LatentDx: LATENT MULTI-AGENT COMMUNICATION FOR CROSS-HOSPITAL RARE-DISEASE DIAGNOSIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Rare diseases affect approximately 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 LatentDx, a latent multi-agent communication framework in which hospital agents keep private clinical records and retrieved cases local, and send compact latent representations to a host agent for rare-disease diagnosis. In the same-backbone setting, LatentDx-H transmits distilled KV blocks. In the cross-family setting, LatentDx-X transmits compact hidden-state sequences for projection into host input embeddings. On CROSSRARE-BENCH, a self-built large-scale rare-disease benchmark with hospital-level partitions, LatentDx supports strong cross-hospital diagnostic performance, with evaluations in both settings also showing reduced reconstructable clinical content relative to raw-latent communication baselines. Code and data are available at `https://anonymous.4open.science/r/LatentDx/`.

1 INTRODUCTION

Rare diseases collectively encompass over 7,000 conditions and affect approximately 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.*

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). For rare diseases, DeepRare and RareAgents show that similar-case retrieval can improve diagnostic performance (Zhao et al., 2026; Chen et al., 2026a). 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 diagnostic 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?*

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

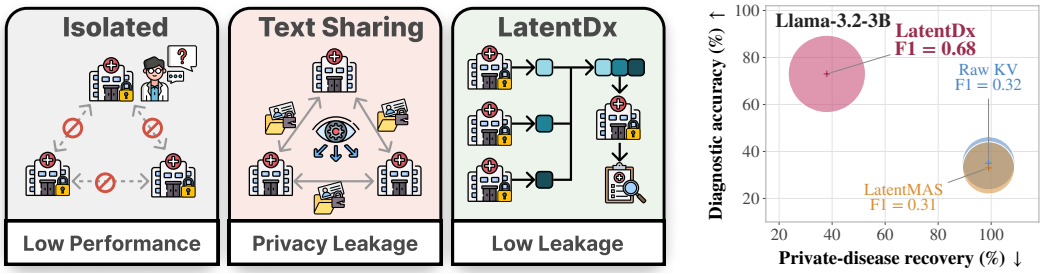


Figure 1: **Motivation and utility–leakage profile of `LatentDx`. Left:** Isolated diagnosis cannot use distributed evidence. Text sharing exposes retrieved records. `LatentDx` exchanges compact latent blocks while keeping clinical text local. **Right:** Overall diagnostic accuracy versus retrieved-disease recovery for latent communication methods on CROSSRARE-BENCH in the same-backbone `Llama-3.2-3B` setting (Tables 1 and 3). Bubble area is proportional to macro F1. Recovery is measured by learned inversion.

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., 2026; 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; Luo et al., 2026). Thus, cross-hospital latent communication should be evaluated not only for diagnostic utility, but also for empirical leakage from transmitted latent states.

To address these two challenges, we introduce `LatentDx`, 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 block, rather than retrieved text or raw model states, to a host agent for diagnosis. In both same-backbone and cross-family evaluations, compact latent representations reduce how much clinical content is reconstructable compared with raw model states. The same framework extends to settings where hospitals use different LLM backbones by aligning local latent states to the host agent's input-embedding 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 `LatentDx` as `LatentDx-H` for same-backbone hospital agents through latent KV distillation, and `LatentDx-X` for cross-family agents through latent alignment.
- **Benchmark.** We build CROSSRARE-BENCH, a large-scale cross-hospital rare-disease diagnosis benchmark with hospital-level partitions for jointly evaluating diagnostic utility and empirical leakage in inference-time collaboration under text-sharing constraints.
- **Evaluation.** We evaluate diagnostic utility and empirical clinical-content leakage in both settings. In the same-backbone setting, `LatentDx-H` achieves competitive diagnostic utility relative to supervised plaintext-sharing references without transmitting retrieved clinical text. Evaluations in both settings show reduced recoverability of private clinical content relative to raw-latent baselines under the evaluated attacks.

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; Wang et al., 2025a). This direction has been extended to evidence-based diagnostic workflows, EHR-grounded decision support, and hospital-scale agent simulation (Wang et al., 2026; 2025b; Li et al., 2024). MDAgents adapts agent collaboration to medical task complexity (Kim et al., 2024). For rare diseases, DeepRare combines specialized tools with similar-case retrieval (Zhao et al., 2026), while RareAgents combines multi-disciplinary reasoning with retrieval-based patient

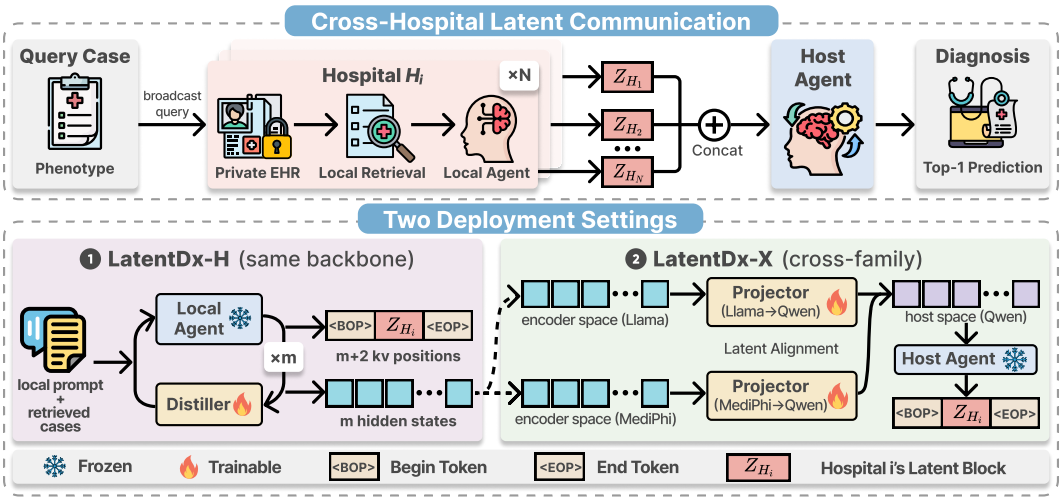


Figure 2: **Overview of LatentDx. Top:** Hospital agents keep private EHR records local, retrieve relevant local cases, and send compact latent blocks to a host agent for diagnosis. **Bottom:** LatentDx supports two deployment settings: LatentDx-H uses latent KV distillation for same-backbone hospital agents, while LatentDx-X uses lightweight projectors to map transmitted hidden-state sequences into host input embeddings for cross-family collaboration.

memory (Chen et al., 2026a). PhenoBrain and RareBench provide phenotype-based diagnostic modeling and benchmark evaluation (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). The agent-based studies demonstrate the value of collaboration in medical decision-making. We likewise adopt a collaborative diagnostic workflow, but focus on what hospital agents should exchange when retrieved clinical text cannot be shared across institutions. We formulate this communication problem and develop LatentDx to exchange compact latent representations, jointly evaluating diagnostic utility and empirical leakage.

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., 2025). These methods aggregate distributed experience into training-time updates, but they do not let a diagnosing hospital condition the current query on 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 LatentDx 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 & 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 (Zou et al., 2026; Du et al., 2026), while KV-alignment methods learn cache projection across LLMs (Dery et al., 2026). Further work on systems shows that KV caches can be compressed, reused, and transmitted efficiently (Zhang et al., 2023; Xiao et al., 2024; Liu et al., 2024; Chen et al., 2026b). Beyond cache efficiency, Cache-to-Cache (C2C) studies cross-model semantic transfer through the projection and fusion of token-aligned KV caches computed by both models from a shared input (Fu et al., 2026). However, these studies primarily target general reasoning, cross-model transfer, and inference efficiency. They leave open how latent

communication can support diagnosis from institution-local case repositories while limiting the recoverability of private retrieved-case information.

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. `LatentDx` 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 leakage settings in Sec. 3.1, then present `LatentDx-H` for same-backbone latent distillation in Sec. 3.2 and `LatentDx-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. The Human Phenotype Ontology (HPO) provides standardized terms for phenotypic abnormalities (Gargano et al., 2024). We represent each query as an HPO phenotype set q_{HPO}, whose codes are deterministically rendered into canonical term names for prompting. The query hospital broadcasts these phenotypes to participating hospitals to seek diagnostic support. We assume that this query sharing and the exchange of protocol-defined latent representations are authorized, while retrieved clinical records must remain local. 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 retrieved cases contribute their disease labels and phenotype descriptions, which constitute the private clinical text in x_i. In `LatentDx`, the retrieved records, local prompt x_i, and prompt-position KV states remain within the hospital. During inference, each hospital agent sends only a compact latent representation $\mathcal{T}_{H_i}$ to the host agent, typically the query hospital, providing diagnostic evidence without directly exposing private case text. The host outputs a single disease name, which the evaluator maps to an identifier in Online Mendelian Inheritance in Man (OMIM), a catalog of human genes and genetic phenotypes (Amberger et al., 2019). The normalized diagnosis is

$$\hat{y} = \mathrm{Norm}_{\mathrm{OMIM}}\left[\mathrm{Host}\left(q_{\mathrm{HPO}},\ (\mathcal{T}_{H_i})_{i=1}^{N}\right)\right],\ \hat{y} \in \mathcal{Y}_{\mathrm{OMIM}}. \tag{1}$$

Here, $\mathrm{Norm}_{\mathrm{OMIM}}$ denotes the evaluator's disease-name mapping (Appendix B.1.3), and $\mathcal{T}_{H_i}$ denotes the transmitted object: a compact KV block including contextualized boundary positions in `LatentDx-H`, a compact hidden-state sequence in `LatentDx-X`, or prompt-length KV states in the raw-KV baseline. The host consumes these objects through mode-specific interfaces.

Leakage setting. We evaluate 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 `LatentDx`. The observer knows the participating LLM architectures and weights, but cannot access private hospital databases. Given $\mathcal{T}_{H_i}$, an attacker $\mathcal{A}$ targets a secret s_i held by hospital H_i:

$$\hat{s}_i = \mathcal{A}(\mathcal{T}_{H_i}), \qquad \rho_i = \mathrm{score}(\hat{s}_i, s_i). \tag{2}$$

Attacks differ in what they take s_i to be: the local prompt content x_i including retrieved clinical evidence, the retrieved disease label alone, or the bit recording whether a given case lies in $\mathcal{D}_i$. Our first instantiation of $\mathcal{A}$ is a frozen-LLM continuation attack on same-backbone KV transmissions, which injects the intercepted 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. Here $s_i = x_i$ and score is token F1 or BERTScore. Sec. 4.5 adds four stronger attackers, including ones that train a dedicated model on the transmitted representation. Our results show substantially reduced recoverability of private clinical information relative to other baselines under the evaluated attacks.

3.2 `LatentDx-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. Here, distillation denotes task-supervised autoregressive context compression. Figure 3 illustrates this distillation pipeline.

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

$$\mathcal{M}_{H_i} = \{(\mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)})\}_{l=1}^{L}, \qquad \mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)} \in \mathbb{R}^{T_i \times d_{\mathrm{KV}}}. \tag{3}$$

where L is the number of layers, T_i is the local prompt length, and $d_{\mathrm{KV}} = n_{\mathrm{KV}} d_{\mathrm{head}}$ is the per-layer KV width, with n_{KV} heads of dimension d_{head}. The matrices $\mathbf{K}_{H_i}^{(l)}, \mathbf{V}_{H_i}^{(l)}$ are the layer-l key and value states. Direct prompt-length KV sharing avoids explicit clinical text transmission but exposes internal representations of the entire local prompt, including retrieved private cases. `LatentDx-H` instead keeps $\mathcal{M}_{H_i}$ local and transmits a compact block $\mathcal{T}_{H_i}$ containing distilled latent KV states and two contextualized boundary positions. Retrieval details are in Appendix A.2.

