MedLatent: Latent Multi-Agent Communication for Cross-Hospital Rare-Disease Diagnosis

Anonymous ACL submission

Abstract

Rare diseases affect over 300 million patients across more than 7,000 conditions, yet no single hospital encounters enough cases of any one condition for reliable diagnosis. Cross-hospital collaboration could help by allowing a diagnosing institution to use distributed, case-specific diagnostic evidence, but privacy regulations restrict the transmission of identifiable clinical text across institutional boundaries. This setting raises two challenges: existing medical agent systems often rely on textual evidence exchange, while raw latent states such as hidden states and KV caches may still reveal prompt-derived clinical content. We introduce **MedLatent**, a latent multi-agent communication framework in which hospital agents keep private clinical records and retrieved cases local, and send compact latent KV blocks to a host agent for rare-disease diagnosis. **MedLatent** supports two deployment settings: same-backbone hospital agents use latent KV distillation, while hospitals with different LLM backbones use cross-family latent alignment. On CROSSRARE-BENCH, a self-built large-scale rare-disease benchmark with hospital-level partitions, **MedLatent** improves cross-hospital diagnostic performance while reducing reconstructable clinical content relative to raw-latent communication baselines. Code and data are available at https://anonymous.4open.science/r/MedLatent/.

1 Introduction

Rare diseases collectively encompass over 7,000 conditions and affect more than 300 million people worldwide (Chen et al., 2024), yet each individual condition is so uncommon that any single hospital may encounter only a handful of cases. Diagnostic experience is therefore scattered across institutions, and no single site can accumulate sufficient evidence for reliable recognition. Cross-hospital collaboration could help by allowing the diagnosing institution to use relevant experience stored at other hospitals. However, privacy regulations such as HIPAA, GDPR, and analogous institutional policies restrict the transmission of identifiable clinical text across institutional boundaries (Zhong et al., 2025). This creates the central communication constraint of cross-hospital rare-disease diagnosis: hospitals need to use distributed, case-specific diagnostic evidence, but the clinical records and retrieved text that contain this evidence cannot be shared.

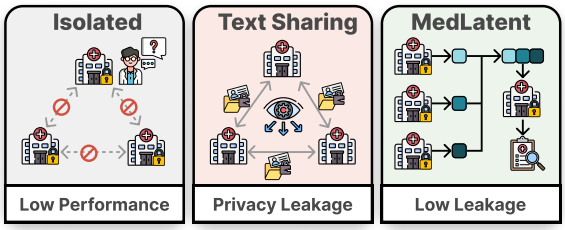


Figure 1: **Motivation for latent cross-hospital communication.** Isolated diagnosis keeps private clinical records local but cannot use distributed diagnostic experience. Direct text sharing enables cross-hospital evidence aggregation but exposes patient-identifiable clinical text. **MedLatent** replaces textual evidence exchange with compact latent KV blocks, allowing hospital agents to transmit diagnosis-oriented state while keeping clinical records and retrieved evidence local.

Challenge 1: Existing collaboration relies on textual evidence exchange. Medical agent systems provide a natural framework for hospital diagnostic reasoning by retrieving medical knowledge, integrating heterogeneous evidence, and performing multi-step reasoning over clinical records (Tang et al., 2024; Kim et al., 2024; Wang et al., 2025a). However, these systems typically communicate through natural-language evidence, such as retrieved cases, summaries, or agent messages. This assumption breaks down in cross-hospital rare-disease diagnosis. Because each rare condition may appear only a few times at any single institution, the evidence most relevant to a new query is often distributed across hospital-local case repositories. Yet the retrieved clinical text that carries this evidence cannot leave the institution that stores it. Hospitals therefore need to share case-specific diag-

nostic signal for the current query without sending the clinical text that encodes it. Figure 1 summarizes the resulting tradeoff: isolated diagnosis keeps records local but cannot use distributed diagnostic experience, while direct text sharing pools evidence but exposes patient-identifiable clinical records. This leaves a key question unanswered: *what should hospital agents exchange during inference when retrieved clinical text cannot be shared?*

Challenge 2: Raw latent states can still leak clinical content. Latent model representations offer a natural candidate for this missing communication object. Instead of sending natural-language evidence, hospital agents could send internal model states such as hidden states or KV-cache representations, which recent work has shown can support reasoning and model collaboration (Hao et al., 2025; Zou et al., 2025; Du et al., 2026; Fu et al., 2026). However, replacing text with latent states does not automatically resolve the leakage concern. Embeddings, hidden states, and KV caches can retain substantial information about the original prompt, including retrieved private cases (Morris et al., 2023; Dong et al., 2025). Thus, cross-hospital latent communication should be evaluated not only for diagnostic utility, but also for empirical reconstruction leakage from transmitted latent states.

To address these two challenges, we introduce `MedLatent`, a latent multi-agent communication framework for cross-hospital rare-disease diagnosis. The key idea is that each hospital agent keeps its private clinical records and retrieved cases local, reasons over local evidence, and sends only a compact latent KV block, rather than retrieved text or raw model states, to a host agent for diagnosis. Compared with raw model states, this compact block reduces how much clinical content is reconstructable from the transmitted representation. The same framework extends to settings where hospitals use different LLM backbones by aligning local latent states to the host agent's latent space. Our contributions are summarized as follows.

- **Formulation.** We formulate cross-hospital rare-disease diagnosis as a clinical-text-restricted latent communication problem, where hospital agents exchange compact latent blocks rather than retrieved clinical text during inference.
- **Method.** We instantiate `MedLatent` in two settings: `MedLatent-H` for same-backbone hospital agents through latent KV distillation, and `MedLatent-X` for cross-family hospital agents through latent alignment.
- **Benchmark.** We build CROSSRARE-BENCH, a large-scale cross-hospital rare-disease diagnosis benchmark with hospital-level partitions for evaluating inference-time collaboration under text-sharing constraints.
- **Evaluation.** We evaluate both diagnostic utility and empirical clinical-content leakage, showing that `MedLatent` improves cross-hospital diagnostic performance while reducing reconstructable clinical content relative to other latent communication baselines.

2 Related Work

Medical diagnostic agents. Recent medical LLM agents improve clinical reasoning through retrieval, multi-agent collaboration, and knowledge-grounded decision making (Tang et al., 2024; Kim et al., 2024; Wang et al., 2025a). This direction has been extended to evidence-based diagnostic workflows, EHR-grounded decision support, hospital-scale agent simulation, and multi-disciplinary rare-disease reasoning (Wang et al., 2026, 2025b; Li et al., 2024; Chen et al., 2026a). For rare diseases, DeepRare advances agentic differential diagnosis, while PhenoBrain and RareBench provide phenotype-based diagnostic modeling and benchmark evaluation (Zhao et al., 2026; Mao et al., 2025; Chen et al., 2024). These systems also build on phenotype-driven pipelines that match patient phenotypes to candidate diseases or genes (Birgmeier et al., 2020; Robinson et al., 2020; Zhai et al., 2023). Despite these advances, existing diagnostic agents and phenotype-based systems mainly operate within a single clinical trust boundary or rely on textual clinical evidence. They therefore do not define what hospital agents should exchange during inference when private clinical text cannot be shared across institutions.

Privacy-aware hospital collaboration. Privacy-aware collaboration has a long history in medical AI, especially through federated learning and hospital deployment frameworks. Federated LLM methods share gradients, adapters, LoRA updates, or model parameters rather than patient records (Ye et al., 2024; Zhang et al., 2024; Liu et al., 2025), and medical FL infrastructure has been developed for real cross-institution settings (Pati et al., 2022; Roth et al., 2023; Cremonesi et al., 2023). These methods aggregate distributed experience into training-time updates, but they do not let a diagnosing hospital condition the current query on

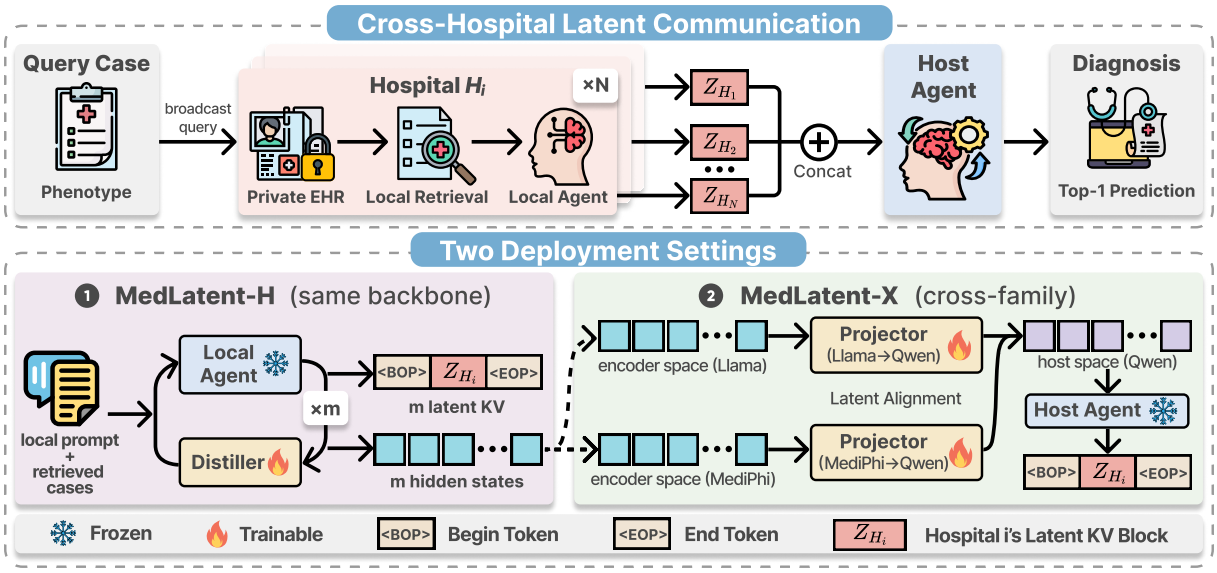


Figure 2: **Overview of `MedLatent`. Top:** Hospital agents keep private EHR records local, retrieve relevant local cases, and send compact latent blocks to a host agent for diagnosis. **Bottom:** `MedLatent` supports two deployment settings: `MedLatent-H` uses latent KV distillation for same-backbone hospital agents, while `MedLatent-X` uses lightweight projectors for cross-family latent alignment when hospitals use different LLM backbones.

case-specific evidence retrieved from other hospitals' local databases. A complementary line studies privacy-aware LLM inference by splitting models, adding local privacy noise, or compressing transmitted representations (Du et al., 2023; Mai et al., 2024; Yang et al., 2024). These approaches protect single-client inference or delegated computation, whereas `MedLatent` operates in a different regime: patient-specific cross-hospital collaboration during inference, where hospitals exchange diagnostic signal without transmitting clinical text.

