FEDAGENTBENCH: TOWARDS AUTOMATING REAL-WORLD FEDERATED MEDICAL IMAGE ANALYSIS WITH SERVER–CLIENT LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning (FL) allows collaborative model training across healthcare sites without sharing sensitive patient data. However, real-world FL deployment is often hindered by complex operational challenges that demand substantial human efforts in cross-client coordination and data engineering. This includes: (a) selecting appropriate clients (hospitals), (b) coordinating between the central server and clients, (c) client-level data pre-processing, (d) harmonizing non-standardized data and labels across clients, and (e) selecting FL algorithms based on user instructions and cross-client data characteristics. However, the existing FL works overlook these practical orchestration challenges. These operational bottlenecks motivate the need for autonomous, agent-driven FL systems, where intelligent agents at each hospital client and the central server agent collaboratively manage FL setup and model training with minimal human intervention. To this end, we first introduce: (i) an agent-driven FL framework that captures key phases of real-world FL workflows from client selection to training completion, and (ii) a benchmark dubbed FedAgentBench that evaluates the ability of LLM agents to autonomously coordinate healthcare FL. Our framework incorporates 40 FL algorithms, each tailored to address diverse task-specific requirements and cross-client characteristics. Furthermore, we introduce a diverse set of complex tasks across 201 carefully curated datasets, simulating 6 modality-specific real-world healthcare environments, *viz.*, Dermatoscopy, Ultrasound, Fundus, Histopathology, MRI, and X-Ray. We assess the agentic performance of 14 open-source and 10 proprietary LLMs spanning small, medium, and large model scales. While some agent cores such as GPT-4.1 and DeepSeek V3 can automate various stages of the FL pipeline, our results reveal that more complex, interdependent tasks based on implicit goals remain challenging for even the strongest models.

1 INTRODUCTION AND BACKGROUND

Federated Learning (FL) (Li et al., 2021b; McMahan et al., 2017; Li et al., 2020a) allows collaborative model training across multiple healthcare institutions (*e.g.*, hospitals) without sharing raw medical data. A typical FL workflow involves several tightly coupled components: selecting suitable clients for training, preprocessing heterogeneous data locally, harmonizing labels and datasets across clients, coordinating client-server communication, selecting optimal FL algorithm, and aggregating model updates in the server. These components must be executed in a precise and orchestrated manner across multiple clients. Real-world execution of an FL pipeline necessitates close coordination by human data scientists and machine learning engineers in server and client locations, who are tasked with managing a range of demanding communicational and technical operations. These include selecting appropriate client nodes based on task relevance and resource availability, implementing local data preprocessing pipelines (e.g., normalization, filtering, schema mapping), and harmonizing cross-site inconsistencies of data and label spaces. Additionally, they must determine the most suitable FL algorithms, and manage training schedules and aggregation strategies. This manual orchestration poses a significant barrier to scalable and real-time deployment of FL, particularly in sensitive domains like healthcare, where institutions store diverse yet complementary datasets that cannot be centralized due to privacy and compliance constraints. Moreover, many healthcare facilities, especially in low- and middle-income countries (LMICs) and rural areas, lack the resources to hire dedicated data scientists or machine learning engineers, further limiting their ability to participate