Distilling compact latent blocks. After encoding its local prompt, each hospital agent processes a learned begin-of-block embedding $\langle \mathrm{BOP} \rangle$, then uses the distiller's learned start embedding for the first of m latent steps. The lightweight distiller ϕ maps hidden states to input embeddings,

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

where $W \in \mathbb{R}^{d_{\mathrm{model}} \times d_{\mathrm{model}}}$ is a bias-free linear projection and d_{model} is the backbone hidden dimension. The resulting vector is fed into the frozen local LLM backbone as the next input embedding without decoding text. After m latent steps, the hospital processes a learned end-of-block embedding $\langle \mathrm{EOP} \rangle$. Let $\mathcal{Z}_{H_i}$ denote the KV states of the m latent positions, excluding both boundaries, and $\mathcal{M}_{H_i}^{\mathrm{full}}$ the complete cache including the prompt. The transmitted block is

$$\mathcal{T}_{H_i} = \mathrm{Tail}_{m+2}\big(\mathcal{M}_{H_i}^{\mathrm{full}}\big). \tag{5}$$

Here, Tail_{m+2} retains the final $m+2$ positions of every layer's keys and values, leaving the prompt positions local. The boundary embeddings are shared trainable parameters, but their KV states are computed in the local prompt context. Each hospital therefore transmits $m+2$ contextualized KV positions (34 for $m = 32$), independent of the local prompt length T_i. The host concatenates the blocks along the position axis as $\mathcal{T}_{H_{1:N}} = \mathcal{T}_{H_1} \oplus \cdots \oplus \mathcal{T}_{H_N}$ to produce the diagnosis.

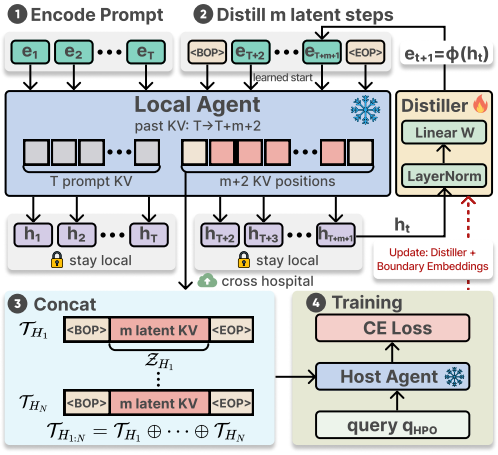


Figure 3: `LatentDx-H`**: same-backbone latent distillation.** A hospital agent encodes its local prompt with a frozen LLM backbone, keeping the prompt-length states local. A trainable distiller generates m latent steps. Their KV states and two contextualized boundary positions form the transmitted block of $m+2$ positions. The host agent aggregates these blocks for diagnosis. Diagnosis cross-entropy updates only the distiller and boundary embeddings.

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

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

where $\mathcal{T}_{H_{1:N}}$ denotes the concatenated transmitted KV blocks, each containing m latent positions and two contextualized boundary positions. 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 centrally train the latent interface on the public training split with simulated hospital partitions and disjoint test cases.

3.3 LatentDx-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. LatentDx-X addresses this mismatch through cross-family latent alignment.

Cross-family projector. In LatentDx-X, host-family hospitals retain the LatentDx-H KV interface, while foreign-family hospitals transmit compact hidden-state sequences. This reuses compatible same-family KV blocks directly, while resolving cross-family incompatibility through host-side projection and re-encoding. A lightweight projector maps these states into the host agent's input-embedding space. Let $\mathcal{E}$ denote an encoder backbone family and $\mathcal{H}$ denote the host backbone family. Let $\mathbf{h}^{(\mathcal{E})}_{1:m_\mathcal{E}} \in \mathbb{R}^{m_\mathcal{E} \times d_\mathcal{E}}$ denote the concatenated compact latent states from hospital agents using encoder family $\mathcal{E}$, where $m_\mathcal{E}$ is the total number of latent positions in that group. The projector maps it to host input embeddings

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

In our implementation, $\psi_{\mathcal{E}\to\mathcal{H}}$ is a token-wise LayerNorm followed by a bias-free linear projection applied at the host. Each foreign hospital's projected sequence is wrapped with the host's boundary embeddings and independently encoded by the frozen host into an $(m+2)$-position KV block. Host-family hospitals contribute their LatentDx-H KV blocks directly. All KV blocks are then concatenated along the position axis for diagnosis.

Joint projector training. For each host backbone family $\mathcal{H}$, we train one projector $\psi_{\mathcal{E}\to\mathcal{H}}$ for each non-host encoder family $\mathcal{E}$. 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 uses the same diagnosis cross-entropy objective as Eq. 6, conditioned on the assembled host-space KV prefix. Only the projectors and boundary embeddings are updated. The encoder LLM backbones, encoder-side distillers, and host LLM backbone remain frozen. For the backbone checkpoints used during projector training, the same projector set can be reused across hospital compositions without mixture-specific retraining.

4 Experiments

The experiments below are organized around one question: whether LatentDx retains the diagnostic benefit while limiting what an adversary recovers from the transmitted state.

4.1 Benchmark Construction

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). The benchmark couples hospital-local retrieval with explicit communication interfaces and a joint diagnostic-utility and empirical-leakage evaluation protocol. Each case carries HPO codes and a ground-truth OMIM identifier. After label resolution, deduplication and frequency filtering, 8,022 cases across 235 OMIM diseases remain, split $90/5/5$ into training, validation and test. We partition the training cases into five hospital-local databases, and at inference each agent retrieves only from its own $\mathcal{D}_i$, so diagnostic evidence is distributed across hospitals while retrieved clinical text stays local. The main results use a balanced partition. Appendix D.3 repeats them under a non-IID Dirichlet(α=0.3) partition in which each disease concentrates in a few hospitals, raising the mean per-disease single-hospital share from 0.20 to 0.65. LatentDx maintains strong diagnostic utility on both hosts under this skewed partition.

CrossRare-Bench represents each case as a list of canonical HPO term names. To evaluate narrative evidence, we build CrossRare-Bench-Text, in which both the query and the hospital-retrieved records are rendered as clinical narratives for every diagnosis instance. Cohorts, hospital partitions, and retrieval budget remain fixed, so only the surface form of the phenotype evidence changes. Construction details for both corpora are in Appendices A.1 and D.2.

Table 1: **Diagnostic utility on CROSSRARE-BENCH.** Accuracy (Acc), macro F1 (F1) and LLM-as-judge score (Judge) on each cohort and the full test set (Overall). **Best** and second best method per column, ranked separately for each backbone, with tied methods both marked. For `MediPhi` and `Llama-3.1-8B`, see Table 15.

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
	LatentDx	**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
	LatentDx	**0.75**	**0.64**	**0.77**	**0.70**	**0.62**	**0.74**	**0.73**	**0.72**	**0.78**	**0.73**	**0.68**	**0.77**

4.2 EXPERIMENTAL SETUP

Baselines. We compare `LatentDx` against eight training-free baselines in three groups, all defined in Appendix B.1.1. ❶ **Non-collaborative** methods see only the query hospital: `Zero-shot` and `Single RAG`. ❷ **Text / structured sharing** methods send retrieved case text, structured retrieval outputs or natural-language assessments across hospital boundaries: `Dist. RAG`, `Struct. RAG`, `TextMAS`, and `Global RAG`, a centralized text-sharing baseline that retrieves from the pooled databases of the same participating hospitals. ❸ **Latent communication** methods send latent state: `Raw KV` and `LatentMAS`. We also evaluate a separately trained single-hospital ($N = 1$) `LatentDx` baseline under the same diagnosis-supervised recipe (Appendix E.2). Supervised text-sharing references use `Dist. RAG` with prefix tuning or host-side LoRA (Appendix D.6). All methods in the main comparison are evaluated on `Llama-3.2-3B`, `Qwen3-4B` and `MediPhi`, and on the larger `Llama-3.1-8B` in Appendix D.1.

Implementation details. We set m=32 latent positions and use a fixed subset of N=3 of the five hospital partitions, with hospital order held fixed across test queries and collaborative methods, unless otherwise specified. Diagnostic retrieval uses cosine HPO similarity with matched total budgets across retrieval-based methods, excluding the query case itself during training and evaluation. We report diagnostic utility as accuracy, macro F1 and an LLM-as-judge score, and leakage in three kinds, one per secret the attacker targets. Metrics and settings are detailed in Appendices B.2–B.3.

4.3 DIAGNOSTIC UTILITY

Table 1 reports diagnostic utility in the same-backbone setting. ❶ **Compact latent communication supports diagnostic utility across model families and scales.** On `Llama-3.2-3B` and `Qwen3-4B`, `LatentDx-H` achieves diagnostic utility competitive with supervised `Dist. RAG` using prefix tuning or host-side LoRA, without transmitting retrieved clinical text (Appendix D.6). Additional results on `MediPhi` and `Llama-3.1-8B` show that compact latent communication also supports strong diagnostic performance across the evaluated model families and scales (Appendix D.1). ❷ **Cross-hospital collaboration provides additional diagnostic utility.** On both backbones, three-hospital collaboration improves overall accuracy, macro F1 and judge score over independently

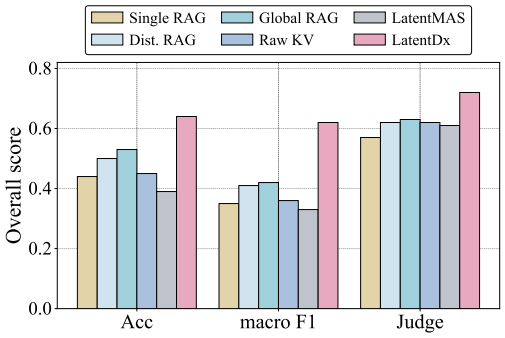


Figure 4: **CROSSRARE-BENCH-TEXT.** Overall scores on `Llama-3.2` with both the query and the hospital-retrieved records rendered as clinical narratives. All three hosts in Appendix D.2.

Table 2: **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`	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`	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

trained query-only ($N = 0$) and single-hospital ($N = 1$) supervised baselines (Appendix E.2). This supports the value of combining evidence from multiple hospital-local databases, even when the single-hospital interface has already been trained for diagnosis. We report diagnostic utility across three independent training seeds in Appendix D.4.

4.4 CROSS-FAMILY COLLABORATION

Table 2 evaluates `LatentDx-X` across hospital compositions using `Qwen3-4B`, `Llama-3.2-3B` and `MediPhi`. Diagnostic performance remains close to the same-backbone setting across the evaluated compositions, with each host reusing the same projector set without mixture-specific retraining. Appendix C.2 reports cross-family payload sizes and host-side re-encoding FLOPs.

4.5 LEAKAGE UNDER ATTACK

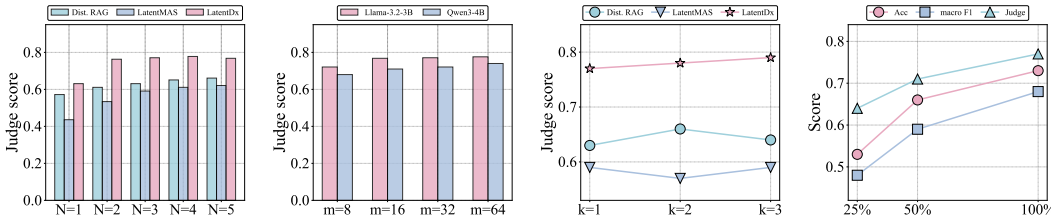
Table 3 evaluates same-backbone `LatentDx-H` under two training-free and three learned attacks. Text-sharing baselines are omitted because their transmitted text is directly observable. ❶ **Latent distillation reduces leakage that raw latent communication leaves exposed.** `LatentDx-H`'s prompt-reconstruction scores are at or near the no-transmission controls, where the attacker receives no transmitted representation. This indicates little additional prompt recovery from the communicated blocks under this attack. Both `Raw KV` and `LatentMAS` leak under all five attacks, showing that operating in latent space is not protective by itself. `LatentMAS` retains the prompt KV cache alongside its additional latent states, leaving the original evidence exposed. Across all five evaluated attacks on both hosts, `LatentDx-H` achieves lower leakage than these baselines while retaining the diagnostic utility shown in Table 1. ❷ **The utility–leakage advantage persists under matched supervision and latent length.** To compare with another supervised method, we evaluate `MLP-Pool32`, which replaces hospital-side autoregressive latent distillation with mean pooling of local prompt hidden states followed by an MLP that produces $m = 32$ latent input embeddings. Both methods use the same diagnosis objective and latent-position budget. `LatentDx-H` achieves higher diagnostic utility and lower leakage under all three evaluated learned attacks on both backbones (Appendix E.1). These results support the benefit of our latent distillation interface over supervised mean-pooling under matched supervision and latent length. ❸ **Cross-family collaboration retains a favorable leakage profile relative to raw-latent baselines.** Under the same five-attack protocol, host-side evaluation of `LatentDx-X` also shows lower leakage than the same-backbone `Raw KV` and `LatentMAS` references in both evaluated cross-family directions (Appendix F.4).