Latent communication. Recent work has explored whether LLMs can reason or communicate through continuous internal representations rather than text. Coconut performs continuous reasoning by feeding hidden states back as input embeddings (Hao et al., 2025), while related work studies soft prompts, pause tokens, and compressed KV-style representations (Li and Liang, 2021; Mu et al., 2023; Goyal et al., 2024; Zhang et al., 2025; Kuzina et al., 2026). In multi-agent and cross-model settings, LatentMAS and Interlat exchange latent states, while Cache-to-Cache and KV-alignment methods learn KV-cache projection or fusion across LLMs (Zou et al., 2025; Du et al., 2026; Fu et al., 2026; Dery et al., 2026). Systems work further shows that KV caches can be compressed, reused, and transmitted efficiently (Zhang et al., 2023; Xiao et al., 2024; Liu et al., 2024). Recent position work on FedRefine argues for federated inference with heterogeneous LLMs and KV-cache communication (Chen et al., 2026b). Together, these studies establish latent communication as a promising model-collaboration mechanism, but mainly target general reasoning, efficiency, or same-input cross-model communication rather than cross-hospital diagnosis, where agents retrieve private cases and require joint evaluation of diagnostic utility and empirical reconstruction leakage.

3 Method

A central question in cross-hospital rare-disease diagnosis is what hospital agents should exchange during inference. This communication object should avoid both textual evidence exchange and raw prompt-length KV exchange: retrieved clinical text cannot be shared across hospitals, while full KV states can still expose prompt-derived clinical content. `MedLatent` meets these requirements by having each hospital agent keep retrieved cases, local prompts, and prompt-length KV caches local, while sending only a fixed-length diagnosis-oriented latent block instead of retrieved text or raw KV states. We first formalize the inference and reconstruction-leakage settings in Sec. 3.1, then present `MedLatent-H` for same-backbone latent distillation in Sec. 3.2 and `MedLatent-X` for cross-family latent alignment in Sec. 3.3.

3.1 Problem Setup

Task formulation. We consider a cross-hospital network of hospital agents $\mathcal{S} = \{H_1, \ldots, H_N\}$.

Each hospital agent H_i has a local LLM backbone and a private case database $\mathcal{D}_i$ of historical patient records. Given a query case, we represent the query only as a standardized HPO phenotype set q_{HPO}, whose codes are deterministically rendered into canonical HPO term names for prompting. The query is therefore not a free-text clinical note. This structured phenotype representation is broadcast to all hospital agents. Each hospital agent then retrieves relevant cases only from its own private database $\mathcal{D}_i$ and constructs a local prompt x_i from the query phenotypes and retrieved local cases. The local prompt x_i, raw clinical text, and retrieved patient records never leave the hospital that stores them. During inference, each hospital agent sends only a transmitted representation $\mathcal{T}_{H_i}$ to the host agent, typically the query hospital agent, and the goal is to produce a single OMIM-coded diagnosis:

$$\hat{y} = \text{Host}\left(q_{\text{HPO}}, \bigoplus_{i=1}^{N} \mathcal{T}_{H_i}\right), \ \hat{y} \in \mathcal{Y}_{\text{OMIM}}. \quad (1)$$

Here, $\mathcal{T}_{H_i}$ denotes the transmitted object. In `MedLatent`, $\mathcal{T}_{H_i} = \mathcal{Z}_{H_i}$ is the compact latent block, while in the raw-KV baseline, $\mathcal{T}_{H_i} = \mathcal{M}_{H_i}$ is the prompt-length KV cache.

Reconstruction leakage setting. We evaluate reconstruction leakage under a passive observer of the inter-hospital communication channel. The observer can read the transmitted object $\mathcal{T}_{H_i}$ from each hospital agent, such as a prompt-length KV cache in the raw-KV baseline or a compact latent block in `MedLatent`. The observer knows the participating LLM architectures and weights, but cannot access private hospital databases. Given $\mathcal{T}_{H_i}$, an attacker $\mathcal{A}$ attempts to reconstruct the local prompt content x_i, which includes retrieved clinical evidence from hospital H_i:

$$\tilde{x}_i = \mathcal{A}(\mathcal{T}_{H_i}), \qquad \rho_i = \text{sim}(\tilde{x}_i, x_i). \quad (2)$$

We instantiate $\mathcal{A}$ as a frozen-LLM continuation attack: the attacker injects the intercepted transmitted state as past key–value states into the frozen LLM that produced them, prompts it to repeat the previous hospital case content, and greedily decodes the reconstruction. We instantiate sim with token F1 and BERTScore, and report this empirical reconstruction leakage alongside diagnostic utility. Appendix I further evaluates prompt-based extraction attacks for retrieved disease-label and phenotype leakage.

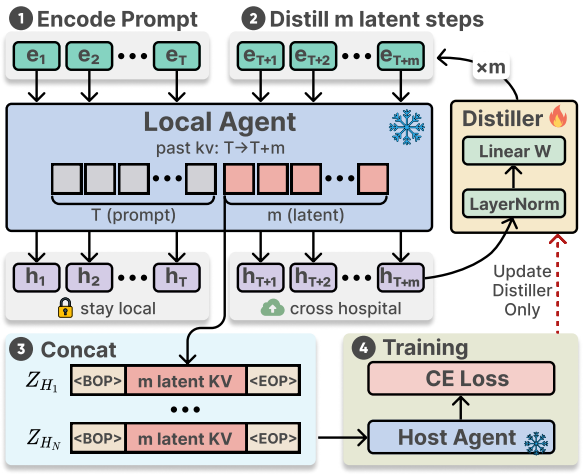


Figure 3: `MedLatent-H`: **same-backbone latent distillation.** A hospital agent encodes its local prompt with a frozen LLM backbone, keeping the prompt-length states local. A trainable distiller generates m latent steps, and only the corresponding m latent KV positions are sent as the hospital's compact block. The host agent aggregates these blocks for diagnosis. Training uses diagnosis cross-entropy and updates only the distiller and boundary embeddings.

3.2 `MedLatent-H`: Same-Backbone Latent Distillation

We first consider the same-backbone setting, where all hospital agents use the same LLM backbone. Their KV spaces are naturally aligned, so the main challenge is compression: each hospital agent must summarize its local prompt state into a compact latent block rather than transmitting the full prompt-length KV cache.

Local retrieval and prompt encoding. Each hospital agent receives the broadcast phenotype query q_{HPO} and retrieves phenotypically similar cases only from its private database $\mathcal{D}_i$ using an HPO-based retrieval procedure. The retrieved cases are combined with the query phenotypes to form a local prompt x_i, which remains inside hospital H_i. A frozen local LLM backbone then encodes x_i and produces a prompt-length KV cache

$$\mathcal{M}_{H_i} = \{(\mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)})\}_{l=1}^{L}, \ \mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)} \in \mathbb{R}^{T_i \times d}, \quad (3)$$

where L is the number of layers, T_i is the local prompt length, d is the per-layer KV dimension, and $\mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)}$ are the layer-l key and value states. Directly transmitting $\mathcal{M}_{H_i}$ gives the raw-KV baseline: it shares no explicit clinical text, but exposes internal representations of the entire local prompt, including retrieved private cases. `MedLatent-H` instead keeps $\mathcal{M}_{H_i}$ local and transmits only a compact latent block $\mathcal{Z}_{H_i}$ distilled from it. Details of the retrieval procedure are provided in Appendix B.

Distilling compact latent blocks. Within each hospital agent, starting from the final prompt hidden state h_{T_i}, we generate compact latent positions autoregressively. We use a minimal distiller so that performance gains can be attributed to the compact latent communication mechanism rather than to additional model capacity. The distiller ϕ maps the current hidden state back to the input-embedding space,

$$\phi(h) = W \cdot \mathrm{LayerNorm}(h), \tag{4}$$

where $W \in \mathbb{R}^{d\times d}$ is a bias-free linear projection. The resulting vector is fed into the frozen local LLM backbone as the next input embedding without decoding text. Repeating this procedure for m latent steps produces m latent positions. `MedLatent-H` then keeps only the KV states corresponding to these latent positions, denoted by $\tilde{\mathcal{M}}_{H_i}$. This compact cache has the same layer-wise key–value structure as $\mathcal{M}_{H_i}$, but contains only the m latent positions rather than all T_i prompt positions. We wrap it with learned boundary embeddings to form the transmitted hospital block

$$\mathcal{Z}_{H_i} = \big[\langle \mathrm{BOP}\rangle, \tilde{\mathcal{M}}_{H_i}, \langle \mathrm{EOP}\rangle\big]. \tag{5}$$

Here, $\langle \mathrm{BOP}\rangle$ and $\langle \mathrm{EOP}\rangle$ are learned begin- and end-of-block embeddings that delimit each hospital agent's contribution in the latent chain. The transmitted block has a fixed length of m latent positions, independent of the local prompt length T_i. The host agent aggregates $\mathcal{Z}_{H_1} \oplus \cdots \oplus \mathcal{Z}_{H_N}$ and produces the final diagnosis.

Training objective. We train the distiller ϕ and the boundary embeddings using diagnosis cross-entropy on the host agent's output. Let $y = (y_1, \ldots, y_{|y|})$ denote the target answer sequence. The loss is

$$\mathcal{L}_{\mathrm{Distill}} = -\sum_{t=1}^{|y|} \log p\big(y_t \mid \mathcal{Z}_{H_{1:N}},\, y_{<t}\big), \tag{6}$$

where $\mathcal{Z}_{H_{1:N}}$ denotes the concatenated latent blocks from all hospitals. The loss supervises the final diagnosis tokens generated by the host agent. Since the intermediate latent positions have no target text labels, they are learned indirectly through gradients from this diagnosis loss. During training, we update only the distiller and boundary embeddings, while all local and host LLM backbones remain frozen. We train the latent interface centrally on the public training split under simulated hospital partitions, with train and test cases disjoint.

3.3 `MedLatent-X`: Cross-Family Latent Alignment

In practice, hospitals may deploy different LLM backbones due to local infrastructure or domain adaptation. Different backbones may have different hidden dimensions and representation statistics, so a compact latent block produced by one hospital agent may not be directly consumable by the host agent. `MedLatent-X` addresses this mismatch through cross-family latent alignment.