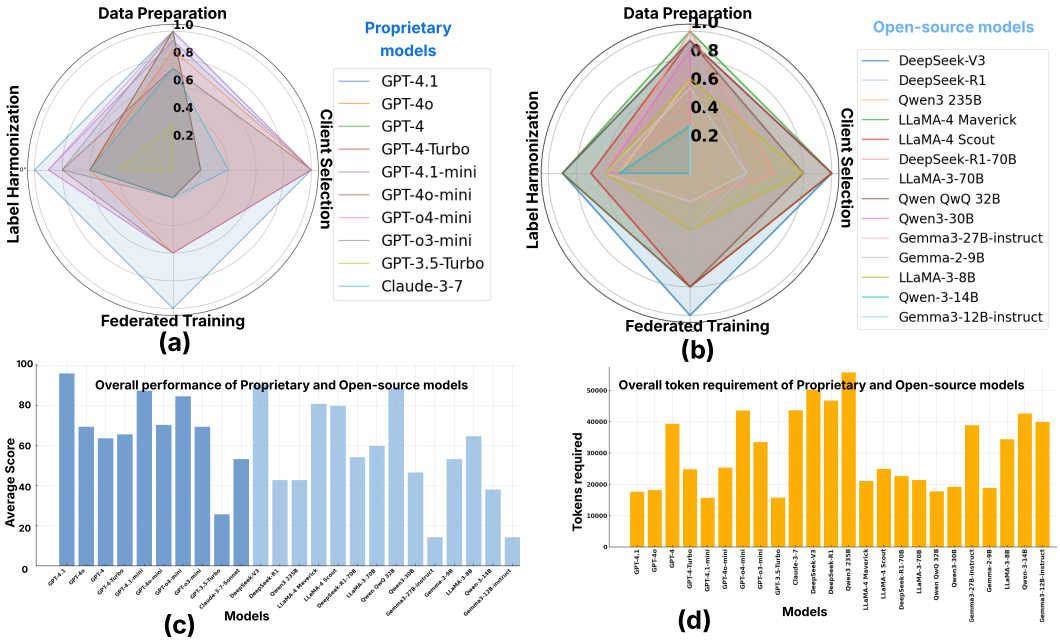


Figure 1: Performance of 24 LLM Agents on 4 FL sub-tasks over 6 healthcare environments. (a) and (b) show the performance of proprietary and open-source models respectively on four subtasks each, *viz.*, Client Selection, Data preprocessing, Label Harmonization, and Federated Training. (c) and (d) show the average score and mean overall token requirement of all models across all tasks.

in FL initiatives despite having valuable local data. To this end, in this paper, we investigate the capabilities of LLM Agents in tackling these issues with minimal human intervention.

The rapid advancement of LLMs has led to the emergence of autonomous AI agents capable of executing complex, multi-step tasks across various domains (Gur et al.; Gou et al.; Cai et al.; Li et al., 2023a; Wang et al., 2023; Wu et al.; Mei et al., 2024; Chu et al., 2025; Qiu et al., 2024; Luo et al., 2025). This capability can be particularly transformative for real-world healthcare FL, where agent-based automation can reduce the operational burden on healthcare sites and enable broader participation in collaborative AI development. There are no existing works on agent-driven FL workflow; for general-purpose agents or agentic FL works, refer to **Related Works in Appendix A**.

To this end, we introduce an agentic FL framework (see Figs. 2 & 3) along with a benchmark **FedAgentBench** (see Fig. 1), designed to systematically evaluate the performance of LLM-driven agents in orchestrating FL workflows. To ensure comprehensive coverage, we incorporate 201 datasets, 6 major medical imaging modalities, and 40 representative FL algorithms designed for diverse real-world healthcare objectives and cross-client data compositions. To the best of our knowledge, this is the first work addressing FL problem-solving capabilities of LLM Agents directly dealing with server and client interactions. Our benchmark makes the following key contributions:

(1) Technical contribution: We first **present a plug-and-play modular agentic FL framework** supporting 40 FL algorithms and 24 LLM agents. It also allows for easy integration of new FL algorithms, agents and tasks with minimal adaptation. It is a unified FL framework with multi-faceted scenarios, multiple imaging modalities, and complex FL workflow structures. It encompasses four realistic and interlinked agent-driven FL phases: (i) **Client Selection**, where server and client agents communicate dataset suitability, (ii) **Data Preprocessing**, involving data restructuring, cleaning, and standardization using learned tools, (iii) **Label Harmonization**, where agents align inconsistent label taxonomies across clients, and (iv) **Federated Model Training**, where selected algorithms are deployed in a decentralized setup. It is worth noting that while we simulate healthcare environments in this work, the framework can be readily extended to other FL settings such as finance, IoT, etc.

(2) Dataset and Task contribution: To evaluate the effectiveness of LLM agents in real-world healthcare tasks, we construct a realistic simulation of inter-hospital collaboration within a FL framework in representative clinical scenarios. Specifically, we curate and publicly release **six medical imaging FL agentic environments** comprising a total of **201 datasets** and a diverse collection of tasks spanning a range of difficulties. To introduce greater variability across clients, we systematically modify the original image resolutions, file format extensions, and intensity distributions

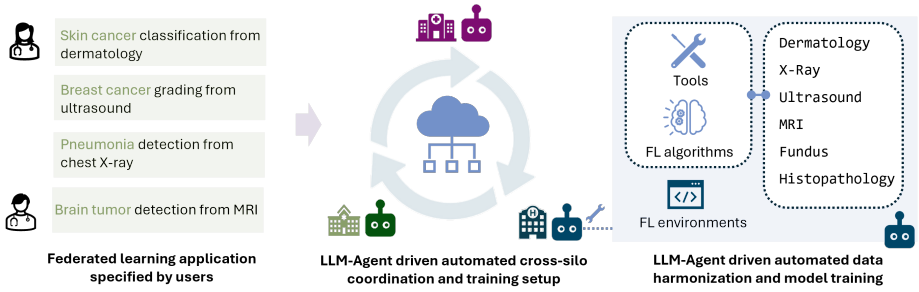


Figure 2: Overview of our agent-driven FL setup. First, user defines task specification. Accordingly, LLM agents perform server-client coordination and complete required tasks using available tools and FL algorithms in any of the 6 modality-specific healthcare environments.

of the client datasets. Additionally, we carefully inject noisy and irrelevant samples spanning images from other modalities, text files, and other extraneous formats into client data directories to simulate realistic uncurated data environments and reflect the challenges of real-world clinical settings.