4.6 HYPERPARAMETER ANALYSIS

Figure 5 ablates four settings of the experimental configuration, each varied in isolation on the `Llama-3.2` host. ❶ **Hospital agents N.** With the trained checkpoint fixed, `LatentDx`'s judge score improves as more hospitals contribute evidence at inference. This indicates that the same trained interface can benefit from additional hospital evidence. ❷ **Latent block length m.** Larger latent

Table 3: **Same-backbone leakage under attacks of increasing attacker capability.** Lower is better. No tx. denotes an attack-specific no-transmission control: the attacker receives no transmitted representation. It serves as the control baseline for measuring additional leakage attributable to transmission. Learned-attack entries are means over three random seeds. Reduction is the relative drop from `Raw KV` to `LatentDx`. Metrics are not comparable across rows. `MediPhi` and full details appear in Appendix F.3.

		Qwen3-4B					Llama-3.2-3B				
Attack	**Metric**	No tx.	Raw KV	LatentMAS	LatentDx	**Reduction**	No tx.	Raw KV	LatentMAS	LatentDx	**Reduction**
Training-free attacks											
Prompt reconstruction	Token F1 ↓	0.05	0.60	0.60	0.07	↓88%	0.07	0.68	0.82	0.07	↓90%
Template-aware extraction	Disease recall ↓	0.00	0.89	0.90	0.20	↓78%	0.00	0.88	0.75	0.16	↓82%
Learned attacks											
Attribute probe	OMIM exact ↓	0.17	0.62	0.62	0.23	↓63%	0.17	0.61	0.60	0.20	↓67%
Inversion decoder	OMIM exact ↓	0.27	0.97	0.96	0.40	↓59%	0.27	0.99	0.99	0.38	↓62%
Membership inference	TPR@1%FPR ↓	0.00	0.99	0.93	0.17	↓83%	0.00	1.00	0.83	0.03	↓97%



(a) Hospital agents N (b) Latent block length m (c) Retrieval depth k (d) Training data size

Figure 5: **Hyperparameter analysis.** Other settings follow the main-experiment defaults. (a) and (c) vary hospital count N and retrieval depth k on `Llama-3.2`, comparing `Dist. RAG`, `LatentMAS` and `LatentDx`. `LatentDx` varies N only at inference with a fixed checkpoint and trains its interface separately for each k. (b) varies latent length m for `LatentDx` on `Llama-3.2` and `Qwen3`. (d) varies the training-data fraction for `LatentDx` on `Llama-3.2`. Panels (a)–(c) report judge score, and (d) reports all three utility metrics. Full results in Appendix E.3.

blocks generally improve diagnostic performance. Since longer blocks also lengthen the transmitted state and raise inference cost, we use m=32 in the main experiments, giving a per-prompt KV footprint about one-fifth to one-sixth that of `Raw KV` (Appendix C.2). Appendix C.1 profiles the rest of the cost: at three hospitals `LatentDx` adds 0.01s per query and 0.7GB of peak memory over `Raw KV`, and fitting the interface is a one-time 5.5 GPU-hours. ❸ **Retrieval depth** k**.** `LatentDx` achieves higher judge scores with greater retrieval depth. ❹ **Training data.** Performance rises across the whole sweep and has not saturated at the full split. ❺ **Extended ablations.** Appendix E.4 further examines the hospital-side distiller architecture for compressing local evidence and the host-side aggregation mechanism for combining evidence across hospitals.

5 Conclusion

We introduced `LatentDx`, a latent multi-agent communication framework for cross-hospital rare-disease diagnosis. It keeps retrieved cases and prompt-length KV caches local while allowing hospitals to exchange compact, diagnosis-oriented latent representations in both same-backbone and cross-family settings. Experiments on CrossRare-Bench show that this interface supports strong diagnostic performance while substantially reducing recoverable clinical information relative to raw-latent baselines under the evaluated attacks. Supervised controls further support the diagnostic benefit of cross-hospital collaboration and the utility–leakage advantage of latent distillation over mean-pooling. Diagnostic utility remains strong across the evaluated model families and scales, on narrative evidence, and under non-IID hospital partitions. These results support compact latent representations as a promising communication interface for cross-hospital diagnosis without sharing private clinical records.

AI USE STATEMENT

We used Large Language Models (ChatGPT/Claude/Gemini) for grammatical correction in this manuscript. Separately, `GPT-5.5` generated the paired clinical narratives in CROSSRARE-BENCH-TEXT from HPO term names under the constrained protocol described in Appendix D.2. It received no disease name, OMIM identifier, or causal gene, and the outputs were screened for explicit diagnostic-label leakage. LLMs played no role in research ideation or methodology. All technical contributions, analyses, and scientific conclusions are the authors' own.

ETHICS STATEMENT

`LatentDx` is a research framework intended to assist, rather than replace, clinical judgment. Because rare-disease predictions can be wrong or perform unevenly across diseases, institutions and patient groups, real use would require clinical validation and expert review.

REPRODUCIBILITY STATEMENT

Training hyperparameters, implementation details, and evaluation protocols are provided in Appendices B and F, with dataset construction and retrieval procedures in Appendix A. Code and data are available in our anonymous repository at `https://anonymous.4open.science/r/LatentDx/`.

REFERENCES

Joanna S. Amberger, Carol A. Bocchini, Alan F. Scott, and Ada Hamosh. OMIM.org: leveraging knowledge across phenotype–gene relationships. *Nucleic Acids Research*, 47(D1):D1038–D1043, 2019. doi: 10.1093/nar/gky1151. URL `https://doi.org/10.1093/nar/gky1151`.

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. AMELIE speeds Mendelian diagnosis by matching patient phenotype and genotype to primary literature. *Science Translational Medicine*, 12(544):eaau9113, 2020. doi: 10.1126/scitranslmed.aau9113.

Xuanzhong Chen, Xiaohao Mao, Qihan Guo, Lun Wang, Shuyang Zhang, and Ting Chen. RareBench: Can LLMs serve as rare diseases specialists? In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, pp. 4850–4861, Barcelona, Spain, 2024. ACM. doi: 10.1145/3637528.3671576. URL `https://doi.org/10.1145/3637528.3671576`.

Xuanzhong Chen, Ye Jin, Xiaohao Mao, Lun Wang, Shuyang Zhang, and Ting Chen. RareAgents: Autonomous multi-disciplinary team for rare disease diagnosis and treatment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):101–109, March 2026a. doi: 10.1609/aaai.v40i1.36969. URL `https://ojs.aaai.org/index.php/AAAI/article/view/36969`.

Zihan Chen, Zeshen Li, Howard H. Yang, Tony Q. S. Quek, and Jihong Park. Federated inference for heterogeneous LLM communication and collaboration. *arXiv preprint arXiv:2603.28772*, 2026b. URL `https://arxiv.org/abs/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. Consistent performance of large language models in rare disease diagnosis across ten languages and 4917 cases. *EBioMedicine*, 121:105957, 2025. doi: 10.1016/j.ebiom.2025.105957. URL `https://doi.org/10.1016/j.ebiom.2025.105957`.

Jean-Philippe Corbeil, Amin Dada, Jean-Michel Attendu, Asma Ben Abacha, Alessandro Sordoni, Lucas Caccia, Francois Beaulieu, Thomas Lin, Jens Kleesiek, and Paul Vozila. A modular approach for clinical SLMs driven by synthetic data with pre-instruction tuning, model merging, and clinical-tasks alignment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19352–19374, Vienna, Austria, July

2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.950. URL https://aclanthology.org/2025.acl-long.950/.

Francesco Cremonesi, Marc Vesin, Sergen Cansiz, Yannick Bouillard, Irene Balelli, Lucia Innocenti, Riccardo Taiello, Santiago Silva, Samy-Safwan Ayed, Melek Önen, Fanny Orlhac, Christophe Nioche, Bastien Houis, Romain Modzelewski, Nathan Lapel, Renaud Schiappa, Olivier Humbert, and Marco Lorenzi. Fed-BioMed: Open, transparent and trusted federated learning for real-world healthcare applications. In *Federated Learning Systems*, volume 832 of *Studies in Computational Intelligence*, pp. 19–41. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-031-78841-3_2. URL https://doi.org/10.1007/978-3-031-78841-3_2.

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. A corpus of GA4GH phenopackets: case-level phenotyping for genomic diagnostics and discovery. *Human Genetics and Genomics Advances*, 6(1):100371, 2025. doi: 10.1016/j.xhgg.2024.100371. URL https://doi.org/10.1016/j.xhgg.2024.100371.

Lucio M. Dery, Zohar Yahav, Henry Prior, Qixuan Feng, Jiajun Shen, and Arthur Szlam. Latent space communication via K-V cache alignment, 2026. URL https://arxiv.org/abs/2601.06123. Google DeepMind technical report.

Tian Dong, Yan Meng, Shaofeng Li, Guoxing Chen, Zhen Liu, and Haojin Zhu. Depth gives a false sense of privacy: LLM internal states inversion. In *34th USENIX Security Symposium (USENIX Security '25)*, pp. 1629–1648, Seattle, WA, USA, 2025. USENIX Association. ISBN 978-1-939133-52-6. URL https://www.usenix.org/conference/usenixsecurity25/presentation/dong-tian.

Minxin Du, Xiang Yue, Sherman S. M. Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. 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)*, pp. 2665–2679, New York, NY, USA, 2023. Association for Computing Machinery. doi: 10.1145/3576915.3616592.

Zhuoyun Du, Runze Wang, Huiyu Bai, Zouying Cao, Xiaoyong Zhu, Yu Cheng, Bo Zheng, Wei Chen, and Haochao Ying. Enabling agents to communicate entirely in latent space. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27106–27129. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1248. URL https://aclanthology.org/2026.acl-long.1248/.

Tianyu Fu, Zihan Min, Hanling Zhang, Jichao Yan, Guohao Dai, Wanli Ouyang, and Yu Wang. Cache-to-Cache: Direct semantic communication between large language models. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2510.03215. ICLR 2026.

Michael A. Gargano, Nicolas Matentzoglu, Ben Coleman, Eunice B. Addo-Lartey, Anna V. Anagnostopoulos, Joel Anderton, Paul Avillach, Anita M. Bagley, Eduard Bakštein, James P. Balhoff, et al. The Human Phenotype Ontology in 2024: phenotypes around the world. *Nucleic Acids Research*, 52(D1):D1333–D1346, 2024. doi: 10.1093/nar/gkad1005. URL https://doi.org/10.1093/nar/gkad1005.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=ph04CRkPdC.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling (COLM)*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, and Hae Won Park. MDAgents: An adaptive collaboration of LLMs for medical decision-making. In *Advances in Neural Information Processing Systems*, volume 37, pp. 79410–79452. Curran Associates, Inc., 2024. doi: 10.52202/079017-2522. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/90d1fc07f46e31387978b88e7e057a31-Abstract-Conference.html.

Anna Kuzina, Maciej Pióro, and Babak Ehteshami Bejnordi. KaVa: Latent reasoning via compressed KV-cache distillation. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2510.02312.

Junkai Li, Yunghwei Lai, Weitao Li, Jingyi Ren, Meng Zhang, Xinhui Kang, Siyu Wang, Peng Li, Ya-Qin Zhang, Weizhi Ma, and Yang Liu. Agent Hospital: A simulacrum of hospital with evolvable medical agents, 2024. URL https://arxiv.org/abs/2405.02957.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *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)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Xiao-Yang Liu, Rongyi Zhu, Daochen Zha, Jiechao Gao, Shan Zhong, Matt White, and Meikang Qiu. Differentially private low-rank adaptation of large language model using federated learning. *ACM Transactions on Management Information Systems*, 16(2):1–24, March 2025. doi: 10.1145/3682068. URL https://doi.org/10.1145/3682068.

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. CacheGen: KV cache compression and streaming for fast large language model serving. In *Proceedings of the ACM SIGCOMM 2024 Conference*, pp. 38–56, Sydney, Australia, 2024. ACM. doi: 10.1145/3651890.3672274. URL https://doi.org/10.1145/3651890.3672274.

Zhifan Luo, Shuo Shao, Su Zhang, Lijing Zhou, Yuke Hu, Chenxu Zhao, Zhihao Liu, and Zhan Qin. Shadow in the cache: Unveiling and mitigating privacy risks of KV-cache in LLM inference. In *Proceedings of the 33rd Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2026. Internet Society. ISBN 979-8-9919276-8-0. doi: 10.14722/ndss.2026.240258. URL https://doi.org/10.14722/ndss.2026.240258.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. 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*, pp. 34281–34302. PMLR, 2024. URL https://proceedings.mlr.press/v235/mai24a.html.