Cross-family projector. To handle cross-family settings, `MedLatent-X` uses a lightweight projector to map compact latent states into the host agent's input-embedding space. Let E denote an encoder backbone family and H denote the host backbone family. Let $\mathbf{h}^{(E)}_{1:m_E} \in \mathbb{R}^{m_E\times d_E}$ denote the concatenated compact latent states from hospital agents using encoder family E, where m_E is the total number of latent positions in that group. The projector maps this sequence to host input embeddings

$$\mathbf{e}^{(H)}_{1:m_E} = \psi_{E\to H}\big(\mathbf{h}^{(E)}_{1:m_E}\big) \in \mathbb{R}^{m_E\times d_H}. \tag{7}$$

In our implementation, $\psi_{E\to H}$ is a token-wise LayerNorm followed by a bias-free linear projection. The projected sequence is wrapped with boundary tokens and passed to the frozen host LLM through input embeddings. The host backbone then materializes host-space latent KV states for final aggregation and diagnosis.

Joint projector training. For each host backbone family H, we train one projector $\psi_{E\to H}$ for each non-host encoder family E. During training, each batch samples a hospital-family composition from the available backbone families, while ensuring that at least one non-host family is present. Native-family hospital agents use the host family's frozen latent distiller, while non-host-family agents use their frozen encoder-side latent distillers followed by the corresponding cross-family projector. The host agent aggregates all native and projected latent blocks and is supervised with the same diagnosis cross-entropy objective as Eq. 6. Only the projectors and boundary embeddings are updated. The encoder LLM backbones, encoder-side distillers, and host LLM backbone remain frozen. The resulting projectors can be reused across different hospital compositions at inference time, without retraining for each mixture.

Table 1: **Diagnostic utility on CROSSRARE-BENCH.** We report accuracy (Acc), macro F1 (F1), and LLM-as-judge score (Judge) on three source cohorts and the full benchmark. RareBench, Zenodo, and Phenopacket are the three cohorts that make up CROSSRARE-BENCH, while Overall reports performance on the full test set. Methods are grouped into non-collaborative, text/structured-sharing, and latent-communication categories. Within each model block, **bold** marks the best and underline marks the second best per column. Pink cells denote `MedLatent`. `MediPhi` results are reported in Table 13.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
	Non-collaborative												
Qwen3-4B	Zero-shot	0.11	0.08	0.30	0.03	0.05	0.37	0.01	0.01	0.18	0.03	0.02	0.24
	Single RAG	0.53	0.37	0.58	0.55	0.44	0.59	0.58	0.51	0.70	0.57	0.47	0.66
	Text / structured sharing												
	Dist. RAG	0.56	0.44	0.72	0.65	0.62	0.71	0.61	0.54	0.72	0.61	0.53	**0.72**
	Global RAG	0.58	0.41	0.68	0.63	0.61	0.67	0.62	0.55	0.72	0.62	0.53	0.70
	Struct. RAG	0.61	0.45	0.70	0.64	0.56	0.66	**0.64**	0.56	**0.73**	**0.64**	0.52	0.71
	TextMAS	0.53	0.37	0.61	0.49	0.49	0.65	0.56	0.48	0.72	0.54	0.44	0.69
	Latent communication												
	Raw KV	0.63	0.45	**0.75**	**0.67**	**0.64**	**0.73**	0.56	0.49	0.70	0.59	0.48	**0.72**
	LatentMAS	0.61	0.46	0.72	0.66	0.63	0.71	0.57	0.51	0.71	0.60	0.49	0.71
	MedLatent	**0.72**	**0.61**	**0.75**	0.59	0.61	0.67	**0.64**	**0.63**	**0.73**	**0.64**	**0.61**	**0.72**
	Non-collaborative												
Llama-3.2-3B	Zero-shot	0.02	0.01	0.21	0.02	0.02	0.09	0.00	0.00	0.10	0.01	0.00	0.11
	Single RAG	0.37	0.36	0.53	0.28	0.29	0.43	0.43	0.37	0.62	0.39	0.34	0.57
	Text / structured sharing												
	Dist. RAG	0.44	0.40	0.60	0.37	0.44	0.62	0.47	0.40	0.64	0.45	0.40	0.63
	Global RAG	0.46	0.41	0.53	0.36	0.45	0.56	0.50	0.44	0.66	0.47	0.42	0.62
	Struct. RAG	0.53	0.35	0.63	0.31	0.35	0.65	0.43	0.37	0.69	0.42	0.34	0.67
	TextMAS	0.33	0.24	0.49	0.33	0.39	0.48	0.31	0.25	0.47	0.32	0.25	0.47
	Latent communication												
	Raw KV	0.28	0.24	0.46	0.27	0.39	0.50	0.40	0.35	0.64	0.35	0.32	0.58
	LatentMAS	0.23	0.20	0.44	0.26	0.34	0.60	0.38	0.35	0.62	0.33	0.31	0.59
	MedLatent	**0.75**	**0.64**	**0.77**	**0.70**	**0.62**	**0.74**	**0.73**	**0.72**	**0.78**	**0.73**	**0.68**	**0.77**

4 Experiments

We evaluate `MedLatent` on diagnostic utility and empirical reconstruction leakage. Experiments use CROSSRARE-BENCH, a rare-disease benchmark built from public case repositories and partitioned into hospital-local databases. We compare against non-collaborative, text/structured-sharing, and latent communication baselines, then report same-backbone, cross-family, and analysis results.

4.1 Benchmark Construction

Source datasets. We build CROSSRARE-BENCH from three public rare-disease case repositories: RareBench (Chen et al., 2024), the Zenodo phenotype–disease collection (Chimirri et al., 2025), and the GA4GH Phenopacket Store (Danis et al., 2025). Each case contains HPO codes and a ground-truth OMIM disease identifier. After label resolution, deduplication, and frequency filtering, the benchmark contains 8,022 cases across 235 OMIM diseases. We split them 90/5/5 into 7,220 training, 401 validation, and 401 test cases. Full filtering details are provided in Appendix A.

Hospital-level partition. We partition the training cases into five hospital-local databases. During inference, each hospital agent retrieves only from its own private database $\mathcal{D}_i$, and no agent can access another hospital's case records. This distributes diagnostic evidence across hospitals while keeping retrieved clinical text local. Detailed partition statistics are provided in Appendix A.

4.2 Experimental Setup

Baselines. We compare `MedLatent` against eight baselines in three groups. ❶ **Non-collaborative baselines:** `Zero-shot` uses only the query phenotypes, and `Single RAG` retrieves cases only from the query hospital's local database. ❷ **Text / structured-sharing baselines:** `Distributed RAG` lets each hospital retrieve from its own database and shares the retrieved text with the host agent. `Global RAG` pools all hospital databases into a single global store, representing a centralized text-sharing setting with full retrieval access. `Struct. RAG` shares only structured retrieval outputs, namely each hospital's retrieved disease label and HPO similarity score, without transmitting retrieved case text. `TextMAS` is a parallel text-based multi-agent baseline in which hospital agents communicate through natural-language assessments. ❸ **Latent communication baselines:** `Raw KV` trans-

Table 2: **Empirical reconstruction leakage from transmitted latent representations.** We report lower-is-better Token F1 and BERTScore under the same passive observer attack. The reduction column shows the relative drop from `Raw KV` to **`MedLatent`**. Text-sharing baselines directly transmit retrieved clinical text, so their leakage is direct exposure rather than reconstruction from a latent representation.

Model	Metric	Raw KV	LatentMAS	MedLatent	Reduc.
Qwen3-4B	Token F1 ↓	0.60	0.60	**0.07**	↓88%
	BERTScore ↓	0.93	0.93	**0.81**	↓13%
Llama-3.2-3B	Token F1 ↓	0.68	0.82	**0.07**	↓90%
	BERTScore ↓	0.95	0.97	**0.78**	↓18%
MediPhi	Token F1 ↓	0.76	0.75	**0.07**	↓91%
	BERTScore ↓	0.95	0.95	**0.78**	↓18%

Table 3: **Effect of cross-family heterogeneity.** We vary the model-family composition of three hospital agents. Deg0, Deg1, and Deg2 use one, two, and three model families, respectively. ΔAcc reports the absolute accuracy change from Deg0 on the percentage-point scale. Q = `Qwen3-4B`, L = `Llama-3.2-3B`, and M = `MediPhi`.

Host	Degree (comp.)	Acc	F1	Judge	ΔAcc
Qwen3-4B	Deg0 (Q:Q:Q)	0.64	0.61	0.72	–
	Deg1 (Q:L:Q)	0.63	0.60	0.70	−0.5
	Deg2 (Q:L:M)	0.62	0.60	0.69	−2.2
Llama-3.2-3B	Deg0 (L:L:L)	0.73	0.68	0.77	–
	Deg1 (L:Q:L)	0.72	0.66	0.76	−0.7
	Deg2 (L:Q:M)	0.71	0.65	0.76	−1.6

mits the full prompt-length KV cache defined in §3.2. `LatentMAS` is a latent multi-agent baseline that exchanges latent states without **`MedLatent`**'s compact KV distillation. We evaluate all methods across three host LLMs: `Llama-3.2-3B`, `Qwen3-4B`, and `MediPhi`.

Evaluation metrics. We evaluate diagnostic utility and empirical reconstruction leakage. For diagnostic utility, we report accuracy, macro F1, and an LLM-as-judge score using `Qwen3-30B`. Accuracy requires an exact OMIM match, while the judge score compares the predicted diagnosis with the ground-truth disease and is less sensitive to disease-name surface forms. For reconstruction leakage, we report token F1 and BERTScore between the reconstructed text and the source local prompt containing retrieved clinical evidence. Lower values indicate less reconstructable clinical content under the attack defined in Sec. 3.1.

Implementation details. We set m=32 latent positions in all main experiments and use N=3 hospital agents unless otherwise specified. Distributed methods retrieve one case from each hospital-local database, while `Global RAG` retrieves the top three cases from the pooled database, giving all retrieval-based methods the same total retrieval budget. We fix this budget to isolate the effect of the communication mechanism from retrieval-depth changes. Detailed configurations are provided in Appendix D.

4.3 Diagnostic Utility

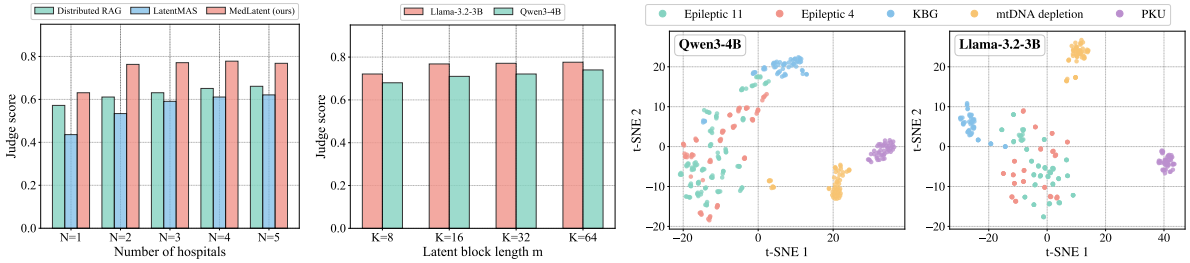
Table 1 reports diagnostic performance in the same-backbone setting, where all hospital agents and the host use the same LLM backbone. Across all three host LLMs, including `MediPhi` results in Appendix G, **`MedLatent`** achieves the strongest overall utility, with the best overall accuracy and macro F1 and competitive or best judge scores. These gains show that compact latent blocks can serve as an effective diagnosis-oriented communication object for aggregating distributed hospital evidence.