(3) Empirical contribution: As a part of FedAgentBench, we evaluate the performance of 24 LLM agents across diverse FL tasks based on task completion rate (*i.e.*, success rate), token efficiency, and overall time required. We investigate how varying levels of prompt granularity affect task execution and systematically compare agent performance across different autonomy tiers: guided tool invocation, autonomous planning, and fully independent script generation. Our analysis provides a comprehensive assessment of agentic capabilities and limitations in supporting real-world collaborative healthcare workflows. We will open-source and continuously update the benchmark on Github repository to support FL research and help healthcare data holders fully realize the value of cross-silo data.

Research Questions. FedAgentBench is designed around 5 central research questions that capture the core operational challenges faced by LLM agents in FL workflows (Detailed in §3.2 and 3.3):

RQ1: Are there particular phases of the FL workflow that are especially challenging for LLM agents? How does LLM agent performance vary across different phases of the workflow? A1: Across 24 models, we observe a consistent difficulty hierarchy: *Label Harmonization* > *Data Preprocessing* > *Federated Training* > *Client Selection* with harmonization emerging as the dominant bottleneck due to its need for multi-hop semantic alignment across heterogeneous client taxonomies.

RQ2: What role does the granularity of prompts or instructions play in how reliably agents complete different steps of the workflow? A2: Fine-grained, structured prompts substantially increase success rates, especially for the complex semantic phases. By contrast, goal-oriented prompts often lead to reasoning drift, skipped steps, and hallucinated structures.

RQ3: To what extent can we rely on scale alone to predict how well an agent will perform? Does choosing a larger LLM translate into more dependable agent behaviour? A3: Empirically, model scale is not a reliable predictor of performance. Several mid-sized models (e.g., Qwen QwQ-32B, LLaMA-4 Scout) outperform much larger models, indicating that instruction-following ability and architectural grounding outweigh parameter count.

RQ4: Do challenging real-world Federated Learning subtasks such as label harmonization and data preprocessing expose systematic weaknesses in current LLM agents? A4: Yes, these tasks consistently surface systematic failure modes including misaligned label mappings, multi-step workflow collapse, speculative reasoning, and poor grounding in tool outputs and workspace structure.

RQ5: How pronounced is the difference in performance between proprietary and open-source agents across the FL workflow? A5: The performance gap exists but is phase-dependent: proprietary models excel in the hardest stages (preprocessing, harmonization), while strong open-source models often match or exceed them in simpler stages (client selection, training initiation).

2 FedAgentBench Framework

2.1 Problem Formulation and Overview

Given a user-defined task specification for federated medical image analysis, denoted as $\mathcal{T}$, our objective is to construct and execute a complete FL pipeline through collaborative decision-making by a set of autonomous agents. As outlined in Fig. 3, FedAgentBench consists of two main components:

(i) Federated medical imaging workspace $\mathcal{W}$ which can be sub-categorized to server workspace $\mathcal{W}_s$ and client workspace $\mathcal{W}_c$ as well as **(ii) Multi-agent coordination system** $\mathcal{A}$. The workspace $\mathcal{W}$ encapsulates the critical resources required for FL pipeline construction and includes: (1) client metadata files (data cards) containing natural language descriptions of local datasets (in $\mathcal{W}_c$), (2) FL algorithm specifications (in $\mathcal{W}_s$) and tool usage descriptions (in $\mathcal{W}_c$ and $\mathcal{W}_s$) and (3) structured code templates for each phase of the FL workflow (in $\mathcal{W}_c$ and $\mathcal{W}_s$).

Built on top of this workspace, the agents operate under a divide-and-conquer strategy to address the complexity and modularity of the entire FL process. The server-client agent system $\mathcal{A} = \{S_1, S_2, S_3, S_4, C_1, C_2, C_3\}$ comprises 7 role-specialized LLM agents (see Fig. 3) responsible for: (1) client selection and server-client communication or orchestration (S_1, S_2, C_1), (2) data preprocessing and cleaning (C_2), (3) label harmonization (C_3), and (4) federated model selection and training (S_3, S_4). The collaborative pipeline proceeds iteratively as agents can invoke tools, write scripts, or reason over workspace content to solve subtasks, with execution feedback enabling adaptation. This process can be formally represented as: $\{D_i, R_i\} = \mathcal{A}(D_{i-1}, R_{i-1}, \mathcal{T} \mid \mathcal{W})$ where D_i denotes the code, decisions, or configurations generated or modified in the i-th iteration, and R_i represents execution results or tool feedback (e.g., logs, errors, evaluation metrics), with $D_0 = R_0 = \emptyset$. The goal is to produce a complete, executable FL pipeline satisfying task specification $\mathcal{T}$, measured in terms of success and efficiency under real-world constraints simulated by $\mathcal{W}$.