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. A phenotype-based AI pipeline outperforms human experts in differentially diagnosing rare diseases using EHRs. *npj Digital Medicine*, 8(1):68, 2025. doi: 10.1038/s41746-025-01452-1. URL https://doi.org/10.1038/s41746-025-01452-1.

Meta. Llama-3.1-8B-Instruct. Official model card, 2024a. URL https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct.

Meta. Llama-3.2-3B-Instruct. Official model card, 2024b. URL https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct.

John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 12448–12460. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.765. URL https://aclanthology.org/2023.emnlp-main.765/.

Jesse Mu, Xiang Lisa Li, and Noah D. Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2304.08467.

Sarthak Pati, Ujjwal Baid, Brandon Edwards, Micah Sheller, Shih-Han Wang, G. Anthony Reina, Patrick Foley, Alexey Gruzdev, Deepthi Karkada, Christos Davatzikos, et al. Federated learning enables big data for rare cancer boundary detection. *Nature Communications*, 13(1): 7346, 2022. doi: 10.1038/s41467-022-33407-5. URL https://www.nature.com/articles/s41467-022-33407-5.

Peter N. Robinson, Vida Ravanmehr, Julius O. B. Jacobsen, Daniel Danis, Xingmin Aaron Zhang, Leigh C. Carmody, Michael A. Gargano, Courtney L. Thaxton, UNC Biocuration Core, Guy Karlebach, Justin Reese, Manuel Holtgrewe, Sebastian Köhler, Julie A. McMurry, Melissa A. Haendel, and Damian Smedley. Interpretable clinical genomics with a likelihood ratio paradigm. *American Journal of Human Genetics*, 107(3):403–417, 2020. doi: 10.1016/j.ajhg.2020.06.021.

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. NVIDIA FLARE: Federated learning from simulation to real-world, 2023. URL https://arxiv.org/abs/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. MedAgents: Large language models as collaborators for zero-shot medical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 599–621, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.33. URL https://aclanthology.org/2024.findings-acl.33/.

Ziqing Wang, Chengsheng Mao, Xiaole Wen, Yuan Luo, and Kaize Ding. AMANDA: Agentic medical knowledge augmentation for data-efficient medical visual question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 24810–24832, Suzhou, China, November 2025a. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.1350. URL https://aclanthology.org/2025.findings-emnlp.1350/.

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. ColaCare: Enhancing electronic health record modeling through large language model-driven multi-agent collaboration. In *Proceedings of the ACM on Web Conference 2025*, pp. 2250–2261. ACM, 2025b. doi: 10.1145/3696410.3714877. URL https://doi.org/10.1145/3696410.3714877.

Ziyue Wang, Junde Wu, Linghan Cai, Chang Han Low, Xihong Yang, Qiaxuan Li, and Yueming Jin. MedAgent-Pro: Towards evidence-based multi-modal medical diagnosis via reasoning agentic workflow. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZOuU0udyA4.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

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

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 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)*, pp. 6137–6147, Barcelona, Spain, 2024. ACM. doi: 10.1145/3637528.3671582. URL https://doi.org/10.1145/3637528.3671582.

Weiqi Zhai, Xiaodi Huang, Nan Shen, and Shanfeng Zhu. Phen2Disease: a phenotype-driven model for disease and gene prioritization by bidirectional maximum matching semantic similarities. *Briefings in Bioinformatics*, 24(4):bbad172, 2023. doi: 10.1093/bib/bbad172. URL https://academic.oup.com/bib/article/24/4/bbad172/7185480.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the FederatedGPT: Federated instruction tuning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6915–6919, Seoul, Korea, 2024. IEEE. doi: 10.1109/ICASSP48485.2024.10447454.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, 2025. URL https://openreview.net/forum?id=ByQdHPGKgU.

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. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 34661–34710, 2023. URL https://arxiv.org/abs/2306.14048.

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. An agentic system for rare disease diagnosis with traceable reasoning. *Nature*, 651(8106):775–784, 2026. doi: 10.1038/s41586-025-10097-9.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 400–410, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.38. URL https://aclanthology.org/2024.acl-demos.38/.

Xiaoying Zhong, Siyi Li, Zhao Chen, Long Ge, Dongdong Yu, Shijia Wang, Liangzhen You, and Hongcai Shang. Considerations for patient privacy of large language models in health care: scoping review. *Journal of Medical Internet Research*, 27:e76571, 2025. doi: 10.2196/76571. URL https://doi.org/10.2196/76571.

Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. Latent collaboration in multi-agent systems. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2511.20639.

APPENDIX ROADMAP

- **Appendix A: Data and task construction** describes the benchmark sources, filtering and splits, hospital partitions, and local retrieval procedure.
- **Appendix B: Experimental setup** defines the baselines and access patterns, gives the prompts and evaluation metrics, and reports the model and training configurations.
- **Appendix C: System efficiency and deployment** reports latency, memory, hospital scaling, interface-training cost, communication volume, and cross-family deployment details.
- **Appendix D: Utility and robustness** provides utility results across model families and scales, free-text evaluation, non-IID hospital experiments, training-seed variability, a Raw KV protocol comparison, and supervised text-sharing references.
- **Appendix E: Representation analysis and ablations** covers the `MLP-Pool32` comparison, supervised query-only and hospital-count controls, retrieval and data sensitivity, and component design.
- **Appendix F: Privacy and leakage** defines the attack settings and reports reconstruction, attribute, probing, inversion, and membership-inference results in same-backbone and cross-family settings.

A DATA AND TASK CONSTRUCTION

A.1 DATASET CONSTRUCTION AND STATISTICS

Source datasets. We construct CROSSRARE-BENCH 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. In the base CROSSRARE-BENCH setting, the query shown to models is the structured HPO phenotype set, rendered as canonical HPO term names.

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.

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.

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

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

A.2 HPO-Based Local Retrieval

In the base CrossRare-Bench setting, both query and database cases are represented as HPO phenotype-code sets, such as $\{\texttt{HP:0000480}, \texttt{HP:0000612}, \ldots\}$. For prompting, each HPO code is deterministically rendered to its canonical term name using the ontology mapping, for example $\texttt{HP:0000480} \mapsto$ "Retinal coloboma."

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 $\mathcal{D}_i$. For diagnostic retrieval during both training and evaluation, the query case itself is excluded from the candidate pool before top-k selection. This exclusion removes only the query case, not other cases with the same disease or phenotype profile. Distributed methods retrieve one case per hospital by default, 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 `LatentDx`, 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 diagnostic retrieval at hospital H_i

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

$\mathcal{C}_i \leftarrow \mathcal{D}_i \setminus \{q\}$ ▷ Exclude the query case itself

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

for each case $c \in \mathcal{C}_i$ **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 top-$\mathrm{k}_{c \in \mathcal{C}_i}$ s_c

B EXPERIMENTAL SETUP

B.1 BASELINES AND PROMPT TEMPLATES

B.1.1 BASELINE DEFINITIONS

Non-collaborative. `Zero-shot` uses only the query phenotypes. `Single RAG` retrieves cases only from the query hospital's own local database.

Text / structured sharing. `Distributed RAG` (`Dist. RAG` in tables) lets each participating hospital retrieve from its own database and shares the retrieved text with the host agent. `Global RAG` is a centralized text-sharing baseline that retrieves from the pooled databases of those same participating hospitals. `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.

Latent communication. `Raw KV` and `LatentMAS` are training-free methods that forward prompt-length KV states using the sequential relay protocol described in Appendix B.1.6.

B.1.2 BASELINE ACCESS PATTERNS

Table 6 summarizes information access in the main comparison. 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. `LatentDx` and the latent baselines aggregate cross-hospital evidence without transmitting retrieved clinical text.

Table 6: Access patterns in the main comparison. 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.

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

Note. `Raw KV` and `LatentMAS` relay cumulative KV caches between hospitals. `LatentDx` sends independently generated hospital blocks directly to the host.

B.1.3 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 `LatentDx`.

B.1.4 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 B.1.3.

B.1.5 TEXT-SHARING COLLABORATION BASELINES

Text-sharing baselines use cross-hospital evidence by transmitting retrieved cases or natural-language messages to the host agent. Under the default $N = 3$ setting, `Distributed RAG` retrieves the top-ranked case from each participating hospital, whereas `Global RAG` retrieves the top three cases from the union of those same hospital databases. Both use cosine HPO similarity and a total budget of three retrieved cases, and share the retrieved text with the host agent. `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. All RAG retrieval uses the HPO-based similarity procedure described in Appendix A.2.

B.1.6 LATENT COMMUNICATION BASELINES

We include two training-free latent communication baselines, both following the sequential relay protocol of `LatentMAS`. In a fixed hospital order, each hospital encodes its local prompt conditioned on the cumulative KV cache from earlier hospitals and forwards the extended cache. The host appends the diagnosis question to the final cache. `Raw KV` forwards prompt-position states directly, while `LatentMAS` additionally performs two autoregressive latent steps before forwarding. Both baselines use frozen LLM backbones and train no interface. Appendix D.5 compares `Raw KV` with an independently encoded parallel variant.

B.1.7 FULL PROMPT TEMPLATES

We list the prompt templates used in our experiments. All prompts are model-agnostic, and $\{\cdots\}$ denote runtime fills. The latent methods (`LatentDx`, `Raw KV`, `LatentMAS`) share the same system prompt (A), hospital-agent prompt (B), and final question (C). `TextMAS` and RAG baselines use separate prompts because their collaboration formats differ. All RAG baselines share system prompt F1. The judge (G1/G2) and attacker (H1/H2) prompts are used only for evaluation.

Table 7 maps each method to the complete prompt templates below.

Table 7: **Method-to-prompt mapping.** Letters refer to the prompt boxes that follow. Latent methods share prompt components A/B/C.

Method	System	Hospital / agent	Final question
`LatentDx` (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
`Struct. RAG`	F1	F3 (labels/scores)	F3
Judge (eval)	G1	–	G2
Attacker (verbatim)	–	–	H1
Attacker (template-aware)	–	–	H2

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>
```

F1. Shared system prompt (all RAG baselines)

```
You are a specialist in the field of rare diseases. You will be provided and asked about a complicated
clinical case; read it carefully and then provide a diverse and comprehensive differential 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
</answer>
```

In F3, {dis_h} is the disease name of hospital h's top-ranked retrieved case. Its similarity to the query, {sim_h}, is the cosine similarity between their IC-weighted HPO vectors (Appendix A.2), formatted to two decimal places. The prompt includes no retrieved-case phenotypes or clinical notes.

F3. Struct. RAG user prompt (default $N = 3$)

```
The following hospitals provide structured non-text diagnostic signals from their local retrieval results:
Hospital 1: retrieved candidate disease = [{dis_1}]; retrieval similarity = {sim_1}
Hospital 2: retrieved candidate disease = [{dis_2}]; retrieval similarity = {sim_2}
Hospital 3: retrieved candidate disease = [{dis_3}]; retrieval similarity = {sim_3}

Patient's phenotype: {test_phenotype}
Based on the query phenotype and the structured hospital signals 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>
```

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}
```

H1. Attacker prompt (verbatim 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.
```

H2. Attacker prompt (template-aware extraction)