The baseline comparisons further show that the improvement is not due to latent communication alone. Text/structured-sharing baselines expose the host to retrieved cases, natural-language messages, or structured retrieval outputs, but aggregate them through prompt-level context. Raw latent baselines avoid explicit text, but full KV caches or unconstrained latent states do not consistently improve diagnosis. In contrast, **`MedLatent`** learns compact diagnosis-oriented latent blocks under the final diagnosis objective, which better supports cross-hospital aggregation. `Struct. RAG` is competitive on some hosts, but remains below **`MedLatent`** overall, suggesting that disease labels and retrieval scores are useful but too coarse to replace compact latent blocks.

4.4 Reconstruction Leakage

Table 2 reports empirical reconstruction leakage from transmitted representations under the attack defined in Sec. 3.1. We measure how much of each hospital's local prompt can be recovered using token F1 and BERTScore. Text-sharing baselines are not listed because they directly transmit retrieved clinical text. For a passive observer, the source content is exposed, so reconstruction is trivial. Among latent communication methods, both `Raw KV` and `LatentMAS` leak substantial prompt-derived content, showing that latent states alone do not remove reconstruction leakage.

`MedLatent` consistently reduces leakage across all three model families. Token F1 drops by 88–91% relative to `Raw KV`, indicating much less exact token recovery, while BERTScore decreases by 13–18%. Because original BERTScore can remain high for templated medical prompts with shared terminology, Appendix H also reports rescaled



(a) Hospital agents N (b) Latent block length m (c) Disease structure in latent space

Figure 4: **Analysis of compact latent communication.** (a) Performance as the number of hospital agents varies. (b) Performance as the latent KV block length varies. (c) t-SNE visualization of compact latent blocks from two encoder families. `MedLatent` benefits from additional hospital evidence and remains effective with compact blocks. The t-SNE panels suggest disease-discriminative structure within each encoder family, but do not imply cross-family alignment.

BERTScore normalized against a random-sentence baseline. Under this normalization, `MedLatent` obtains negative scores, indicating that its reconstructions are less similar to the source prompt than random text. Together with Table 1, these results show that `MedLatent` preserves diagnostic utility while reducing the clinical content reconstructable from transmitted latent states. Appendix I further shows that this reduction extends to retrieved disease-label and phenotype-term leakage, which `MedLatent` keeps at or below 20% even under a template-aware extraction attack.

4.5 Cross-Family Deployment

Table 3 evaluates `MedLatent-X` when hospital agents use different LLM backbone families. As the composition changes from Deg0 to Deg2, the host aggregates compact latent blocks from increasingly heterogeneous encoder families, making alignment more difficult. Performance drops only mildly for both hosts: accuracy decreases by a small amount, macro F1 remains stable, and the judge score is largely preserved. This suggests that the cross-family projectors retain most diagnosis-relevant signal when mapping non-host latent states into the host space. The resulting host-specific projectors can be reused across hospital compositions without retraining for each mixture.

4.6 Latent Communication Analysis

Hyperparameter sensitivity. Figure 4a and Figure 4b analyze the number of hospital agents N and the latent block length m. As N increases, `MedLatent` receives evidence from more hospital-local databases and achieves stronger diagnostic performance, while outperforming other baselines. This suggests that compact latent blocks can aggregate additional hospital evidence rather than being overwhelmed by more inputs. Varying m shows the expected capacity–cost tradeoff: larger latent blocks carry more diagnosis-relevant information and generally improve performance, but the gains become moderate after $m = 32$. Since larger blocks also increase transmitted state length and inference cost, we use $m = 32$ as a practical balance in the main experiments. Appendix E further shows that the m=32 block gives a 5–6$\times$ communication reduction relative to `Raw KV` baseline.

Latent disease structure. Figure 4c visualizes compact latent blocks from two encoder families using t-SNE. Several diseases form separated clusters within each encoder family, suggesting that the learned latent blocks retain disease-discriminative structure after compression. Related disease subtypes can partially overlap, which is consistent with diagnostic similarity rather than arbitrary label separation. The visualization therefore suggests that compact latent blocks preserve disease-relevant structure within each encoder family.

5 Conclusion

We introduced `MedLatent`, a latent multi-agent communication framework for cross-hospital rare-disease diagnosis. `MedLatent` replaces textual evidence exchange and raw prompt-length KV exchange with compact latent blocks, allowing hospital agents to keep retrieved cases and local prompts local while sharing diagnosis-oriented state. We instantiate this idea through same-backbone latent distillation and cross-family latent alignment. Experiments on CROSSRARE-BENCH show that `MedLatent` improves diagnostic utility while reducing empirical reconstruction leakage from transmitted latent representations. These results suggest that compact latent blocks are a promising communication object for clinical-text-restricted cross-hospital collaboration.

6 Limitations

CROSSRARE-BENCH is built from publicly available rare-disease case repositories, which provide standardized HPO-based cases and OMIM labels but do not cover the full diversity of real clinical records. The current evaluation uses a controlled set of source repositories, hospital partitions, LLM backbones, hospital-agent configurations, and retrieval budgets. Broader deployments may involve larger case collections, more disease categories, stronger institution-specific data patterns, more agent backbones, and different retrieval or collaboration policies. Finally, our leakage analysis uses prompt-based reconstruction and extraction attacks with a frozen white-box model rather than trained inversion or probing models. A more complete characterization would include learned attribute probes, stronger reconstruction attacks, inference attacks, and other adaptive adversaries.

References

Johannes Birgmeier, Maximilian Haeussler, Cole A. Deisseroth, Ethan H. Steinberg, Karthik A. Jagadeesh, Alexander J. Ratner, Harendra Guturu, Aaron M. Wenger, Mark E. Diekhans, Peter D. Stenson, David N. Cooper, Christopher Ré, Alan H. Beggs, Jonathan A. Bernstein, and Gill Bejerano. 2020. AMELIE speeds Mendelian diagnosis by matching patient phenotype and genotype to primary literature. *Science Translational Medicine*, 12(544):eaau9113.

Xuanzhong Chen, Ye Jin, Xiaohao Mao, Lun Wang, Shuyang Zhang, and Ting Chen. 2026a. Rareagents: Autonomous multi-disciplinary team for rare disease diagnosis and treatment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):101–109. AAAI 2026 Oral.

Xuanzhong Chen, Xiaohao Mao, Qihan Guo, Lun Wang, Shuyang Zhang, and Ting Chen. 2024. Rarebench: Can llms serve as rare diseases specialists? In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, pages 4850–4861, Barcelona, Spain. ACM.

Zihan Chen, Zeshen Li, Howard H Yang, Tony QS Quek, and Jihong Park. 2026b. Federated inference for heterogeneous llm communication and collaboration. *arXiv preprint arXiv:2603.28772*.

Leonardo Chimirri, J Harry Caufield, Yasemin Bridges, Nicolas Matentzoglu, Michael Gargano, Mario Cazalla, Shihan Chen, Daniel Danis, Alexander JM Dingemans, Klara Gehle, et al. 2025. Consistent performance of large language models in rare disease diagnosis across ten languages and 4917 cases. *EBioMedicine*, 121.

Francesco Cremonesi, Marc Vesin, Sergen Cansiz, Yannick Bouillard, Irene Balelli, Lucia Innocenti, Santiago Silva, Samy-Safwan Ayed, Riccardo Taiello, Laetita Kameni, Richard Vidal, Fanny Orlhac, Christophe Nioche, Nathan Lapel, Bastien Houis, Romain Modzelewski, Olivier Humbert, Melek Önen, and Marco Lorenzi. 2023. Fed-BioMed: Open, transparent and trusted federated learning for real-world healthcare applications. *Preprint*, arXiv:2304.12012.

Daniel Danis, Michael J Bamshad, Yasemin Bridges, Andrés Caballero-Oteyza, Pilar Cacheiro, Leigh C Carmody, Leonardo Chimirri, Jessica X Chong, Ben Coleman, Raymond Dalgleish, et al. 2025. A corpus of ga4gh phenopackets: case-level phenotyping for genomic diagnostics and discovery. *Human Genetics and Genomics Advances*, 6(1).

Lucio Dery, Eran Yahav, Joseph Prior, Shaojin Feng, Yi Shen, and Arthur Szlam. 2026. Latent space communication via K-V cache alignment. *Preprint*, arXiv:2601.06123. Google DeepMind technical report.

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 (USENIX Security '25)*, pages 1629–1648, Seattle, WA, USA. USENIX Association.

Minxin Du, Xiang Yue, Sherman S. M. Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. 2023. DP-Forward: Fine-tuning and inference on language models with differential privacy in forward pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23)*, pages 2665–2679, New York, NY, USA. Association for Computing Machinery.

Zhuoyun Du, Runze Wang, Huiyu Bai, Zouying Cao, Xiaoyong Zhu, Yu Cheng, Bo Zheng, Wei Chen, and Haochao Ying. 2026. Enabling agents to communicate entirely in latent space. *Preprint*, arXiv:2511.09149. Accepted to ACL 2026.

Tianyu Fu, Zihan Min, Hanling Zhang, Jichao Yan, Guohao Dai, Wanli Ouyang, and Yu Wang. 2026. Cache-to-cache: Direct semantic communication between large language models. In *The Fourteenth International Conference on Learning Representations (ICLR)*. ICLR 2026.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2025. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling (COLM)*.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, and Hae Won Park. 2024. Mdagents: An adaptive collaboration of llms for medical decision-making. In *Advances in Neural Information Processing Systems*, volume 37, pages 79410–79452. Curran Associates, Inc. Oral.

Anna Kuzina, Maciej Pioro, Paul N. Whatmough, and Babak Ehteshami Bejnordi. 2026. KaVa: Latent reasoning via compressed KV-cache distillation. In *The Fourteenth International Conference on Learning Representations (ICLR)*.

Junkai Li, Yunghwei Lai, Weitao Li, Jingyi Ren, Meng Zhang, Xinhui Kang, Siyu Wang, Peng Li, Ya-Qin Zhang, Weizhi Ma, and Yang Liu. 2024. Agent hospital: A simulacrum of hospital with evolvable medical agents. *Preprint*, arXiv:2405.02957.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiao-Yang Liu, Rongyi Zhu, Daochen Zha, Jiechao Gao, Shan Zhong, Matt White, and Meikang Qiu. 2025. Differentially private low-rank adaptation of large language model using federated learning. *ACM Transactions on Management Information Systems*, 16(2):1–24.