2.2 Client Dataset Curation and FL Algorithm Integration

Broad coverage of real-world medical specialties and data sets: We construct FedAgentBench clients by adapting **201 publicly available datasets** with 2D and 3D dimensionality across 6 different medical imaging modalities *viz.* **25 Dermatology, 33 Ultrasound, 63 Fundus, 32 X-Ray, 28 MRI, and 20 Histopathology datasets**. It spans a broad range of tasks, including disease classification (e.g., tumor detection, cancer subtype identification), disease staging or grading (e.g., cancer and diabetic retinopathy severity levels), anatomical or pathological region segmentation (e.g., tumor or stroke localization), object detection, regression, reconstruction, *etc*. Each client is simulated to comprise one or more of these datasets, reflecting the diversity and heterogeneity typical of real-world healthcare institutions. We construct a datacard accompanying each client based on the metadata sourced from its original publication, repository or website. **See Appendix C.1 & Listings 6-8**.

Cross-client data heterogeneity beyond distribution shifts: In order to introduce greater variability across clients and better emulate the heterogeneity found in real-world clinical data silos, we systematically modify several aspects of the original datasets:

(i) Structured Dataset Perturbations: We introduce systematic modifications to dataset characteristics, such as varying image resolutions (e.g., downsampling images), altering file format extensions (e.g., converting `.png` files to `.jpeg`, `.bmp`, or `.tiff`), and modifying intensity distributions to reflect differences in scanner settings or preprocessing pipelines.
(ii) Inclusion of Uncurated and Irrelevant Files: To reflect the messiness of real-world clinical storage, we inject non-image and unrelated files into client directories. These include textual notes (`.txt`, `.doc`, `.pdf`), spreadsheets (`.csv`, `.xls`), and miscellaneous files (e.g., `.log`, `.xml`, `.ini`). For example, our dermatoscopy dataset contains lesion images mixed with dermatologist notes in `.pdf` format and other unrelated documents.
(iii) Simulation of Label and Modality Noise: We simulate common data quality issues by introducing random duplication of 2-5 samples, injecting 2-5 anatomically or modality-inconsistent images, and deliberately corrupting labels of 2-5 samples to model annotation noise in each dataset.

These artifacts challenge the robustness of agent-based preprocessing and reflect the complexities encountered in real hospital PACS or data repositories. See Appendix C for more details.

Algorithm suite for a wide spectrum of FL settings: As a part of the benchmark design, we also curate a comprehensive suite of **40 FL algorithms** by integrating and adapting existing implementations. This algorithm collection spans a broad spectrum of FL paradigms enabling standardized and reproducible evaluation across diverse medical imaging tasks **(See Appendix §C.4)**. This includes:

(i) Classical FL algorithms such as `FedAvg`, `FedProx`, and `Scaffold`, which address baseline aggregation and client drift; **(ii) Personalized FL algorithms** like `Per-FedAvg`, `pFedMe`, and `FedRep`, which tailor models to heterogeneous client data distributions; **(iii) Regularization-based approaches** like `Ditto` which impose constraints to preserve global knowledge during local updates; **(iv) Knowledge Distillation-based methods** such as `FedDF`, enabling model-agnostic communica-