```
The previous hidden context is a hospital case that follows this exact template:
"A similar case from Hospital N: The patient has a rare disease [DISEASE], and his/her phenotype is as
follows: [PHENOTYPE_1, PHENOTYPE_2, ...]."
Recover the disease name and the phenotype terms from that hidden case. Output exactly:
Disease: <disease name>
Phenotypes: <comma-separated phenotype terms>
```

B.2 Evaluation Metrics

Diagnostic utility. Accuracy requires an exact OMIM match between the predicted and ground-truth disease. Macro F1 is the unweighted mean of per-class F1 over the OMIM diseases present as gold labels in the evaluated set, not all 235 training diseases. For each included class c, we compute $\mathrm{F1}_c = 2\mathrm{TP}_c/(2\mathrm{TP}_c + \mathrm{FP}_c + \mathrm{FN}_c)$. A class with no true positives receives zero. The full 401-case test set contains 161 gold classes. The RareBench, Zenodo, and Phenopacket cohorts contain 30, 28, and 124, respectively. RareBench pools its RAMEDIS, LIRICAL, MME, and HMS cases before computing F1. Likewise, Overall is computed on all 401 cases, not by averaging cohort F1 scores. Predictions outside the evaluated gold-label set, or outputs that cannot be mapped to OMIM, contribute a false negative to their gold class but no false positive to any evaluated class and no additional class to the average. These rules are identical across methods.

The LLM-as-judge score uses `Qwen3-30B` to compare the predicted diagnosis with the ground-truth disease. It is less sensitive to disease-name surface forms than exact match, and so credits a correct diagnosis written under a synonym.

Leakage. Leakage is what an attacker recovers from the transmitted representation. Each kind targets a different secret and is therefore scored differently. Values are comparable within a kind but not across kinds, and lower is better throughout.

Reconstruction leakage takes the secret to be the source local prompt x_i, which contains the retrieved clinical evidence. We score the attacker's output with token F1 and BERTScore F1 (Zhang et al., 2020). Appendix F.1 also reports rescaled BERTScore F1 and defines the package-baseline rescaling.

Attribute leakage takes the secret to be the retrieved disease itself. Two attacks recover it in different output spaces, so two scores appear. Disease recall is the fraction of cases in which the retrieved disease label string appears in the attacker's greedy continuation, and is used for the training-free extraction attacks. OMIM exact match is the fraction of cases in which a trained attacker predicts the correct OMIM identifier, and is used for the learned probe and inversion decoder, which additionally report HPO micro-F1 over the recovered phenotype terms. Disease recall and OMIM exact match are not interchangeable: the first searches free-form generated text for a string, the second selects one label out of the OMIM space.

Membership leakage takes the secret to be a single bit, namely whether a given case lies in a hospital's retrieval database. We report the true positive rate at a 1% false positive rate, the operating point at which a membership attack is conventionally judged. Chance is 0.01.

B.3 Training and Implementation Details

B.3.1 Model backbones

We evaluate three instruction-tuned LLM backbones: `Llama-3.2-3B-Instruct` (Meta, 2024b), `Qwen3-4B-Instruct` (Yang et al., 2025), and `MediPhi` (Corbeil et al., 2025), a Phi-3.5-based medically adapted model. We additionally evaluate `Llama-3.1-8B-Instruct` (Meta, 2024a) as a larger host in the same-backbone setting (Appendix D.1). The main comparison uses these backbones off the shelf and keeps them frozen.

Table 8: LLM backbones used in our experiments. `MediPhi` is a medically adapted model built on `Phi-3.5-mini-instruct`. Backbones remain frozen in the main comparison.

Backbone	Parameters	Hidden size
`Llama-3.2-3B-Instruct`	3.21B	3072
`Qwen3-4B-Instruct`	4.02B	2560
`MediPhi`	3.82B	3072
`Llama-3.1-8B-Instruct`	8.03B	4096

B.3.2 Default inference configuration

Unless otherwise specified, we use a fixed subset of $N = 3$ of the five hospital partitions, with hospital order held fixed across test queries and collaborative methods. Distributed methods retrieve one local case per hospital, while `Global RAG` retrieves three cases from the pooled databases of those same hospitals. We set $m = 32$ distilled latent positions per hospital block. In same-backbone `LatentDx-H`, the two contextualized boundary positions yield $m + 2 = 34$ transmitted KV positions. The host concatenates independently generated hospital KV blocks in the fixed order into a prefix of $N(m + 2)$ positions, preserving their local rotary phases without re-rotation. Host tokens use position IDs starting from $N(m + 2)$ and attend to all valid prefix positions, with padding masked out. Hospital blocks are not re-encoded after concatenation, and training and inference use the same position and attention conventions. Decoding uses greedy generation with temperature 0. All methods use the single-answer output format in Appendix B.1.3.

B.3.3 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.

Table 9: Training hyperparameters for the latent interface.

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
Default training seed	42

B.3.4 TRAINABLE PARAMETERS

In `LatentDx`, 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 projector set stays under 0.6%.

Table 10: **Trainable parameter counts.** The distiller ϕ and boundary embeddings are trained per family for same-backbone `LatentDx-H`. Cross-family projectors $\psi_{\mathcal{E}\to\mathcal{H}}$ are trained for non-host encoder families in `LatentDx-X`. The host-specific 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`	`Llama-3.2-3B`	`MediPhi`
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 $\psi_{\mathcal{E}\to\mathcal{H}}$ (each)	7.87M (0.20%)	7.87–9.44M (0.25–0.29%)	–
Projector set (total)	15.74M (0.39%)	17.31M (0.54%)	–
Hidden size d	2560	3072	3072
Frozen backbone	4.02B	3.21B	3.82B

B.3.5 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 C.3.

B.3.6 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.

C System Efficiency and Deployment

C.1 Runtime, Memory, and Hospital Scaling

Table 11 profiles latency and memory for the simulated single-GPU pipeline with a `Llama-3.2-3B` host on one NVIDIA H100 80GB GPU. The method comparison uses the same N=3 configuration as the main experiments, and the scaling study varies N from 1 to 5.

Table 11: **System efficiency and scaling.** Panels (a)–(b) report inference time and peak GPU memory for the simulated `Llama-3.2-3B` pipeline on a single NVIDIA H100 80GB GPU, with $N = 3$ hospitals in panel (a). Panel (c) reports one-time training cost per module with all LLM backbones frozen.

(a) Method comparison ($N = 3$)		
Method	**Time/query (s)**	**Peak memory (GB)**
`Raw KV`	0.24	9.8
`LatentMAS`	1.28	12.3
`LatentDx`	0.25	10.5
(b) `LatentDx` **scaling**		
Hospitals (N)	**Time/query (s)**	**Peak memory (GB)**
1	0.11	9.2
2	0.19	10.2
3	0.25	10.5
4	0.34	10.8
5	0.42	11.2
(c) One-time training cost		
Component	**H100 GPU-h**	**Trainable params.**
`LatentDx-H` distiller	5.5	9.45M
`LatentDx-X` projector (each)	1.2	7–10M

At three hospitals, `LatentDx` adds only 0.01 seconds per query and 0.7 GB of peak GPU memory relative to `Raw KV`. `LatentMAS` is slower because it executes latent reasoning sequentially across agents, whereas `LatentDx` independently compresses each hospital's evidence before host-side aggregation. From one to five hospitals, `LatentDx` latency increases from 0.11 to 0.42 seconds and peak memory from 9.2 to 11.2 GB. With all LLM backbones frozen, interface training costs 5.5 H100 GPU-hours per `LatentDx-H` distiller and 1.2 per `LatentDx-X` projector as a one-time expense.

C.2 Communication Footprint

We quantify the representation footprint per local prompt, comparing compact latent blocks with prompt-length KV states. For a bf16 KV block with L layers, length T, and per-layer KV dimension d_{KV}, the block size is

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

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. For each local prompt, `Raw KV` contributes T_i prompt positions, `LatentMAS` adds two latent steps, and same-backbone `LatentDx-H` sends $m = 32$ distilled latent positions plus two contextualized boundary positions. The relative per-prompt KV footprint of `LatentDx-H` is therefore $(m + 2)/T_i = 34/T_i$ for each backbone.

Table 12 reports the 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, a `LatentDx-H` block occupies 18–21% of the corresponding prompt-length KV states, whose footprint is 4.8–5.6× larger. At the 320-token prompt cap, the relative size is $34/320 \approx 10.6\%$, making the prompt-length block 9.4× larger. Rows for text and structured baselines identify their transmitted object types.

In `LatentDx-X`, a foreign-family hospital sends an $[m, d_{\mathcal{E}}]$ hidden-state tensor, requiring $2md_{\mathcal{E}}$ bytes in bf16. Host-family hospitals retain the `LatentDx-H` KV interface. Table 12 reports payloads by sender backbone at $m = 32$, in decimal MB (10^6 bytes).

Table 12: **Representation footprint per local prompt.** Sizes are computed in bf16 using each backbone's local prompt length T_i and KV geometry. Same-backbone LatentDx-H sends $m = 32$ distilled latent positions plus two contextualized boundary positions. Relative size is computed as $(m + 2)/T_i$.

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	$T_i + 2$	24.24 MB	21.33 MB	75.22 MB
Raw KV prompt cap	320	47.2 MB	36.7 MB	125.8 MB
LatentDx-H compact block	$m + 2 = 34$	5.01 MB	3.90 MB	13.37 MB
LatentDx-X foreign hidden states	$m = 32$	0.16 MB	0.20 MB	0.20 MB
LatentDx-H / Raw KV average	$(m + 2)/T_i$	0.209×	0.185×	0.180×

Note. Raw KV and LatentMAS sizes count one hospital's prompt segment. Relay messages additionally carry accumulated earlier-hospital states.

Cross-family communication and computation. Table 13 follows the compositions in Table 2, with hospital 0 co-located with the host and two remote hospitals. Payloads count the two remote evidence tensors only, excluding the local block, query broadcast, and protocol overhead. We approximate transformer computation as $2P_{\text{tr}}T$, where P_{tr} excludes embedding and LM-head parameters. Each foreign block requires re-encoding $T = m + 2 = 34$ positions, while host-family KV blocks are used directly.

Table 13: **Cross-family communication and host-side computation.** Per-query remote payload and re-encoding cost for the hospital compositions in Table 2.

Host	Composition	Remote payload (MB)	Foreign blocks	Re-encoding (GFLOPs)
Qwen3-4B	Q:Q:Q	10.03	0	0
	Q:L:Q	5.21	1	247
	Q:L:M	0.39	2	494
Llama-3.2-3B	L:L:L	7.80	0	0
	L:Q:L	4.06	1	192
	L:Q:M	0.36	2	383

The evaluated test queries average about 114 question tokens and 19 answer tokens. Under this accounting, query/answer processing plus local-block construction costs $970 + 1{,}427 = 2{,}397$ GFLOPs for Qwen3-4B and $751 + 1{,}229 = 1{,}980$ GFLOPs for Llama-3.2-3B, before foreign-block re-encoding. In these configurations, one or two foreign blocks increase this baseline host computation by approximately 10% or 20%, respectively.

C.3 Cross-Family Deployment Details

This appendix expands the cross-family variant LatentDx-X (§3.3) and the heterogeneity evaluation (§4.4). In this setting, hospitals may use different LLM families, such as Qwen3-4B, Llama-3.2-3B, and MediPhi. Foreign-family hidden states are projected and independently re-encoded into host-space KV blocks, while host-family hospitals contribute their LatentDx-H KV blocks directly.

Components. For a host family $\mathcal{H}$, LatentDx-X uses three frozen components and one trainable component. **(i) Per-family distiller.** Each family $\mathcal{E}$ has a frozen distiller $\phi_{\mathcal{E}}$ that compresses a hospital's local prompt into m=32 compact latent positions in family $\mathcal{E}$'s space. **(ii) Cross-family projector.** For each foreign family $\mathcal{E}$, a trainable projector $\psi_{\mathcal{E}\to\mathcal{H}}$ maps a latent block $[m, d_{\mathcal{E}}]$ into the host embedding space $[m, d_{\mathcal{H}}]$. It is implemented as token-wise $\text{LayerNorm}(d_{\mathcal{E}})$ followed by a bias-free $\text{Linear}(d_{\mathcal{E}}, d_{\mathcal{H}})$. **(iii) Boundary embeddings.** Each projected foreign block is wrapped with the host's learned ⟨BOP⟩/⟨EOP⟩ embeddings before independent host-side encoding. Host-family hospitals directly supply their boundary-containing LatentDx-H KV blocks.

Latent-cache construction. Since HPO retrieval is deterministic, foreign-family latents are pre-computed once per family. For each family $\mathcal{E}$, every hospital retrieves from its fixed private database