Yuhan Liu, Hanchen Li, Yihua Cheng, Siddhant Ray, Yuyang Huang, Qizheng Zhang, Kuntai Du, Jiayi Yao, Shan Lu, Ganesh Ananthanarayanan, Michael Maire, Henry Hoffmann, Ari Holtzman, and Junchen Jiang. 2024. CacheGen: KV cache compression and streaming for fast large language model serving. In *Proceedings of the ACM SIGCOMM 2024 Conference*, pages 38–56, Sydney, Australia. ACM.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. 2024. Split-and-denoise: Protect large language model inference with local differential privacy. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 34281–34302. PMLR.

Xiaohao Mao, Yu Huang, Ye Jin, Lun Wang, Xuanzhong Chen, Honghong Liu, Xinglin Yang, Haopeng Xu, Xiaodong Luan, Ying Xiao, Siqin Feng, Jiahao Zhu, Xuegong Zhang, Rui Jiang, Shuyang Zhang, and Ting Chen. 2025. A phenotype-based AI pipeline outperforms human experts in differentially diagnosing rare diseases using EHRs. *npj Digital Medicine*, 8(1):68.

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 (EMNLP)*, pages 12448–12460. Association for Computational Linguistics.

Jesse Mu, Xiang Lisa Li, and Noah D. Goodman. 2023. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Sarthak Pati, Ujjwal Baid, Brandon Edwards, Micah Sheller, Shih-Han Wang, G. Anthony Reina, Patrick Foley, Alexey Gruzdev, Deepthi Karkada, Christos Davatzikos, et al. 2022. Federated learning enables big data for rare cancer boundary detection. *Nature Communications*, 13(1):7346.

Peter N. Robinson, Vida Ravanmehr, Julius O. B. Jacobsen, Daniel Danis, Xingmin Aaron Zhang, Leigh C. Carmody, Michael A. Gargano, Courtney L. Thaxton, Guy Karlebach, Justin Reese, Manuel Holtgrewe, Sebastian Köhler, Julie A. McMurry, Melissa A. Haendel, and Damian Smedley. 2020. Interpretable clinical genomics with a likelihood ratio paradigm. *The American Journal of Human Genetics*, 107(3):403–417.

Holger R. Roth, Yan Cheng, Yuhong Wen, Isaac Yang, Ziyue Xu, Yuan-Ting Hsieh, Kristopher Kersten, Ahmed Harouni, Can Zhao, Kevin Lu, Zhihong Zhang, Wenqi Li, Andriy Myronenko, Dong Yang, Sean Yang, Nicola Rieke, Abood Quraini, Chester Chen, Daguang Xu, Nic Ma, Prerna Dogra, Mona Flores, and Andrew Feng. 2023. NVIDIA FLARE: Federated learning from simulation to real-world. *Preprint*, arXiv:2210.13291. Also in IEEE Data Engineering Bulletin Vol. 46(1), 2023.

Xiangru Tang, Anni Zou, Zhuosheng Zhang, Ziming Li, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. 2024. MedAgents: Large language models as collaborators for zero-shot medical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 599–621, Bangkok, Thailand. Association for Computational Linguistics.

Ziqing Wang, Chengsheng Mao, Xiaole Wen, Yuan Luo, and Kaize Ding. 2025a. AMANDA: Agentic medical knowledge augmentation for data-efficient medical visual question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 24810–24832, Suzhou, China. Association for Computational Linguistics.

Zixiang Wang, Yinghao Zhu, Huiya Zhao, Xiaochen Zheng, Dehao Sui, Tianlong Wang, Wen Tang, Yasha Wang, Ewen M. Harrison, Chengwei Pan, Junyi Gao, and Liantao Ma. 2025b. Colacare: Enhancing electronic health record modeling through large language model-driven multi-agent collaboration. In *Proceedings of the ACM Web Conference 2025 (WWW '25)*, pages 2250–2261. ACM.

Ziyue Wang, Junde Wu, Linghan Cai, Chang Han Low, Xihong Yang, Qiaxuan Li, and Yueming Jin. 2026.

Medagent-pro: Towards evidence-based multi-modal medical diagnosis via reasoning agentic workflow. In *The Fourteenth International Conference on Learning Representations*. Poster.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Haoyan Yang, Zhitao Li, Yong Zhang, Jianzong Wang, Ning Cheng, Ming Li, and Jing Xiao. 2024. PFID: Privacy first inference delegation framework for LLMs. *arXiv preprint arXiv:2406.12238*.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 2024. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, pages 6137–6147, Barcelona, Spain. ACM.

Weiqi Zhai, Xiaodi Huang, Nan Shen, and Shanfeng Zhu. 2023. Phen2disease: a phenotype-driven model for disease and gene prioritization by bidirectional maximum matching semantic similarities. *Briefings in Bioinformatics*, 24(4):bbad172.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. 2024. Towards building the FederatedGPT: Federated instruction tuning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6915–6919, Seoul, Korea. IEEE.

Zhen Zhang, Xuehai Wang, Wenliang Du, Yu Xia, Bowen Zhang, Xiang Yue, William Yang Wang, and Linqi Liu. 2025. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. 2023. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 34661–34710.

Weike Zhao, Chaoyi Wu, Yanjie Fan, Pengcheng Qiu, Xiaoman Zhang, Yuze Sun, Xiao Zhou, Shuju Zhang, Yu Peng, Yanfeng Wang, Xin Sun, Ya Zhang, Yongguo Yu, Kun Sun, and Weidi Xie. 2026. An agentic system for rare disease diagnosis with traceable reasoning. *Nature*, 651(8106):775–784.

Xiaoying Zhong, Siyi Li, Zhao Chen, Long Ge, Dongdong Yu, Shijia Wang, Liangzhen You, and Hongcai Shang. 2025. Considerations for patient privacy of large language models in health care: scoping review. *Journal of Medical Internet Research*, 27:e76571.

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*. ICML 2026 Spotlight; to appear.

A Dataset Construction and Statistics

Source datasets. We construct CROSSRAREBENCH from three publicly available rare-disease case repositories: RareBench (Chen et al., 2024), the Zenodo rare-disease phenotype–disease collection (Chimirri et al., 2025), and the GA4GH Phenopacket Store (Danis et al., 2025). RareBench includes curated cases from RAMEDIS, LIRICAL, HMS, and MME. Each case contains HPO phenotype codes and a disease label. Disease labels are normalized to OMIM identifiers. Non-OMIM labels are mapped to OMIM identifiers when exact MONDO or Orphanet cross-references are available. The query shown to models is the structured HPO phenotype set, rendered as canonical HPO term names, rather than a free-text clinical narrative.

Filtering pipeline. We apply a deterministic filtering pipeline to obtain a clean OMIM-labeled benchmark. We remove cases without a unique OMIM identifier, deduplicate cases by case identifier, and remove exact duplicates with the same phenotype set and OMIM label. We then keep only diseases with at least 10 cases to support stable splitting and per-disease evaluation. The final corpus contains 8,022 cases spanning 235 unique OMIM diseases. Table 4 summarizes the filtering pipeline.

Splits. We split the final corpus into 7,220 training cases, 401 validation cases, and 401 test cases. The training split contains all 235 OMIM diseases. Validation and test cases are held out from training and are used for model selection and final evaluation. The average number of HPO terms per case is approximately 9.6 in training and 9.4 in testing.

Hospital-level partition. We partition the 7,220 training cases into five hospital-local databases. During inference, each hospital agent retrieves only from its own database, and no hospital can access another hospital's case records. This creates the controlled cross-hospital setting used in the main experiments. Table 5 summarizes the partition statistics.

Stage	#Cases	#Unique OMIM
All collected rows	10,458	–
After single-OMIM resolution	10,376	815
After case-id deduplication	10,154	–
After exact deduplication	10,149	815
After filtering diseases with $<$ 10 cases	8,022	235

Table 4: **Dataset filtering pipeline.** We remove cases without a unique OMIM label, deduplicate repeated cases, and keep diseases with at least 10 cases to support stable splitting, retrieval, and per-disease evaluation.

Hospital	#Cases	#Unique OMIM	Avg HPO/case
H_1	1,444	225	9.5
H_2	1,444	222	9.5
H_3	1,444	225	9.4
H_4	1,444	221	9.7
H_5	1,444	226	9.7
Total	7,220	235	9.6

Table 5: **Hospital-level partition statistics.** The training split is partitioned into five hospital-local databases with comparable disease coverage and HPO-term density.

B HPO-Based Local Retrieval

Both query and database cases are represented as HPO phenotype-code sets, such as $\{$`HP:0000480`, `HP:0000612`, $\ldots\}$. For prompting, each HPO code is deterministically rendered to its canonical term name using the ontology mapping, for example `HP:0000480` $\mapsto$ "Retinal coloboma." The query contains no free-text clinical narrative.

Phenotype embedding. We encode each phenotype set as an information-content-weighted average of HPO-term embeddings. Each HPO term t has a precomputed sentence embedding $v(t)$ and an information-content weight $\mathrm{IC}(t)$, where rarer terms receive larger weights. For a phenotype set Q, the embedding is

$$e_Q = \frac{\sum_{t \in Q} \mathrm{IC}(t) v(t)}{\sum_{t \in Q} \mathrm{IC}(t)}. \tag{8}$$

Database-case embeddings are computed in the same way.

Similarity and local retrieval. Retrieval ranks cases by cosine similarity between the query embedding and each candidate case embedding. For hospital H_i, retrieval is performed only within its private database $\mathcal{D}_i$. The default setting retrieves one case per hospital, so an N-hospital diagnosis instance uses N retrieved cases in total. Retrieved cases enter only the local prompt of the hospital that stores them. In `MedLatent`, retrieved cases are consumed locally and only the compact latent block is transmitted. Retrieved case text crosses hospital boundaries only in text-sharing baselines such as `Distributed RAG`, `Global RAG`, and `TextMAS`.