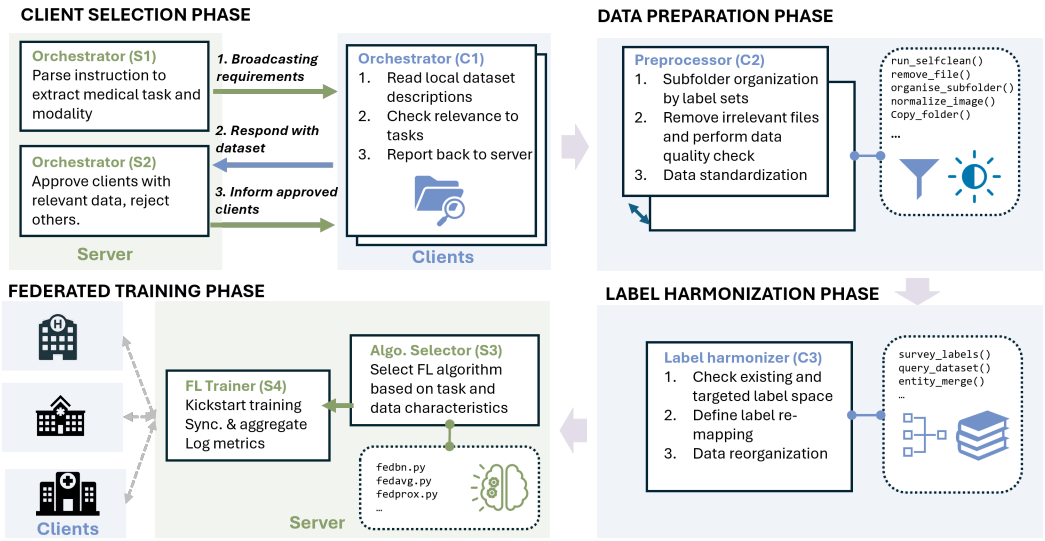


Figure 3: An overview of the FedAgentBench Framework. It comprises 7 role-specialized LLM agents $(S_1, S_2, S_3, S_4, C_1, C_2, C_3)$ for completing 4 distinct phases of the FL workflow (see §2.3)

tion via logits; **(v) Domain generalization techniques** like `FedSR`, `FedDG`, and `FedIRM`, which aim to learn invariant representations across non-IID clients; and **(vi) Optimization and scheduling variants**, such as `FedNova` which address stability, and convergence rate.

2.3 Federated Agentic Framework Construction

FL workflows typically follow a common set of phases, from which we abstract the key human roles and tasks fundamental to their execution as discussed below **(See Appendix B.2 for more details)**:

1. Client orchestrator agents: These agents act as the coordinators of the framework by communicating between the server and clients as well as by selecting the most suitable clients for the task based on the user requirements and individual client responses (see Fig. 4). Server agent S_1 interprets the user-defined task $\mathcal{T}$ and communicates imaging modality/task requirements to initiate client selection. For this, it first parses $\mathcal{T}$ and broadcasts a query to all Client Agents (*i.e.*, healthcare sites). Each Client Agent C_1 reads local dataset description file, which contains metadata about available datasets, including label sets/imaging types. Based on semantic and modality matching, C_1 evaluates relevance of its datasets to $\mathcal{T}$, returning only matching datasets (if any). Server Agent S_2 collects these responses and selects a subset of relevant clients $\mathcal{C}_{\text{sel}}$, which are then approved for further processing **(see Figs. 9-14 in Appendix D)**.

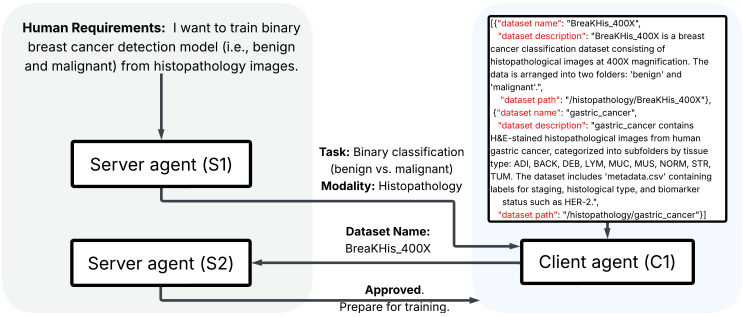


Figure 4: Client orchestrator agents S_1, C_1, and S_2 in a histopathology-based breast cancer classification task

2. Data pre-processor agent: It is responsible for preparing selected client datasets for effective participation in the FL pipeline. Given the diversity of data storage formats and quality issues across real-world sites, Data pre-processor agent C_2 at each client ensures that the dataset adheres to a standardized structure and meets minimum quality criteria. Concretely, it is responsible for standardizing and cleaning datasets at each selected client (see Fig. 5). This includes:

(i) Subfolder Organization: Verifies whether datasets are organized into class-specific subfolders. If disorganized, C_2 restructures the folder hierarchy.
(ii) File Cleaning: Removes irrelevant files (non-image formats `.txt`, `.csv` *etc.*) to ensure format consistency.
(iii) Data Cleaning: Detects and flags duplicates, off-topic samples, and noisy labels, which are

then removed. This ensures all selected clients have curated structurally consistent data, enabling downstream harmonization and consequent training **(see Figs. 35-36 in Appendix D)**.

(iv) Data Normalization/Standardization: Standardizes images across clients based on resolution, intensity, and file extension. This agent thus plays an essential role in bridging the gap between raw, heterogeneous clinical data and the clean, harmonized inputs required for FL. Its operations ensure that all participating clients contribute structurally consistent, high-quality data harmonized across clients, which is crucial for the success of the overall FL system.