using cosine HPO retrieval (Appendix A.2), and the retrieved context is encoded into a stored latent block. Each cache entry maps a {case_id, hospital_id} key to a bf16 tensor of shape $[m{=}32, d_{\mathcal{E}}]$, 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 $N \leq 5$.

Joint cross-family projector training. For each host family, we jointly train a host-specific projector set containing one projector for each foreign encoder family. 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 $\psi_{\mathcal{E}\to\mathcal{H}}$. We ensure that each batch contains at least one foreign hospital so every projector receives gradient updates. 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 14.

Table 14: **Joint 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

Deployment and inference. A host uses the same host-specific projector set across the evaluated hospital compositions. Each host-family hospital constructs its `LatentDx-H` KV block locally. Each foreign-family hospital transmits a $[m, d_{\mathcal{E}}]$ hidden-state block produced by $\phi_{\mathcal{E}}$. The host projects each foreign block with $\psi_{\mathcal{E}\to\mathcal{H}}$, adds its boundary embeddings, and independently encodes it into an $(m+2)$-position KV block. All native and re-encoded KV blocks are then concatenated along the position axis for diagnosis.

Evaluation protocol. For Table 2, 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 score on the 401-case test set.

C.4 Interface Distribution and Maintenance

In a prospective deployment, each hospital would retain its local retrieval database and run the latent interface locally. During inference, the hospital would transmit only its compact latent block. No retrieved clinical text or full-backbone parameters are transmitted by LatentDx. In our evaluated setting, distillers and cross-family projectors are trained centrally on public rare-disease data without access to hospital-private records. They can be distributed as lightweight, versioned modules tied to specified backbone checkpoints. For a fixed host checkpoint, the host-specific projector set can be reused across hospital compositions, as evaluated in Appendix C.3.

Institution-specific calibration through federated updates of compatible distillers or projectors is a direction for future work, with records retained locally. Because an interface operates in a backbone-specific representation space, an independently updated backbone checkpoint requires checkpoint-specific validation and, when necessary, interface refresh or retraining.

D Diagnostic Utility and Robustness

D.1 Additional Diagnostic Utility Results

Table 15 reports diagnostic utility for the medically pretrained MediPhi host and the larger Llama-3.1-8B host under the same protocol as the main results. These results show strong diagnostic utility with both the medically pretrained and larger backbones.

Table 15: **Diagnostic utility on CrossRare-Bench for the MediPhi and Llama-3.1-8B hosts.** Same protocol as Table 1. **Best** and second best method per column, ranked separately for each backbone, with tied methods both marked.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
	Non-collaborative												
	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
MediPhi 3.8B	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
	LatentDx	**0.67**	**0.49**	0.73	0.62	0.59	**0.72**	**0.55**	**0.53**	0.69	**0.59**	**0.49**	**0.73**
	Non-collaborative												
	Zero-shot	0.11	0.10	0.35	0.03	0.05	0.23	0.03	0.02	0.20	0.04	0.03	0.23
	Single RAG	0.42	0.34	0.53	0.55	0.48	0.58	0.54	0.47	0.65	0.52	0.44	0.62
	Text / structured sharing												
	Dist. RAG	0.47	0.39	0.63	0.66	0.63	0.70	0.61	0.55	0.71	0.60	0.52	0.69
Llama-3.1 8B	Global RAG	0.46	0.40	0.60	0.65	0.61	0.71	0.60	0.53	0.68	0.59	0.50	0.67
	Struct. RAG	0.51	0.36	0.63	0.64	0.51	0.70	0.55	0.46	0.71	0.57	0.42	0.69
	TextMAS	0.49	0.38	0.61	0.64	0.53	0.69	0.54	0.48	0.66	0.55	0.45	0.66
	Latent communication												
	Raw KV	0.58	0.45	0.68	0.69	0.61	0.72	0.62	0.54	0.71	0.63	0.51	0.71
	LatentMAS	0.54	0.43	0.65	0.70	0.62	0.73	0.60	0.51	0.71	0.61	0.49	0.70
	LatentDx	**0.74**	**0.63**	**0.79**	**0.71**	**0.65**	**0.74**	**0.68**	**0.65**	**0.76**	**0.70**	**0.63**	**0.76**

D.2 CROSSRARE-BENCH-TEXT: FREE-TEXT CLINICAL NARRATIVES

CROSSRARE-BENCH-TEXT renders both the query case and every hospital-retrieved record as clinical narratives instead of lists of canonical HPO term names. Cohorts, splits, hospital partitions and retrieval budget are unchanged, so the only difference from Table 1 is the surface form of the phenotype evidence presented to the diagnosis models. The prompt templates are otherwise unchanged: their query and retrieved-case fields are filled with the corresponding narrative renderings. We retrain the `LatentDx` distiller on the narrative training split.

Using `GPT-5.5`, we generate a narrative for every query and database-case profile from its HPO term names alone. Disease names, OMIM identifiers, and causal genes are withheld from the generator input. We screen the generated corpus for explicit diagnostic labels before use.

Retrieval remains based on the underlying HPO embeddings in all settings. `Struct. RAG` receives the query as a clinical narrative while retaining the same structured hospital signals: retrieved disease labels and HPO similarity scores.

Table 16 reports results for all nine methods.

Table 16: **Diagnostic utility on CROSSRARE-BENCH-TEXT.** Same protocol, cohorts and splits as Table 1, with both the query and the hospital-retrieved records rendered as clinical narratives instead of HPO term lists. **Best** and second best method per column, ranked separately for each backbone, with tied methods both marked. `Struct. RAG` uses narrative queries with retrieved disease labels and HPO similarity scores.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
Qwen3 4B	*Non-collaborative*												
	Zero-shot	0.18	0.15	0.40	0.02	0.04	0.23	0.02	0.01	0.19	0.04	0.04	0.23
	Single RAG	0.53	0.36	0.58	0.51	0.43	0.57	0.59	0.52	0.72	0.56	0.47	0.67
	Text / structured sharing												
	Dist. RAG	0.63	0.49	**0.75**	0.60	0.59	0.65	0.61	0.56	0.72	0.61	0.54	0.71
	Global RAG	0.63	0.48	0.74	0.59	0.59	0.64	0.61	0.54	**0.73**	0.61	0.52	0.71
	Struct. RAG	0.56	0.41	0.68	0.59	0.47	0.62	**0.62**	0.54	**0.73**	0.61	0.48	0.70
	TextMAS	0.51	0.35	0.61	0.55	0.51	0.64	0.54	0.45	0.68	0.54	0.41	0.66
	Latent communication												
	Raw KV	0.60	0.48	0.70	**0.62**	0.55	0.67	0.57	0.52	0.69	0.58	0.50	0.69
	LatentMAS	0.61	0.47	0.72	0.55	0.50	0.66	0.54	0.51	0.68	0.55	0.48	0.68
	LatentDx	**0.69**	**0.57**	**0.75**	0.60	**0.60**	**0.70**	**0.62**	**0.62**	**0.73**	**0.62**	**0.60**	**0.73**
Llama-3.2 3B	*Non-collaborative*												
	Zero-shot	0.05	0.02	0.35	0.02	0.02	0.13	0.01	0.00	0.13	0.02	0.01	0.16
	Single RAG	0.40	0.32	0.51	0.42	0.35	0.49	0.46	0.38	0.60	0.44	0.35	0.57
	Text / structured sharing												
	Dist. RAG	0.47	0.33	0.58	0.53	0.47	0.60	0.50	0.42	0.64	0.50	0.41	0.62
	Global RAG	0.51	0.41	0.61	**0.55**	0.48	0.63	0.53	0.44	0.64	0.53	0.42	0.63
	Struct. RAG	0.53	0.33	0.63	0.30	0.33	**0.67**	0.40	0.33	0.66	0.39	0.32	0.66
	TextMAS	0.30	0.19	0.54	0.26	0.29	0.56	0.35	0.26	0.59	0.32	0.25	0.57
	Latent communication												
	Raw KV	0.42	0.31	0.60	0.47	0.47	0.58	0.45	0.37	0.64	0.45	0.36	0.62
	LatentMAS	0.32	0.23	0.53	0.38	0.40	0.62	0.41	0.35	0.63	0.39	0.33	0.61
	LatentDx	**0.72**	**0.54**	**0.72**	0.51	**0.59**	0.62	**0.67**	**0.66**	**0.76**	**0.64**	**0.62**	**0.72**
MediPhi 3.8B	*Non-collaborative*												
	Zero-shot	0.21	0.15	0.35	0.02	0.02	0.23	0.02	0.01	0.21	0.05	0.03	0.23
	Single RAG	0.49	0.29	0.53	0.44	0.39	0.56	0.45	0.35	0.64	0.46	0.33	0.61
	Text / structured sharing												
	Dist. RAG	0.54	0.39	0.63	0.58	**0.56**	0.67	0.52	0.45	0.68	0.54	0.43	0.67
	Global RAG	0.56	0.39	0.68	0.59	**0.56**	0.67	0.52	0.43	0.69	0.54	0.42	0.69
	Struct. RAG	0.54	0.37	0.61	0.57	0.51	**0.69**	0.52	0.43	0.69	0.53	0.41	0.68
	TextMAS	0.42	0.28	0.65	0.19	0.22	0.48	0.29	0.21	0.53	0.29	0.22	0.54
	Latent communication												
	Raw KV	0.56	0.39	0.63	0.52	0.53	0.67	0.49	0.42	0.66	0.51	0.41	0.66
	LatentMAS	0.58	0.39	0.67	0.48	0.49	**0.69**	0.51	0.44	0.69	0.51	0.42	0.69
	LatentDx	**0.63**	**0.42**	**0.72**	**0.60**	0.55	**0.69**	**0.54**	**0.46**	**0.70**	**0.57**	**0.45**	**0.70**

D.3 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

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

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 17 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-count Gini 0.095 and entropy 2.30 bits (out of a maximum of 2.32).

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 18 reports per-cohort utility under the skewed partition. `LatentDx` maintains strong diagnostic utility on both host backbones under the skewed partition. These results indicate that the learned interface remains effective when evidence is distributed non-uniformly across hospital retrieval pools.

Table 17: **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

Table 18: **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. **Best** and second best method per column, ranked separately for each backbone, with tied methods both marked.

Model	Method	RareBench			Zenodo			Phenopacket			Overall		
		Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge	Acc	F1	Judge
	Non-collaborative												
Qwen3	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
	LatentDx	**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	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
	LatentDx	**0.68**	**0.50**	**0.72**	**0.66**	**0.59**	**0.71**	**0.72**	**0.71**	**0.77**	**0.70**	**0.65**	**0.75**

D.4 Training-Seed Variability

Table 19 reports overall LatentDx-H diagnostic utility on the 401-case test set for Llama-3.2-3B and Qwen3-4B as the mean and sample standard deviation across three independent training runs with seeds 42, 43, and 44. Seed 42 is the main-experiment checkpoint. These results measure variation across training runs.

Table 19: **Diagnostic utility across training seeds.** Overall mean $\pm$ sample standard deviation for LatentDx-H across three training seeds.

Host	Acc	macro F1	Judge
Llama-3.2-3B	0.73 ± 0.01	0.67 ± 0.01	0.79 ± 0.02
Qwen3-4B	0.63 ± 0.03	0.59 ± 0.03	0.71 ± 0.01

D.5 Raw KV Communication Protocol

The main-table Raw KV follows the sequential relay protocol of LatentMAS. To examine the effect of this protocol choice, Parallel Raw KV uses the independent hospital encoding and direct-to-host communication pattern of LatentDx-H. Each hospital encodes its evidence segment (prompt B) in the shared system-prompt context and sends only that segment's KV states to the host. The host encodes the system prompt once and places the received blocks into contiguous positions by re-rotating their keys with the backbone's rotary module, leaving values unchanged. It then closes the user turn and appends the diagnosis question (prompt C). Blocks do not attend across hospitals.

Raw KV and Parallel Raw KV achieve similar accuracy and macro F1 on both hosts (Table 20). We retain the shared relay protocol in the main table.

Table 20: **Raw KV under two communication protocols.** Overall diagnostic utility on the full test set. Raw KV uses the main-table protocol (Table 1), while Parallel Raw KV encodes hospital prompts independently.

	Llama-3.2-3B			Qwen3-4B		
Method	Acc	F1	Judge	Acc	F1	Judge
Raw KV	0.35	0.32	0.58	0.59	0.48	0.72
Parallel Raw KV	0.37	0.31	0.53	0.59	0.48	0.69

D.6 SUPERVISED TEXT-SHARING REFERENCES

We evaluate `Dist. RAG` with prefix tuning and host-side LoRA as upper-bound-style utility references under direct access to retrieved clinical text. Both methods receive the retrieved disease labels and phenotype descriptions in plaintext, without a latent communication bottleneck, and are trained with diagnosis supervision. Table 21 reports overall diagnostic utility on both hosts.

Table 21: **Supervised text-sharing utility references.** Overall accuracy, macro F1 and judge score on the 401-case test set. The text-sharing references expose retrieved disease labels and phenotype descriptions to the host. `LatentDx-H` communicates compact latent blocks. Frozen `Dist. RAG` and `LatentDx-H` values are reproduced from Table 1.

	`Llama-3.2-3B`			`Qwen3-4B`		
Method	Acc	F1	Judge	Acc	F1	Judge
Text-sharing references						
`Dist. RAG` (frozen)	0.45	0.40	0.63	0.61	0.53	0.72
`Dist. RAG` + prefix tuning	0.72	0.63	0.78	0.66	0.63	0.73
`Dist. RAG` + host tuning (LoRA)	0.76	0.68	0.79	0.69	0.65	0.75
Latent communication						
`LatentDx-H`	0.73	0.68	0.77	0.64	0.61	0.72