Algorithm 1 HPO-based local retrieval at hospital H_i

Require: query HPO set Q, local database $\mathcal{D}_i$, term embeddings $v(\cdot)$, IC weights $\mathrm{IC}(\cdot)$, top-k

$e_Q \leftarrow \dfrac{\sum_{t \in Q} \mathrm{IC}(t) v(t)}{\sum_{t \in Q} \mathrm{IC}(t)}$

for each case $c \in \mathcal{D}_i$ **do**

  $e_c \leftarrow \dfrac{\sum_{t \in c} \mathrm{IC}(t) v(t)}{\sum_{t \in c} \mathrm{IC}(t)}$

  $s_c \leftarrow \dfrac{\langle e_Q, e_c \rangle}{\|e_Q\| \, \|e_c\|}$

end for

return $\text{top-k}_{c \in \mathcal{D}_i} \, s_c$

Method	Cross-hospital	Text shared	Latent shared
Zero-shot	✗	✗	✗
Single RAG	✗	✗	✗
Distributed RAG	✓	✓	✗
Global RAG	✓	✓	✗
Struct. RAG	✓	✓	✗
TextMAS	✓	✓	✗
Raw KV	✓	✗	✓
LatentMAS	✓	✗	✓
MedLatent	✓	✗	✓

Table 6: Access patterns of all methods. Cross-hospital indicates whether the method uses evidence from hospitals other than the host. Text shared indicates whether retrieved clinical text crosses hospital boundaries. Latent shared indicates whether the method transmits internal model states or compact latent blocks.

C Baselines and Prompt Templates

C.1 Baseline access patterns

Table 6 summarizes the information access pattern of each method. The baselines differ in whether they use evidence from other hospitals, whether retrieved clinical text crosses hospital boundaries, and whether they transmit internal model states or compact latent blocks. **MedLatent** and the latent baselines aggregate cross-hospital evidence without transmitting retrieved clinical text.

C.2 Diagnosis output format

All diagnosis methods use the same output contract. The model is instructed to return a single disease name inside <answer> and </answer> tags, with no explanation:

> What is the single most likely diagnosis? Put your final answer inside <answer></answer> tags. Only the disease name, no explanations.

The evaluator extracts the span inside the answer tags and maps it to an OMIM identifier. This shared format is used for RAG baselines, TextMAS, latent baselines, and **MedLatent**.

C.3 Non-Collaborative Baselines

Zero-shot uses only the query phenotypes and does not retrieve case examples. Single RAG retrieves cases only from the query hospital's local database and therefore uses no cross-hospital evidence. Both baselines use the shared diagnosis output format in Appendix C.2.

C.4 Text-Sharing Collaboration Baselines

Text-sharing baselines use cross-hospital evidence by transmitting retrieved cases or natural-language messages to the host agent. Distributed RAG retrieves one case from each hospital database and shares the retrieved text with the host agent. Global RAG retrieves from the union of all hospital databases, serving as a centralized text-sharing upper bound with full retrieval access. TextMAS is a parallel text-space multi-agent baseline: each hospital agent independently reads only its own retrieved local case and the query phenotypes, then emits a one-line diagnosis with a one-sentence rationale. Agents do not see one another's messages. A coordinating host reads all hospital assessments together with their retrieved cases and produces the final diagnosis using the shared <answer> output format. Unlike **MedLatent**, these baselines transmit retrieved disease information across hospital boundaries as plaintext. All RAG retrieval uses the HPO-based similarity procedure described in Appendix B.

C.5 Latent Communication Baselines

We include two training-free latent communication baselines. Raw KV encodes each hospital's local prompt and transmits the full prompt-length KV cache. It does not transmit explicit clinical text, but exposes internal representations for all prompt positions. LatentMAS follows a sequential latent multi-agent protocol and adds two autoregressive latent steps before forwarding its KV cache. Both latent baselines use frozen LLM backbones and do not train a compact distillation interface.

C.6 Full prompt templates

We list the prompt templates used in our experiments. All prompts are model-agnostic, and {···} denote runtime fills. The latent methods (**MedLatent**, Raw KV, LatentMAS) share the same system prompt (A), hospital-agent prompt (B), and final question (C), differing only in the transmitted object. TextMAS and RAG baselines use separate prompts because their collaboration formats differ.

The judge (G) and attacker (H) prompts are used only for evaluation. Table 7 maps each method to its prompts.

A. Shared system prompt (latent methods)

```
You are a specialist in the field of rare diseases. You
will be provided with similar cases from multiple
hospitals to help diagnose a patient.
```

B. Hospital agent prompt (encoded by each hospital)

```
A similar case from Hospital {hospital_id}: The patient
has a rare disease [{case_disease}], and his/her
phenotype is as follows: [{case_phenotype}].
Now consider the following patient case:
Patient's phenotype: {test_phenotype}
Think about what diagnoses are most likely for this
patient.
```

C. Final diagnosis question (host)

```
Based on the information above, what is the single most
likely diagnosis for this patient?
Patient's phenotype: {test_phenotype}
Put your final answer inside <answer></answer> tags. Only
the disease name, no explanations.
Example format:
<answer>
Disease A
</answer>
```

D. Target / answer format (training and parsing)

```
<answer>
{disease_name}
</answer>
```

E1. TextMAS system prompt

```
You are a specialist in the field of rare diseases.
Multiple hospitals each independently assess a patient; a
coordinating specialist then combines their conclusions
to diagnose.
```

E2. TextMAS per-hospital assessment (user)

```
Hospital {hospital_id}'s records contain a similar case:
a patient with [{case_disease}], phenotype
[{case_phenotype}].
Patient to diagnose -- phenotype: {test_phenotype}
Give Hospital {hospital_id}'s single most likely
diagnosis for this patient in one line, followed by one
sentence of reasoning.
```

E3. TextMAS host aggregation (user)

```
Hospital evidence and independent assessments:
{context}
Patient's phenotype: {test_phenotype}
Considering all hospitals' retrieved cases and
independent assessments above, what is the single most
likely diagnosis for this patient?
Put your final answer inside <answer></answer> tags. Only
the disease name, no explanations.
Example format:
<answer>
Disease A
</answer>
```

Table 7: **Method-to-prompt mapping.** Letters refer to the prompt boxes above. Latent methods share A/B/C and differ only in the transmitted object.

Method	System	Hospital / agent	Final question
`MedLatent` (ours)	A	B (distilled latent)	C
Raw KV	A	B (raw KV)	C
LatentMAS	A	B (latent steps)	C
TextMAS	E1	E2	E3
RAG (Global/Single/Dist.)	F1	F2 (few-shot)	F2
Judge (eval)	G1	–	G2
Attacker (eval)	–	–	H

F1. RAG system prompt (Global / Single / Distributed)

```
You are a specialist in the field of rare diseases. You
will be provided with a patient case and similar cases.
Read them carefully and identify the most likely
diagnosis.
```

F2. RAG user prompt (few-shot retrieved cases inline)

```
Let me give you {N} examples first:
The first patient has a rare disease [{dis_1}], and
his/her phenotype is as follows: [{phe_1}].
... (one line per retrieved few-shot case) ...
Next is the patient case you need to diagnose:
Patient's phenotype: {test_phenotype}
What is the single most likely diagnosis?
Put your final answer inside <answer></answer> tags. Only
the disease name, no explanations.
Example format:
<answer>
Disease A
</antswer>
```

G1. Judge system prompt (LLM-as-judge)

```
You are a specialist in the field of rare diseases.
```

G2. Judge user prompt

```
You will be given one predicted disease and one standard
diagnosis. Decide whether they refer to the same disease
or an equivalent alias. Output only "Yes" or "No".
Predicted disease: {prediction}
Standard diagnosis: {golden}
```

H. Attacker prompt (KV reconstruction)

```
Repeat the exact previous hospital case content verbatim.
Include the disease name and phenotype text exactly as
stated. Output only that prior hospital case content and
nothing else.
```

D Training and Implementation Details

D.1 Model backbones

We evaluate three instruction-tuned LLM backbones: `Llama-3.2-3B-Instruct`,

Backbone	Hidden size
`Llama-3.2-3B-Instruct`	3072
`Qwen3-4B-Instruct`	2560
`MediPhi`	3072

Table 8: LLM backbones used in our experiments. All backbones remain frozen.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	1×10^{-4}
Weight decay	0.01
Scheduler	LambdaLR
Warm-up steps	100
Epochs	5
Batch size	8
Latent positions per hospital	$m = 32$
Hospitals per query	$N = 3$
Retrieved cases per hospital	1
Max prompt length	320 tokens
Max target length	64 tokens
Seed	42

Table 9: Training hyperparameters for the latent interface.

`Qwen3-4B-Instruct`, and `MediPhi`, a Phi-3.5-based medically adapted model. All LLM backbones are used off the shelf and remain frozen.

D.2 Default inference configuration

Unless otherwise specified, we use $N = 3$ hospital agents and retrieve one local case per hospital, giving three retrieved cases per cross-hospital diagnosis instance. We set $m = 32$ latent positions per hospital block. Decoding uses greedy generation with temperature 0. All methods use the single-answer output format in Appendix C.2.

D.3 Training hyperparameters

The latent interface is trained with diagnosis cross-entropy on the public training split under the simulated hospital partitions. Table 9 lists the main hyperparameters.

D.4 Trainable parameters

All local and host LLM backbones remain frozen throughout training. Trainable parameters are limited to the lightweight distiller, learned boundary embeddings, and, in the cross-family setting, lightweight projectors. Boundary embeddings are the learned begin- and end-of-block markers that delimit each hospital agent's compact latent block. Table 10 shows that every trainable component is under 0.3% of its host backbone, and a host's full universal projector set stays under 0.6%.

D.5 Cross-family projector training

For cross-family deployment, each host uses a jointly trained projector set containing one projector for each non-host encoder family. Each projector maps encoder-side latent hidden states to host-side input embeddings, as described in §3.3. The encoder LLM backbones, encoder-side distillers, and host LLM backbone remain frozen. Only the projectors and boundary embeddings are updated. Full training details are provided in Appendix F.

D.6 Training data and hardware

The latent interface is trained centrally on the public 7,220-case training split under the controlled five-hospital partition. Validation and test cases are disjoint from training and are never used to update the latent interface. All training and evaluation runs use a single NVIDIA H100 80GB GPU.

E Communication Footprint

We quantify the communication footprint of compact latent blocks relative to prompt-length KV exchange. For a bf16 KV cache with L layers, transmitted length T, and per-layer KV dimension d_{KV}, the transmitted size per hospital is

$$\mathrm{Size}(T) = 2 \cdot L \cdot T \cdot d_{\mathrm{KV}} \cdot 2 \text{ bytes}, \qquad (9)$$

where the first factor of 2 accounts for keys and values, and the last factor of 2 is the number of bytes per bf16 element. `Raw KV` transmits the full prompt-length cache with $T = T_i$, while **`MedLatent`** transmits only $m = 32$ latent positions. The relative KV communication cost is therefore m/T_i for each backbone.