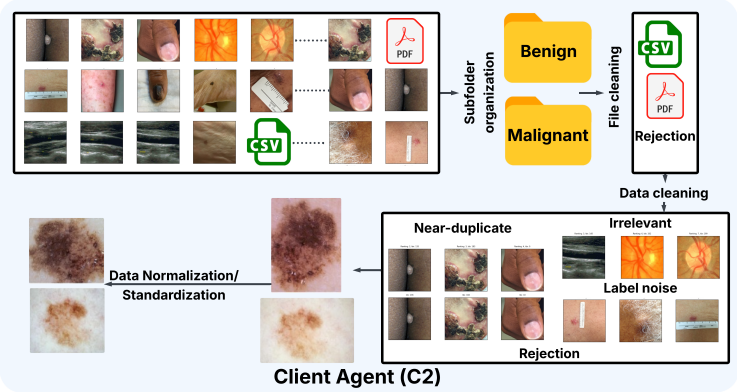


Figure 5: Data pre-processor agent C_2 in skin cancer detection task

3. Task-conditioned label harmonizer agent: This agent (C_3) addresses one of the most critical challenges in multi-institutional FL, *i.e.*, the inconsistency in label nomenclature and granularity across client datasets (see Fig. 6). Due to variations in annotation protocols, terminologies, and domain-specific taxonomies, class labels across clients may not align semantically or structurally. C_3 plays a pivotal role in reconciling these differences based on the user requirements: **(i) Class Inspection**: Enumerates all class labels present in client datasets.
(ii) Label Mapping: Converts fine-grained labels (e.g., `"melanoma"`, `"nevus"`) to harmonized classes (e.g., `"malignant"`, `"benign"`) according to a self-developed mapping schema.

(iii) Data Reorganization: Reorganizes the dataset structure to reflect harmonized labels, aligning image samples with their mapped class definitions. This standardization enables cross-client training without semantic conflicts in label interpretation.

Through these actions, the agent guarantees that all clients adhere to a shared label vocabulary.

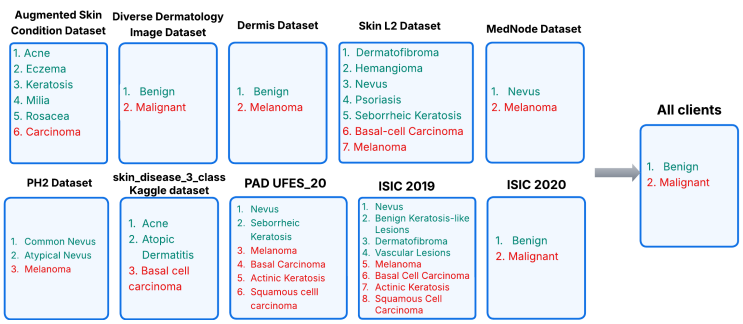


Figure 6: Label harmonization by agent C_3 in dermatology-based skin cancer detection (benign/malignant classes color-coded in green/red)

4. Federated trainer agents: These agents are responsible for initiating the actual federated training process across the selected set of clients and play a central role in converting the prepared environment into a functioning FL system. They initiate and coordinate federated training in 2 steps:

(i) Based on $\mathcal{T}$, **FL Algorithm Selector Agent** (S_3) queries a registry of 40 FL algorithms containing the algorithmic descriptions and then selects a suitable method (e.g., `FedAvg`, `pFedSim`, `FedSR`) based on user requirements.
(ii) **Trainer Agent** (S_4) then distributes training details to approved clients and executes Federated Training. During training, S_4 logs per-client and global metrics (e.g., accuracy) and performs model aggregation. Its modular structure supports plug-and-play experimentation with different FL algorithms and training configurations.

2.4 Privacy Preserving and Modular Design

A key advantage of our framework is its modular design across phases and agent specializations: Each agent component and phase can be independently evaluated, replaced, or extended. More

Table 1: Comparison of LLM agents in **Dermatology** environment based on skin cancer detection task. Here **P, R, F1** indicate Precision, Recall, and F1 score of **selected clients vs. the canonical eligible client set.** **S, D, F** indicate Schema Compliance Rate, Duplicate Removal Rate, and Format Normalization Rate. **E, C, Co** indicate Exact-match Accuracy, Coverage Rate, and Conflict Rate. **T** indicates Training-start verification score.