Data and training. Both references use the same 7,220 training queries and `Dist. RAG` prompts, with the top-ranked case from each of the three participating hospitals and the query case itself excluded from retrieval. Training runs for five epochs with cross-entropy applied only to the target answer sequence, including the disease name, enclosing `<answer>` tags, and end-of-turn token. The remaining optimization settings follow Appendix B.3.3. Each host uses its own chat template, with the default date in the `Llama` template fixed across training and evaluation. Test cases, answer parsing, and evaluation metrics are unchanged.

Prefix tuning. Using PEFT 0.18, we train a shared 102-position KV prefix at every attention layer of the frozen host, without a reparameterization MLP (Li & Liang, 2021). This matches the $N(m+2) = 3 \times 34$ KV-prefix positions of `LatentDx-H`. The prefix is shared across queries and initialized from the first 102 positions of the frozen host's KV cache for the first training sample's chat-formatted system and user prompt. The trainable parameter counts are 5.85M for `Llama-3.2-3B` and 7.52M for `Qwen3-4B`.

Host tuning. We train LoRA adapters (Hu et al., 2022) using LLaMA-Factory (Zheng et al., 2024), with rank 16, scaling parameter 32, dropout 0.05, and adapters on all linear layers. The base-model weights remain frozen. LoRA trains 24.3M and 33.0M parameters for the two hosts, respectively, exceeding the corresponding `LatentDx-H` distillers' 9.45M and 6.56M parameters.

Utility and exposure. For an observer of the transmitted messages, all transmitted case content is directly readable, corresponding to 100% exposure by construction. `LatentDx-H` achieves competitive diagnostic utility, with overall accuracy only 3 and 5 percentage points below the LoRA reference on `Llama-3.2-3B` and `Qwen3-4B`, respectively, while substantially reducing the recoverability of private clinical information under the evaluated attacks (Appendix F).

E Representation Analysis and Ablations

E.1 Trained Bottleneck Control

We compare autoregressive latent distillation with `MLP-Pool32`, a diagnosis-supervised, length-matched control. It mean-pools the contextualized prompt hidden states and uses an MLP to produce $m = 32$ latent input embeddings. Both interfaces use diagnosis cross-entropy supervision and the same latent-position budget. Table 22 reports overall diagnostic utility and three learned attacks on both backbones. The pooling-control attacks use the same interface checkpoint as its utility evaluation and follow the data splits, attack budgets, and three-seed reporting protocol of the corresponding learned attacks in Appendix F. `MLP-Pool32` supports diagnosis but falls below `LatentDx-H` on all three utility metrics. It also exposes more disease information to the probe and inversion decoder, and greater retrieval-database membership leakage on both backbones.

Why can a compact pooled block still leak? One possible explanation is how the two interfaces construct their representations. Mean pooling aggregates contextualized information across the private prompt and may retain features that identify the retrieved disease, phenotypes, or record. The pooled code is then expanded into multiple latent positions. In contrast, `LatentDx-H` constructs its latent sequence recurrently through the frozen local backbone, with downstream diagnosis supervision shaping the generated states. This may organize the evidence differently from a one-shot pooled summary, making private record attributes less accessible to the evaluated attackers. Under matched diagnosis supervision and latent length, the observed utility–leakage difference suggests that representation construction matters for the communication interface.

Table 22: **Utility and leakage of the trained bottleneck control.** Both interfaces use diagnosis supervision and $m = 32$ latent positions. Top: overall diagnostic utility, higher is better. Bottom: learned-attack results on both backbones, averaged over three seeds, lower is better. The `LatentDx-H` attack values are reproduced from Table 3. **Best** within each comparison.

	`MLP-Pool32`			`LatentDx-H`		
Host	Acc	F1	Judge	Acc	F1	Judge
`Qwen3`	0.42	0.35	0.61	**0.64**	**0.61**	**0.72**
`Llama-3.2`	0.56	0.50	0.71	**0.73**	**0.68**	**0.77**

		`MLP-Pool32`		`LatentDx-H`	
Attack	**Metric**	`Qwen3`	`Llama-3.2`	`Qwen3`	`Llama-3.2`
Attribute probe	OMIM exact ↓	0.59	0.37	**0.23**	**0.20**
Inversion decoder	OMIM exact ↓	0.91	0.93	**0.40**	**0.38**
Membership inference	TPR@1%FPR ↓	0.95	0.38	**0.17**	**0.03**

E.2 Training-Matched Hospital-Count Control

To assess the diagnostic benefit of hospital evidence under supervision, we evaluate a query-only baseline ($N = 0$) and training-matched `LatentDx-H` controls on `Qwen3-4B` and `Llama-3.2-3B`. All $N = 0, 1, 2, 3$ configurations are trained separately with diagnosis supervision. The query-only baseline receives no hospital evidence. For $N \geq 1$, each interface is evaluated with the same number of hospital agents used during training. These interfaces use the main experiment's recipe: diagnosis cross-entropy only, $m = 32$ latent positions, learned boundary embeddings, cosine retrieval, and frozen local and host backbones. Only the number of hospitals in each training batch is fixed to the corresponding N. These separately trained controls complement the inference-only hospital-count sweep in Figure 5(a). Table 23 reports overall diagnostic utility on all 401 test cases, with the main-table $N = 3$ result as reference.

Table 23: **Supervised controls for hospital collaboration.** Overall diagnostic utility on the 401-case test set. All N configurations are trained separately with diagnosis supervision. The $N = 0$ baseline uses only the query, without hospital evidence. For $N \geq 1$, training and evaluation use the indicated hospital count. The $N = 3$ row reproduces the main-experiment reference.

	`Qwen3-4B`			`Llama-3.2-3B`		
N	Acc	F1	Judge	Acc	F1	Judge
0	0.41	0.38	0.51	0.64	0.61	0.69
1	0.47	0.41	0.59	0.69	0.64	0.73
2	0.56	0.52	0.66	0.70	0.66	0.76
3	0.64	0.61	0.72	0.73	0.68	0.77

On both backbones, the three-hospital system exceeds both the query-only and single-hospital supervised controls on all three overall metrics. The two-hospital control also improves on the single-hospital control, supporting the utility of combining evidence across hospitals.

E.3 Retrieval and Training-Data Sensitivity

Figure 5(c) plots judge score against retrieval depth for three representative methods, with the `LatentDx` interface trained separately for each k. Table 24 gives the full comparison across all metrics and every baseline. `LatentDx` achieves higher judge scores with greater retrieval depth.

Table 24: **Retrieval depth.** Acc / macro F1 / Judge on the `Llama-3.2-3B` host as the number of cases retrieved per hospital varies. **Best** per column.

Method	Top-1	Top-2	Top-3
`Dist. RAG`	0.45 / 0.40 / 0.63	0.54 / 0.50 / 0.66	0.55 / 0.50 / 0.64
`Global RAG`	0.47 / 0.42 / 0.62	0.51 / 0.47 / 0.66	0.52 / 0.46 / 0.69
`Struct. RAG`	0.42 / 0.34 / 0.67	0.50 / 0.43 / 0.70	0.52 / 0.44 / 0.69
`TextMAS`	0.32 / 0.25 / 0.47	0.40 / 0.33 / 0.60	0.42 / 0.36 / 0.62
`Raw KV`	0.35 / 0.32 / 0.58	0.43 / 0.36 / 0.60	0.45 / 0.37 / 0.61
`LatentMAS`	0.33 / 0.31 / 0.59	0.33 / 0.30 / 0.57	0.37 / 0.31 / 0.59
`LatentDx`	**0.73 / 0.68 / 0.77**	**0.74 / 0.68 / 0.78**	**0.74 / 0.67 / 0.79**

Figure 5(d) varies the fraction of the training split used to fit the latent interface. Validation and test sets are unchanged. At 25%, 50%, and 100% of the training split, respectively, accuracy / macro F1 / judge score is $0.53/0.48/0.64$, $0.66/0.59/0.71$, and $0.73/0.68/0.77$. Performance rises with more data and has not saturated at the full split.

E.4 COMPONENT DESIGN

We further isolate three interface choices on `Llama-3.2-3B`. The distiller and boundary-embedding studies use the default same-backbone configuration. The aggregation study uses the heterogeneous Degree 1 L:Q:L configuration, with latent-block size, training data, and optimization held fixed. Table 25 groups matched comparisons by component.

Table 25: **Component ablations.** Distiller-architecture and boundary-embedding results use the default same-backbone `LatentDx` setting with a `Llama-3.2-3B` host. Host-side aggregation is evaluated under the heterogeneous Deg1 configuration (L:Q:L). Comparisons within each component block hold latent-block size, training data, and optimization fixed. Best values are highlighted, with ties marked jointly.

Variant	Acc	Macro F1	Judge
Distiller architecture			
`Linear`	0.72	0.68	0.75
`LN + Linear` (ours)	0.73	0.68	0.77
`2-layer MLP` (`Linear`→`GELU`→`Linear`)	**0.74**	**0.69**	**0.79**
Boundary embeddings			
`Off`	**0.73**	**0.68**	0.76
`On` (ours)	**0.73**	**0.68**	**0.77**
Host-side aggregation			
`Mean pooling`	0.52	0.44	0.60
`Concatenation` (ours)	**0.72**	**0.66**	**0.76**

The two-layer MLP yields only marginal gains over LayerNorm followed by one linear projection, so we retain the lighter default distiller. Removing the boundary embeddings leaves accuracy and macro F1 unchanged and changes judge score by only 0.01, consistent with their intended role of delimiting hospital-specific blocks. Concatenation substantially outperforms position-wise mean pooling, supporting distinct hospital-specific segments for host attention.

F PRIVACY AND LEAKAGE

F.1 ADDITIONAL RECONSTRUCTION LEAKAGE RESULTS

Same-backbone attack and metric. We reuse the frozen-LLM continuation attack from §3.1. For `Raw KV` and `LatentMAS`, the attack input is the KV segment associated with hospital i's local prompt, including its two additional latent positions for `LatentMAS`, rather than the full cumulative relay cache. For `LatentDx-H`, the attack input is the complete transmitted block $\mathcal{T}_{H_i}$. The observer injects this input 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 $\tilde{x}_i$ of up to 192 tokens. The reference is the user-turn text of the source local prompt x_i, including the retrieved case, query phenotypes, and instructions. We compute BERTScore F1 (Zhang et al., 2020) using `RoBERTa-large` layer-17 embeddings without IDF weighting and enable the package's built-in baseline rescaling (`rescale_with_baseline=True`). The rescaled score is $\hat{s}_F = (s_F - b_F)/(1 - b_F)$, where s_F is the unrescaled F1 and $b_F \approx 0.8312$ is the package's precomputed English baseline from random Common Crawl sentence pairs. This baseline is independent of our data and shared across all evaluated backbones. Each cell averages 2,005 reconstruction–reference pairs from 401 test cases across five hospital databases.

Table 26: **Rescaled BERTScore F1 for KV reconstruction leakage.** Lower is better ($n = 2{,}005$ prompt pairs per cell). Zero corresponds to the package's English baseline, and negative values fall below it. **Best** per row.

Model	Raw KV	LatentMAS	LatentDx
Qwen3	0.593	0.593	**−0.125**
Llama-3.2	0.673	0.812	**−0.322**
MediPhi	0.724	0.712	**−0.308**

Results. Table 26 shows that both raw-latent baselines leak substantial source content. `Raw KV` reaches rescaled BERTScore between 0.59 and 0.72, exposing the hospital's prompt-position states and enabling the model to largely paraphrase the source 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, `LatentDx` attains negative rescaled BERTScore on all three backbones, ranging from −0.125 to −0.322. These scores fall below the package's English baseline. The comparison with `Raw KV` and `LatentMAS` shows that compact distillation reduces reconstruction similarity under this evaluated attack.

F.2 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 recovery of private clinical attributes.

We first evaluate two prompt-based extraction attacks on same-backbone transmissions. The first is the verbatim continuation attack used in Sec. 3.1. The second is a template-aware attack: the attacker is given the hospital-case template and is explicitly asked to recover the retrieved disease label and phenotype terms. Appendix B.1.7 gives the exact prompts in H1 and H2, respectively. Both attacks use the hospital-specific input defined in Appendix F.1, injected into the frozen LLM that produced it. No attacker model is trained.

Table 27: **Attribute-level leakage under prompt-based extraction attacks.** Disease recall is the fraction of cases in which the retrieved disease label appears in the attacker's continuation. 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. **Best** method per column, ranked separately for each backbone.

		Verbatim		Template-aware	
Model	**Method**	Disease recall ↓	Phenotype ↓	Disease recall ↓	Phenotype ↓
