 0.50$; leakage critics and sensitive labels are not used.

Training and tuning. We train communication modules with AdamW using learning rate 1×10^{-4} , batch size 16, bottleneck weight $\beta = 0.10$, exposure-flow weight $\alpha = 0.05$, and leakage weight $\lambda = 0.50$. Leakage critics use learning rate 2×10^{-4} with two critic updates per communication-module update. All baselines and ICLC variants are selected on validation data using the same privacy–utility criterion. No test examples are used for hyperparameter selection, threshold selection, attacker selection, or early stopping.

Data and attacks. Private-context QA uses HotpotQA with 4,000/500/1,000 train/validation/test examples. Collaborative reasoning uses GSM8K, MuSiQue, and ProofWriter; tool-use planning uses ToolBench. Protected fields are inserted only into agent-local contexts and are not required for the public answer or final plan. They include synthetic canaries, task-irrelevant attributes, protected distractors, mock API keys, and confidential constraints. Final privacy results are computed with held-out attackers trained after the communication module is fixed. Attribute leakage uses MLP classifiers or regressors, membership leakage uses a binary attacker reported as attack advantage, and secret reconstruction uses an autoregressive decoder.

Compute and release. The communication module has 18.6M inference-time parameters. Training-time leakage critics have 24.1M parameters and are discarded after training. The reference setup uses $4 \times \text{A100 80GB GPUs}$, with about 8.5 hours per seed and +6.8% inference latency overhead. Upon publication, we will release preprocessing scripts, task construction code, communication modules, configuration files, and evaluation scripts.

J.1 Ethics and Data Handling

All private fields used in our experiments are synthetic or benchmark-derived protected distractors. We do not use real credentials, real medical records, real financial identifiers, or real personally identifying information. Synthetic secrets such as canaries and mock API keys are inserted only to evaluate whether latent communication can expose task-irrelevant confidential fields. ICLC should not be interpreted as a formal privacy guarantee. It reduces empirical recoverability under specified attacker models, but practical deployment should also include access control, audit logging, data minimization, deletion propagation, and task-level authorization policies.