Model	Fine-grained guidance				Goal-oriented guidance			
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Client-Sel	Data-Pre	Label-Harm	Fed-Train
	P, R, F1	S, D, F	E, C, Co	T	P, R, F1	S, D, F	E, C, Co	T
Proprietary Models								
GPT-4.1	0.96, 1.00, 0.98	1.00, 0.97, 1.00	0.61, 0.65, 0.35	0.99	0.88, 0.86, 0.87	1.00, 0.96, 0.98	0.61, 0.61, 0.39	0.85
GPT-4o	0.88, 0.89, 0.88	1.00, 0.94, 0.95	0.18, 0.27, 0.73	0.21	0.79, 0.76, 0.77	0.96, 0.91, 0.92	0.16, 0.24, 0.76	0.18
GPT-4	1.00, 0.92, 0.96	0.02, 0.01, 0.00	0.22, 0.29, 0.71	0.61	0.70, 0.68, 0.69	0.05, 0.00, 0.00	0.00, 0.01, 0.96	0.43
GPT-4-Turbo	0.91, 0.89, 0.90	0.41, 0.33, 0.39	0.19, 0.24, 0.76	0.64	0.88, 0.79, 0.83	1.00, 0.98, 0.97	0.25, 0.29, 0.71	0.45
GPT-4.1-mini	1.00, 1.00, 1.00	1.00, 0.93, 0.98	0.59, 0.65, 0.35	0.61	1.00, 0.97, 0.98	0.57, 0.53, 0.57	0.59, 0.60, 0.40	0.58
GPT-4o-mini	0.64, 0.61, 0.62	1.00, 0.92, 1.00	0.60, 0.63, 0.37	0.61	0.50, 0.56, 0.53	1.00, 0.96, 0.98	0.23, 0.26, 0.74	0.40
GPT-o4-mini	0.94, 0.91, 0.92	0.98, 0.95, 0.96	0.63, 0.71, 0.29	0.57	0.90, 0.80, 0.85	0.74, 0.70, 0.73	0.45, 0.50, 0.50	0.60
GPT-o3-mini	0.86, 0.89, 0.87	0.00, 0.00, 0.00	0.45, 0.49, 0.51	0.58	0.71, 0.77, 0.74	0.05, 0.00, 0.00	0.44, 0.50, 0.50	0.63
GPT-3.5-Turbo	0.32, 0.35, 0.33	0.04, 0.00, 0.00	0.00, 0.03, 0.97	0.18	0.41, 0.30, 0.35	0.43, 0.38, 0.38	0.00, 0.00, 1.00	0.21
Claude-3-7-Sonnet	0.67, 0.68, 0.67	0.44, 0.42, 0.42	0.21, 0.27, 0.73	0.42	0.69, 0.69, 0.69	0.40, 0.38, 0.39	0.26, 0.32, 0.68	0.44
Open-source Models								
Huge Models								
DeepSeek-V3	0.79, 0.78, 0.78	0.97, 0.96, 0.94	1.00, 1.00, 0.00	0.78	0.76, 0.75, 0.75	0.77, 0.73, 0.75	0.81, 0.83, 0.17	0.82
DeepSeek-R1	0.70, 0.65, 0.67	0.00, 0.00, 0.00	0.02, 0.08, 0.92	0.03	0.68, 0.63, 0.65	0.00, 0.00, 0.00	0.01, 0.01, 0.97	0.00
Qwen3 235B	0.62, 0.68, 0.65	0.01, 0.00, 0.00	0.02, 0.09, 0.91	0.00	0.64, 0.69, 0.66	0.08, 0.00, 0.00	0.04, 0.08, 0.92	0.01
LLaMA-4 Maverick	0.65, 0.69, 0.67	0.98, 0.90, 0.97	0.57, 0.66, 0.34	0.37	0.73, 0.64, 0.68	0.98, 0.95, 0.94	0.65, 0.68, 0.32	0.62
LLaMA-4 Scout	0.75, 0.77, 0.76	1.00, 0.93, 0.95	0.66, 0.73, 0.27	0.41	0.79, 0.80, 0.79	1.00, 0.95, 0.97	0.56, 0.64, 0.36	0.44
Large Models								
DeepSeek-R1-70B	0.71, 0.71, 0.71	0.00, 0.00, 0.00	0.02, 0.03, 0.95	0.19	0.64, 0.72, 0.68	0.00, 0.00, 0.00	0.03, 0.09, 0.91	0.00
LLaMA-3-70B	0.72, 0.65, 0.68	0.17, 0.11, 0.12	0.17, 0.20, 0.80	0.43	0.70, 0.66, 0.68	0.41, 0.39, 0.39	0.48, 0.55, 0.45	0.20
Medium Models								
Qwen QwQ 32B	0.94, 0.92, 0.93	1.00, 0.96, 1.00	0.87, 0.89, 0.11	0.84	0.86, 0.93, 0.89	1.00, 0.97, 1.00	0.57, 0.65, 0.35	0.64
Qwen3-30B	0.74, 0.68, 0.71	0.04, 0.04, 0.03	0.05, 0.06, 0.94	0.19	0.74, 0.62, 0.67	0.00, 0.00, 0.00	0.01, 0.04, 0.96	0.20
Gemma3-27B	0.30, 0.38, 0.34	0.00, 0.00, 0.00	0.00, 0.03, 0.97	0.01	0.26, 0.34, 0.29	0.00, 0.00, 0.00	0.00, 0.02, 0.95	0.04
Small Models								
Gemma-2-9B	0.69, 0.67, 0.68	0.24, 0.15, 0.19	0.19, 0.23, 0.77	0.24	0.60, 0.72, 0.65	0.24, 0.15, 0.17	0.17, 0.21, 0.79	0.19
LLaMA-3-8B	0.72, 0.65, 0.68	1.00, 0.92, 0.98	0.38, 0.44, 0.56	0.20	0.71, 0.61, 0.66	0.98, 0.95, 0.97	0.45, 0.51, 0.49	0.19
Qwen-3-14B	0.70, 0.69, 0.69	0.04, 0.00, 0.04	0.06, 0.11, 0.89	0.02	0.59, 0.65, 0.62	0.00, 0.00, 0.00	0.03, 0.07, 0.93	0.04
Gemma3-12B-instruct	0.38, 0.36, 0.37	0.00, 0.00, 0.00	0.00, 0.05, 0.95	0.05	0.34, 0.37, 0.35	0.00, 0.00, 0.00	0.06, 0.08, 0.92	0.04