Qwen3	Raw KV	1.00	1.00	0.89	0.86
	LatentMAS	1.00	1.00	0.90	0.85
	LatentDx	**0.02**	**0.01**	**0.20**	**0.14**
Llama-3.2	Raw KV	0.99	1.00	0.88	0.97
	LatentMAS	0.98	0.99	0.75	0.89
	LatentDx	**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
	LatentDx	**0.04**	**0.03**	**0.05**	**0.04**

Table 27 shows that Raw KV and LatentMAS reveal most retrieved disease labels and phenotype terms under both prompt variants. Template information increases recovery from LatentDx on Qwen3-4B, where disease leakage reaches 0.20 versus 0.89–0.90 for the raw-latent baselines, and phenotype recall reaches 0.14 versus 0.85–0.86. Across backbones, LatentDx keeps disease leakage at or below 0.20 and phenotype recall at or below 0.14. The leakage reduction extends to retrieved disease labels and phenotype terms under both prompt variants.

Disease disclosure under the standard diagnosis prompt. We also test whether an intercepted hospital block reveals the retrieved disease when used for ordinary diagnosis. The attacker supplies one hospital's block to the same frozen host as a KV prefix, appends the standard diagnosis prompt C from Appendix B.1.7, and greedily decodes up to 192 tokens without attacker training. Local prompts use top-1 cosine retrieval and are left-truncated to 320 tokens. LatentDx-H supplies its complete 34-position block. For this test, Raw KV supplies the full KV cache of an independently encoded local prompt. The no-transmission control uses the same diagnosis prompt without a KV prefix.

From the 2,005 query–hospital pairs (401 test queries across five hospitals), we report the 752 pairs where the retrieved disease differs from the query's gold diagnosis. A mention is counted when the retrieved disease name appears as a substring of the output after lowercasing and removing punctuation from both strings. Table 28 shows substantially lower retrieved-disease mention rates for LatentDx-H than for Raw KV on both backbones.

Table 28: **Disease disclosure under the standard diagnosis prompt.** Retrieved-disease mention rate on 752 query–hospital pairs with different retrieved and query diagnoses. Lower is better.

Intercepted object	Llama-3.2-3B	Qwen3-4B
No transmission	0.009	0.009
Raw KV	0.285	0.608
LatentDx-H	0.064	0.045

F.3 Learned Attacks and Full Evaluation Matrix

Table 29: **Same-backbone and cross-family leakage, grouped by receiving host.** This table extends Table 3 with `MediPhi`, additional reconstruction and phenotype metrics, and a five-attack evaluation of `LatentDx-X`. BERTScore F1 is unrescaled. H uses the same sender and host backbone. X uses the sender-to-host direction shown in each block. X attacks operate on host-space KV reconstructed from the transmitted hidden states (Appendix F.4). No tx., `Raw KV`, and `LatentMAS` are the same-backbone controls. No tx. receives side information but no transmitted representation. Learned-attack entries for both H and X are means over three seeds. H reduction is the relative drop from `Raw KV` to `LatentDx-H`. Dashes denote unreported X results. Lower is better, but metrics are not comparable across rows.

Attack	Metric	No tx.	Raw KV	LatentMAS	LatentDx H	LatentDx X	H reduction
Qwen3-4B host; X: Llama-3.2-3B → Qwen3-4B							
Prompt reconstruction	Token F1 ↓	0.05	0.60	0.60	0.07	0.032	↓88%
Prompt reconstruction	BERTScore F1 ↓	0.78	0.93	0.93	0.81	–	↓13%
Template-aware extraction	Disease recall ↓	0.00	0.89	0.90	0.20	0.173	↓78%
Attribute probe	OMIM exact ↓	0.17	0.62	0.62	0.23	0.250	↓63%
Attribute probe	HPO micro-F1 ↓	0.05	0.33	0.33	0.14	–	↓58%
Inversion decoder	OMIM exact ↓	0.27	0.97	0.96	0.40	0.280	↓59%
Inversion decoder	HPO micro-F1 ↓	0.21	0.90	0.90	0.22	–	↓76%
Membership inference	TPR@1%FPR ↓	0.00	0.99	0.93	0.17	0.030	↓83%
Llama-3.2-3B host; X: Qwen3-4B → Llama-3.2-3B							
Prompt reconstruction	Token F1 ↓	0.07	0.68	0.82	0.07	0.094	↓90%
Prompt reconstruction	BERTScore F1 ↓	0.78	0.95	0.97	0.78	–	↓18%
Template-aware extraction	Disease recall ↓	0.00	0.88	0.75	0.16	0.090	↓82%
Attribute probe	OMIM exact ↓	0.17	0.61	0.60	0.20	0.280	↓67%
Attribute probe	HPO micro-F1 ↓	0.05	0.32	0.32	0.13	–	↓59%
Inversion decoder	OMIM exact ↓	0.27	0.99	0.99	0.38	0.410	↓62%
Inversion decoder	HPO micro-F1 ↓	0.23	0.87	0.87	0.27	–	↓69%
Membership inference	TPR@1%FPR ↓	0.00	1.00	0.83	0.03	0.120	↓97%
MediPhi host (same-backbone only)							
Prompt reconstruction	Token F1 ↓	0.07	0.76	0.75	0.07	–	↓91%
Prompt reconstruction	BERTScore F1 ↓	0.77	0.95	0.95	0.78	–	↓18%
Template-aware extraction	Disease recall ↓	0.00	0.76	0.77	0.05	–	↓93%
Attribute probe	OMIM exact ↓	0.17	0.55	0.62	0.22	–	↓60%
Attribute probe	HPO micro-F1 ↓	0.05	0.30	0.31	0.10	–	↓67%
Inversion decoder	OMIM exact ↓	0.27	0.95	0.93	0.40	–	↓58%
Inversion decoder	HPO micro-F1 ↓	0.24	0.84	0.84	0.25	–	↓70%
Membership inference	TPR@1%FPR ↓	0.00	0.79	0.94	0.02	–	↓97%

Side information and controls. We evaluate three learned attacks on same-backbone `LatentDx-H` representations produced by each of `Qwen3-4B`, `Llama-3.2-3B`, and `MediPhi`, together with the corresponding baselines. The attacker passively observes the transmitted representation, knows the participating backbone, prompt template, and retrieval procedure, and may train on auxiliary examples from the same distribution, but it has no access to the target hospital database or the target retrieved record in clear text at test time. In addition to the transmission, the probe and inversion decoder receive the broadcast query HPO set, while membership inference receives a candidate case and target-hospital identifier. The no-transmission control for each attack receives the same side information but no transmitted representation, providing a side-information-only reference for assessing the additional recoverability from transmission. The table entries remain absolute metric values, and Reduction is computed directly from `Raw KV` to `LatentDx`.

Shared representation encoder. All learned attacks use the same representation-encoder design across communication methods, while each backbone–method–attack combination is trained independently from scratch. At each backbone layer, key and value states are concatenated and mapped

through a learned layer-specific bottleneck. The resulting position representations are combined across layers, projected to a common width, augmented with learned positional embeddings and a [CLS] token, and processed by a pre-LN Transformer encoder. Side information is encoded as an additional token after [CLS]. The attribute probe and membership classifier use the [CLS] state, whereas the inversion decoder cross-attends to the full encoded sequence. For Raw KV and LatentMAS, the encoder receives hospital i's prompt-position segment (up to 320 positions), plus its two latent positions for LatentMAS, rather than the full cumulative relay cache. The probe and inversion decoder target the retrieved record in x_i, while membership inference targets the candidate's membership in $\mathcal{D}_i$. For LatentDx-H, the encoder receives all 34 transmitted positions.

For LatentDx-H, the attack input comprises all $m + 2 = 34$ transmitted KV positions: the begin boundary, $m = 32$ latent positions, and the end boundary. The begin-boundary KV is conditioned on the private prompt, and the end-boundary KV additionally incorporates the generated latent sequence.

Attribute probe. Linear readout heads on the pooled representation predict the retrieved case's OMIM class and HPO terms. We train the disease head with softmax cross-entropy and the phenotype head with frequency-weighted binary cross-entropy. The probe inherits the case-level train, validation, and test partitions, so evaluation queries are disjoint from attacker training. Checkpoints and multilabel decision thresholds are selected only on validation data and then fixed for test evaluation. We report OMIM exact match and HPO micro-F1.

Autoregressive inversion. A Transformer encoder–decoder reconstructs a serialized record by cross-attending to the full encoded transmission. Its target contains only the retrieved OMIM identifier and HPO terms, excluding query fields and template boilerplate. We train with teacher forcing and an up-weighted end-of-sequence term, restore the best validation checkpoint, and use greedy decoding at test time. We evaluate the decoded identifier by OMIM exact match and the recovered phenotype terms by HPO micro-F1.

Retrieval-database membership inference. Here the secret is whether a candidate case belongs to a hospital's retrieval database. For membership inference, the observed transmission is generated using the candidate case's unchanged HPO set as the broadcast query. For each candidate, we construct one positive example with its member hospital and one hard negative with the non-member hospital that assigns the highest retrieval score. Candidate content is fixed within the pair, while the target hospital and its intercepted transmission vary. We split candidates $60/20/20$ into train, validation, and test sets, stratified by member hospital and disjoint by case identifier. Both examples for a candidate remain in the same split. A binary classifier operates on the pooled transmission representation together with candidate and hospital embeddings. We select checkpoints by validation ROC-AUC and report test TPR at 1% FPR.

Training and reporting. We optimize every attacker with AdamW, linear warmup followed by decay, gradient clipping, early stopping, and restoration of the best validation checkpoint. Each same-backbone method–attack combination, including its no-transmission control, is trained with three random seeds, and its entries in Tables 3 and 29 report the mean across seeds.

Table 3 reports the full attack ladder on Qwen3-4B and Llama-3.2-3B. Table 29 extends this matrix with MediPhi, the unrescaled BERTScore row, HPO micro-F1 for the learned probe and inversion decoder, and the cross-family results described below. Across the three backbones, LatentDx-H reduces OMIM exact-match leakage by 58–67%, HPO micro-F1 by 58–76%, and membership-inference TPR at 1% FPR by 83–97% relative to Raw KV. Its membership-inference TPR remains between 0.02 and 0.17, compared with 0.79–1.00 for Raw KV. Together with the training-free attacks, these results show a consistent empirical reduction in recoverable clinical information across all three evaluated same-backbone settings. These results characterize empirical leakage under the evaluated attacks, not formal privacy guarantees.

F.4 CROSS-FAMILY LEAKAGE EVALUATION

Threat model and attacked representation. We evaluate `LatentDx-X` under the same five attacks using a white-box host-side insider. The adversary observes the transmitted sequence of $m = 32$ last-layer hidden states from the hospital model and knows the trained cross-family projector, host boundary embeddings, and frozen host backbone. It applies the projector, prepends and appends the host's $\langle \text{BOP} \rangle$ and $\langle \text{EOP} \rangle$ embeddings, and runs a host forward pass to materialize a block of $m + 2 = 34$ KV positions. Each layer's key and value tensors have shape $8 \times 34 \times 128$, with 36 layers for `Qwen3-4B` and 28 for `Llama-3.2-3B`. This derived block has the same tensor structure as the corresponding `LatentDx-H` attack input, allowing the same attacker implementations to be reused. The network payload consists of sender hidden states, and all five attacks use the host-space KV block derived from them.

Directions and protocol. We evaluate `Llama-3.2-3B` hospitals sending to a `Qwen3-4B` host and the reverse direction. Each sender uses its family's trained `LatentDx-H` distiller, followed by the stage-two projector trained for that sender–host pair. We retain the attack architectures, data splits, side information, training budgets, and decision rules from the same-backbone evaluation, training the learned attackers separately for each direction. Each direction–learned-attack combination is trained with three random seeds. Table 29 reports their mean. Prompt-based attacks decode from the derived KV using the receiving host backbone. The five reported metrics target the private source prompt, its retrieved disease label, or retrieval-database membership, as above.

Results. Table 29 places these results beside `LatentDx-H`, grouped by receiving host and with the X direction explicitly indicated. For `Llama-3.2-3B`→`Qwen3-4B`, inversion OMIM exact match is 0.28 and membership TPR at 1% FPR is 0.03, compared with 0.40 and 0.17 for the corresponding H setting. In the reverse direction, these values are 0.41 and 0.12, versus 0.38 and 0.03 for H. Cross-family leakage varies with the sender–host direction and attack.