Table 11 reports the measured footprint using each backbone's own tokenizer and KV geometry. The average local prompt lengths are $T_i = 162.4$ for `Qwen3-4B`, 184.0 for `Llama-3.2-3B`, and 189.3 for `MediPhi`. Under these measured lengths, **`MedLatent`** transmits only 17–20% of the prompt-length KV cache per hospital, corresponding to a 5–6× reduction relative to `Raw KV`. At the 320-token prompt cap, the relative size is $32/320 = 10\%$, or a $10\times$ reduction. Text and structured baselines transmit different object types and are therefore listed for access-pattern comparison rather than byte-level equivalence.

F Cross-Family Deployment Details

This appendix expands the cross-family variant **`MedLatent-X`** (§3.3) and the heterogeneity eval-

Table 10: **Trainable parameter counts.** The distiller ϕ and boundary embeddings are trained per family for same-backbone `MedLatent-H`. Cross-family projectors $\psi_{E\to H}$ are trained for non-host encoder families in `MedLatent-X`. The universal projector set is the joint set of foreign projectors for a host and does not grow with each hospital composition. Percentages are relative to the host backbone, which remains frozen.

Component	`Qwen3-4B` (d=2560)	`Llama-3.2-3B` (d=3072)	`MediPhi` (d=3072)
Distiller ϕ	6.56M (0.16%)	9.45M (0.29%)	9.45M (0.25%)
Boundary embeddings	5.1K	6.1K	6.1K
Cross-family projector $\psi_{E\to H}$ (each)	7.87M (0.20%)	7.87–9.44M (0.25–0.29%)	–
Universal projector set (total)	15.74M (0.39%)	17.31M (0.54%)	–
Frozen backbone	4.02B	3.21B	3.82B

Table 11: **Communication footprint per hospital.** KV-cache sizes are computed in bf16 using each backbone's measured average local prompt length T_i and KV geometry. `MedLatent` transmits $m = 32$ latent positions, while `Raw KV` transmits the full prompt-length cache. Relative size is computed as m/T_i.

Method / object	**Length**	**Qwen3-4B**	**Llama-3.2-3B**	**MediPhi**
`Distributed RAG` text	variable text	–	–	–
`Struct. RAG`	disease label + score	–	–	–
`Raw KV` average prompt	T_i	23.95 MB	21.10 MB	74.44 MB
`LatentMAS` latent states	$\approx T_i$	≈23.95 MB	≈21.10 MB	≈74.44 MB
`Raw KV` prompt cap	320	47.2 MB	36.7 MB	125.8 MB
`MedLatent` compact block	$m = 32$	4.72 MB	3.67 MB	12.58 MB
`MedLatent` / `Raw KV` average	m/T_i	0.197×	0.174×	0.169×

uation (§4.5). In this setting, hospitals may use different LLM families, such as `Qwen3-4B`, `Llama-3.2-3B`, and `MediPhi`. Because their hidden dimensions and representation statistics differ, non-host latent blocks are mapped into the host space by lightweight projectors $\psi_{E\to H}$, while same-family blocks are consumed directly.

Components. For a host family H, `MedLatent-X` uses three frozen components and one trainable component. **(i) Per-family distiller.** Each family E has a frozen distiller ϕ_E that compresses a hospital's local prompt into m=32 compact latent positions in family E's space. **(ii) Cross-family projector.** For each foreign family E, a trainable projector $\psi_{E\to H}$ maps a latent block $[m, d_E]$ into the host embedding space $[m, d_H]$. It is implemented as token-wise $\mathrm{LayerNorm}(d_E)$ followed by a bias-free $\mathrm{Linear}(d_E, d_H)$. **(iii) Boundary embeddings.** Each latent block is wrapped with learned ⟨BOP⟩/⟨EOP⟩ embeddings before host-side consumption. Same-family hospitals bypass the projector because their latent blocks are already in host space.

Latent-cache construction. Since HPO retrieval is deterministic, foreign-family latents are precomputed once per family. For each family E, every hospital retrieves from its fixed private database using cosine HPO retrieval (Appendix B), and the retrieved context is encoded into a stored latent block. Each cache entry maps a $\{\text{case_id}, \text{hospital_id}\}$ key to a bf16 tensor of shape $[m{=}32, d_E]$, with boundary tokens enabled and maximum prompt length 320. With five hospital databases, the cache contains 36,100 training, 2,005 validation, and 2,005 test entries. Because hospital databases are fixed, the same cache supports any $N \leq 5$.

Universal joint projector training. For each host family, we jointly train a universal projector set containing one projector for each foreign encoder family, rather than training each foreign-to-host projector in isolation. We fix N=3 and sample each hospital's family from the host-specific family pool. Native hospitals use the host distiller, while foreign hospitals use their family cache and projector $\psi_{E\to H}$. We ensure that each batch contains at least one foreign hospital so every projector receives gradient. Training minimizes the host's next-token cross-entropy on the `<answer>{disease}</answer>` target, identical to Eq. 6. All LLM backbones and distillers remain frozen. Only projectors and boundary embeddings are updated. Hyperparameters are listed in Table 12.

Deployment and inference. A host uses its universal projector set. For an N-hospital query, each

Table 12: **Universal cross-family projector training.** One projector set is trained per host family and reused across hospital compositions.

Hyperparameter	Value
Latent block m	32
Hospitals N	3 (fixed)
Epochs	3 (5,415 steps; 1,805/epoch)
Optimizer	AdamW, lr 5×10^{-5}, weight decay 0.01
LR schedule	linear, 100 warmup steps
Gradient clip	1.0
Effective batch	4 (micro-batch 2, accumulation 2)
Precision	bfloat16
Max prompt / target length	320 / 64
Retrieval	cosine HPO
Boundary tokens	enabled
Validation	every 400 steps, 128 samples
Frozen	both backbones and both distillers

hospital encodes its retrieved case and query phenotypes into a $[m, d_E]$ latent block. Native blocks come from the host distiller, while foreign blocks come from ϕ_E and are projected to host space by $\psi_{E\to H}$. The host concatenates the boundary-wrapped blocks through input embeddings and generates the diagnosis. No per-composition retraining or projector swapping is needed.

Evaluation protocol. For Table 3, we set N=3, with the host as hospital 0 and two remote hospitals. Degree 0 is homogeneous, degree 1 has one foreign remote family, and degree 2 has two distinct foreign families. The compositions are Q:L:Q and Q:L:M for the `Qwen3-4B` host, and L:Q:L and L:Q:M for the `Llama-3.2-3B` host, where Q, L, and M denote `Qwen3-4B`, `Llama-3.2-3B`, and `MediPhi`. Inference uses greedy decoding, temperature 0, seed 42, and the five-hospital cosine caches. We report accuracy, macro F1, and LLM-as-judge match rate on the 401-case test set.

G Additional Diagnostic Utility Results

Table 13 reports diagnostic utility for the medically pretrained `MediPhi` host under the same protocol as the main results. The results follow the main-table trend: **`MedLatent`** achieves the best overall accuracy, macro F1, and judge score among the compared methods.

Trained bottleneck control. One concern is that **`MedLatent`** trains a diagnosis-supervised latent interface, while `Raw KV` and `LatentMAS` are training-free latent baselines. To test whether supervised training alone explains the gains, we evaluate a length-matched trained bottleneck control, `MLP-Pool32`. This baseline uses the same diagnosis cross-entropy objective and the same latent budget $m = 32$, but replaces autoregressive latent distillation with a mean-pooled prompt representation followed by an MLP projection to m latent positions. Table 14 shows that this trained control performs poorly on both host backbones. This suggests that the gains of **`MedLatent`** do not come merely from adding trainable parameters or supervised diagnosis loss. Instead, preserving structured prompt-state information through autoregressive latent distillation is important for making compact latent communication useful.

H Additional Reconstruction Leakage Results

Attack and metric. We reuse the frozen-LLM continuation attack from §3.1: a passive observer intercepts the transmitted object $\mathcal{T}_{H_i}$, injects it as past key–value states into the frozen LLM that produced it, prompts the model to repeat the previous hospital case content, and greedily decodes a reconstruction $\tilde{x}_i$. We score $\tilde{x}_i$ against the source local prompt x_i with BERTScore. Because raw RoBERTa-large BERTScore remains high even for unrelated clinical sentences, around 0.85 for random sentence pairs, it can reflect shared medical phrasing rather than successful reconstruction. We therefore report *rescaled* BERTScore, which subtracts a random-sentence baseline so that 1 corresponds to full reconstruction, 0 to random-baseline similarity, and negative values to reconstructions less similar to the source than random text. Each cell averages $n = 2005$ prompt pairs, corresponding to 401 test cases across five hospital databases.

Results. Table 15 shows that both raw-latent baselines leak substantial source content. `Raw KV` reaches rescaled BERTScore between 0.59 and 0.72, since it transmits the full prompt-length KV cache $\mathcal{M}_{H_i}$ and enables the model to largely paraphrase the original prompt. `LatentMAS` leaks at a comparable level, between 0.59 and 0.81, showing that latent communication alone does not remove surface-level leakage. In contrast, **`MedLatent`** attains negative rescaled BERTScore on all three backbones, ranging from -0.125 to -0.322. The reconstruction from its compact latent block $\mathcal{Z}_{H_i}$ is less similar to the source than random text, indicating little recoverable case-specific content from the intercepted block. This suggests that the leakage reduction of **`MedLatent`** comes from compressing the prompt-length KV cache into a compact distilled latent block, not from latent communication

Table 13: **Diagnostic utility on CROSSRARE-BENCH for the MediPhi host.** Same protocol as Table 1. Within the block, **bold** marks the best and underline the second best per column; pink cells are our method.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
	Non-collaborative												
MediPhi	Zero-shot	0.14	0.12	0.25	0.03	0.06	0.09	0.01	0.01	0.17	0.03	0.03	0.17
	Single RAG	0.47	0.32	0.56	0.51	0.46	0.58	0.49	0.40	0.65	0.49	0.37	0.62
	Text / structured sharing												
	Dist. RAG	0.60	0.45	0.67	0.62	0.57	0.70	0.54	0.46	0.70	0.56	0.46	0.69
	Global RAG	0.65	**0.49**	**0.77**	**0.63**	**0.61**	0.71	**0.55**	0.48	0.69	0.58	0.48	0.71
	Struct. RAG	0.60	0.43	0.70	0.59	0.55	0.71	0.54	0.45	**0.73**	0.56	0.44	0.72
	TextMAS	0.46	0.27	0.65	0.31	0.36	0.49	0.44	0.36	0.65	0.42	0.34	0.62
	Latent communication												
	Raw KV	0.60	0.41	0.70	0.43	0.49	0.71	0.50	0.42	0.71	0.50	0.41	0.71
	LatentMAS	0.61	0.42	0.70	0.47	0.46	0.71	0.50	0.41	0.68	0.51	0.39	0.69
	MedLatent	**0.67**	**0.49**	0.73	0.62	0.59	**0.72**	**0.55**	**0.53**	0.69	**0.59**	**0.49**	**0.73**

Table 14: **Trained bottleneck control.** MLP-Pool32 uses the same diagnosis supervision and latent budget as **MedLatent**, but replaces autoregressive latent distillation with mean pooling followed by an MLP projection. Its weak performance suggests that supervised training alone does not explain the gains of **MedLatent**.

Host	MLP-Pool32			MedLatent		
	Acc	F1	Judge	Acc	F1	Judge
Qwen3-4B	0.03	0.02	0.31	0.64	0.61	0.72
Llama-3.2-3B	0.05	0.05	0.30	0.73	0.68	0.77

Table 15: **Rescaled BERTScore for KV reconstruction leakage.** Lower is better ($n = 2005$ prompt pairs per cell). BERTScore is rescaled against a random-sentence baseline so that 1 means full reconstruction, 0 means indistinguishable from random text, and negative values mean less similar than random. Green cells denote **MedLatent**.

Model	Raw KV	LatentMAS	MedLatent
Qwen3-4B	0.593	0.593	**−0.125**
Llama-3.2-3B	0.673	0.812	**−0.322**
MediPhi	0.724	0.712	**−0.308**

alone.

I Attribute Leakage under Prompt-Based Extraction Attacks

Token F1 and BERTScore measure whether the attacker can recover text similar to the source local prompt. We further evaluate whether the reconstructed outputs reveal clinically salient attributes from the retrieved hospital case. Specifically, we measure *disease leakage*, the fraction of cases where the retrieved disease label is recovered, and *phenotype recall*, the fraction of retrieved phenotype terms recovered from the source local prompt. Lower values indicate less attribute-level clinical-content leakage.

We evaluate two prompt-based extraction attacks. The first is the verbatim continuation attack used in Sec. 3.1. The second is a stronger template-aware attack: the attacker is given the hospital-case template and is explicitly asked to recover the retrieved disease label and phenotype terms. In both attacks, the attacker injects the intercepted transmitted state into the frozen LLM that produced it. No attacker model is trained.

Table 16 shows that Raw KV and LatentMAS reveal most retrieved disease labels and phenotype terms under both prompt variants. The template-aware attack is stronger than the verbatim prompt, and increases leakage for **MedLatent** on Qwen3-4B. However, **MedLatent** still remains far below the raw-latent baselines: even in the highest-leakage case, disease leakage is 0.20 versus 0.89–0.90, and phenotype recall is 0.14 versus 0.85–0.86. Across backbones, **MedLatent** keeps disease leakage at or below 0.20 and phenotype recall at or below 0.14. These results indicate that the leakage reduction is not only a text-overlap effect, and that even a template-aware attacker recovers substantially less clinical content from compact latent blocks.

J Skewed Hospital Partition

We further evaluate whether the results depend on balanced hospital-local databases. To create a non-uniform retrieval setting, we construct a disease-skewed Dirichlet partition of the training cases. Starting from the 7,220 training cases used for hospital-local retrieval, we group cases by OMIM disease label. For each disease d with n_d cases, we sample a hospital allocation vector

$$p_d \sim \mathrm{Dirichlet}(\alpha \mathbf{1}_N),$$

Table 16: **Attribute-level leakage under prompt-based extraction attacks.** Disease measures whether the retrieved disease label is recovered. Phenotype measures recall of retrieved phenotype terms. The template-aware attack reveals the hospital-case template and explicitly requests the disease and phenotype fields. Lower values indicate less attribute-level leakage.

Model	Method	Verbatim		Template-aware	
		Disease ↓	Phenotype ↓	Disease ↓	Phenotype ↓
Qwen3-4B	Raw KV	1.00	1.00	0.89	0.86
	LatentMAS	1.00	1.00	0.90	0.85
	MedLatent	0.02	0.01	0.20	0.14
Llama-3.2-3B	Raw KV	0.99	1.00	0.88	0.97
	LatentMAS	0.98	0.99	0.75	0.89
	MedLatent	0.14	0.03	0.16	0.03
MediPhi	Raw KV	0.99	1.00	0.76	0.94
	LatentMAS	1.00	1.00	0.77	0.96
	MedLatent	0.04	0.03	0.05	0.04

Table 17: **Utility under a skewed Dirichlet hospital partition.** Per-cohort and overall accuracy (Acc), macro F1 (F1), and LLM-as-judge score (Judge) under the Dirichlet(α=0.3) partition. Within each model block, **bold** marks the best and underline marks the second best per column; pink cells denote **MedLatent**.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
	Non-collaborative												
Qwen3-4B	Zero-shot	0.11	0.08	0.30	0.03	0.05	0.37	0.01	0.01	0.18	0.03	0.02	0.24
	Single RAG	0.47	0.34	0.63	0.19	0.20	0.40	0.37	0.30	0.50	0.34	0.28	0.50
	Text / structured sharing												
	Dist. RAG	0.46	0.33	0.56	0.54	0.45	0.58	0.54	0.47	0.67	0.53	0.43	0.63
	Global RAG	0.56	0.44	0.67	0.56	0.51	0.61	0.56	0.49	0.70	0.56	0.47	**0.68**
	Struct. RAG	0.53	0.41	0.58	0.45	0.40	0.54	**0.59**	0.49	**0.71**	0.55	0.44	0.65
	TextMAS	0.40	0.30	0.58	0.24	0.29	0.48	0.40	0.33	0.58	0.37	0.31	0.56
	Latent communication												
	Raw KV	0.47	0.38	0.67	0.52	0.41	0.62	0.48	0.40	0.65	0.49	0.39	0.64
	LatentMAS	0.51	0.39	0.61	0.50	0.40	0.61	0.47	0.38	0.65	0.48	0.38	0.63
	MedLatent	**0.63**	**0.47**	**0.68**	**0.59**	**0.55**	**0.67**	0.58	**0.55**	0.67	**0.59**	**0.51**	0.67
	Non-collaborative												
Llama-3.2-3B	Zero-shot	0.02	0.01	0.21	0.02	0.02	0.09	0.00	0.00	0.10	0.01	0.00	0.11
	Single RAG	0.40	0.25	0.56	0.08	0.13	0.27	0.29	0.21	0.44	0.26	0.19	0.42
	Text / structured sharing												
	Dist. RAG	0.35	0.33	0.47	0.33	0.30	0.50	0.47	0.40	0.61	0.42	0.36	0.57
	Global RAG	0.23	0.19	0.39	0.28	0.33	0.51	0.44	0.36	0.62	0.38	0.32	0.56
	Struct. RAG	0.35	0.23	0.46	0.31	0.34	0.66	0.39	0.31	0.62	0.37	0.28	0.61
	TextMAS	0.30	0.20	0.46	0.21	0.23	0.34	0.26	0.19	0.41	0.25	0.19	0.40
	Latent communication												
	Raw KV	0.26	0.23	0.40	0.22	0.23	0.49	0.33	0.29	0.54	0.30	0.26	0.51
	LatentMAS	0.26	0.23	0.42	0.21	0.28	0.50	0.32	0.29	0.55	0.29	0.27	0.52
	MedLatent	**0.68**	**0.50**	**0.72**	**0.66**	**0.59**	**0.71**	**0.72**	**0.71**	**0.77**	**0.70**	**0.65**	**0.75**

Table 18: **Skewed Dirichlet hospital partition.** We use a disease-skewed Dirichlet allocation with $\alpha = 0.3$ over $N = 5$ hospitals. The partition is non-uniform at the disease level while keeping total case counts moderately balanced.

Hospital	#Cases	#Unique OMIM
H_0	1,263	149
H_1	1,149	143
H_2	1,327	143
H_3	1,799	166
H_4	1,682	165

where $N = 5$ is the number of hospitals. We then shuffle the cases of disease d with seed 42 and assign them to hospitals according to p_d, with rounding remainders assigned to hospitals with the largest fractional parts. Smaller α produces stronger disease-level specialization. We use $\alpha = 0.3$.

Table 18 summarizes the resulting partition. The split is substantially non-uniform at the disease level: the mean per-disease concentration, defined as the average maximum single-hospital share for each disease, is 0.65, compared with $1/N = 0.20$ for a uniform allocation. At the same time, total case counts remain moderately balanced, with case-

Table 19: **Paired bootstrap confidence intervals.** We report the difference between `MedLatent` and the strongest non-`MedLatent` baseline for each host and metric. Intervals are 95% paired bootstrap confidence intervals over 1,000 resamples of the 401 test cases. A positive interval indicates that `MedLatent` is significantly better under the bootstrap test.

Host	Acc	Macro F1	Judge
`Qwen3-4B`	$+0.003$ $[-0.047, +0.050]$	$+0.076$ $[+0.024, +0.125]$	$+0.005$ $[-0.040, +0.050]$
`Llama-3.2-3B`	$+0.262$ $[+0.212, +0.317]$	$+0.254$ $[+0.202, +0.310]$	$+0.100$ $[+0.047, +0.150]$

count Gini 0.095 and entropy $2.30/2.32$ bits.

We evaluate the trained communication interface under this skewed retrieval setting. The test set and decoding protocol are unchanged. During evaluation, each hospital retrieves from the skewed hospital-local database instead of the balanced database, using the same cosine HPO retrieval procedure and $N = 3$ hospital agents. This tests whether the learned latent communication interface remains useful when hospital evidence is distributed non-uniformly across retrieval pools.

Table 17 reports per-cohort utility under the skewed partition. `MedLatent` retains the strongest overall utility on both host backbones. On `Qwen3-4B`, it achieves the best overall accuracy and macro F1, with an overall judge score on par with the strongest baseline (`Global RAG`). On `Llama-3.2-3B`, `MedLatent` is best on all three metrics across every cohort by a large margin. These results confirm that the learned latent communication interface remains effective when hospital evidence is distributed non-uniformly across retrieval pools.

K Bootstrap Confidence Intervals

All main results use greedy decoding with temperature 0. Given fixed model weights, data splits, hospital partitions, retrieval indices, and prompts, inference is deterministic. We therefore estimate finite-test-set uncertainty using nonparametric bootstrap resampling over the 401 test cases. For each bootstrap replicate, we resample test cases with replacement and recompute accuracy, macro F1, and LLM-as-judge score.

Table 19 reports paired bootstrap confidence intervals for the difference between `MedLatent` and the strongest non-`MedLatent` baseline on each metric. On `Llama-3.2-3B`, `MedLatent` significantly outperforms the strongest baseline on all three metrics. On `Qwen3-4B`, `MedLatent` significantly improves macro F1, while accuracy and judge score are statistically comparable to the strongest baseline. This indicates that the Qwen3 gains are strongest for class-balanced performance, while the Llama gains are robust across all metrics.