importantly, this modularity enables future expansion of the benchmark and adaptation to diverse real-world scenarios. For instance, additional components simulating privacy/safety audits conducted by humans or AI can be seamlessly inserted between server and client agents or workflow phases, without the need for altering the existing workflow.

It is to be noted that our framework enforces data privacy by design, aligning fully with FL principles. We explicitly prevent agents from ever accessing or transmitting raw data, model weights, or sensitive metadata. The server receives approvals/configuration signals only, not images, so the agent layer never handles patient data. Instead, agents operate at orchestration layer only and exchange only predefined information (JSON configs, file paths, status signals). They do not have direct access to raw client data (e.g., patient images) or sensitive metadata and never transmit patient data or intermediate outputs externally. Training is invoked via a tool wrapper that runs locally per client; no raw data leaves clients at the agent layer, *i.e.*, federated training is triggered by the agent, but executed on local clients via tools. All data preprocessing and label harmonization also happen locally at clients. Eg: In label harmonization, the agent creates mapping logic, but the mapping execution and label replacement are performed entirely on the local client side.

3 Experiments and Results

3.1 Implementation and Evaluation Details

We utilize the LangGraph architecture (Langgraph, 2025) for agent construction and workflow graph compilation. Each agent is assigned a tailored toolset, drawn from our proposed suite of 16 tools **(see Appendix B.1)**, with the selection guided by the agent's specific role and the need to omit redundant or irrelevant functionalities. In order to assess the capabilities of existing LLM agents, we validate a total number of 24 models on the FedAgentBench datasets, including: (1) 10 representative proprietary LLMs: GPT 4.1, GPT-4o, GPT-4, GPT-4-Turbo, GPT 4.1-mini, GPT-4o-mini, GPT o4-mini, GPT o3-mini, GPT-3.5 Turbo, and Claude-3.7 Sonnet. (2) 14 state-of-the-art open-sourced LLMs ranging from 9B to 685B: LLaMA series models (LLaMA-4 Maverick, LLaMA-4 Scout, LLaMA-3 70B, LLaMA-3 8B), DeepSeek series models (DeepSeek-V3, Deepseek-R1, DeepSeek-R1-Distill-Llama-70B), Qwen series models (Qwen 3 235B, Qwen QwQ 32B, Qwen 3 30B, Qwen 3 14B) and Gemma series models (Gemma 3 27B Instruct, Gemma 3 12B Instruct, Gemma 2 9B Instruct). We utilize APIs from (OpenAI, 2025), (Groq, 2025), (Deep Infra, 2025).

Evaluation metrics: We evaluate the agentic performance using a total of **13 key metrics** in different steps of the FL workflow: **(1) For each step**, we use **Success Rate over 5 runs** which is a binary indicator of task success/completion. It evaluates the ability of the multi-agent framework to generate executable outputs that satisfy the task requirements. **(2) For client selection step,** we use **Precision, Recall, and F1 score** of selected clients vs. the canonical eligible client set (and not of model performance). **(3) For data pre-processing step**, we use **(i) Schema Compliance Rate**, *i.e.*, proportion of correctly structured folders/files, **(ii) Duplicate Removal Rate**, *i.e.*, proportion of duplicates removed, and **(iii) Format Normalization Rate**, *i.e.*, proportion of files correctly normalized (e.g., format, resolution). **(4) For label harmonization step**, we use: **(i) Exact-match Accuracy** of label mappings vs. the canonical schema, **(ii) Coverage Rate**, *i.e.*, proportion of local classes successfully mapped, **(iii) Conflict Rate**, *i.e.*, proportion of classes with ambiguous mappings. **(5) For federated training step**, we use **Training Start Verification** as the metric to determine whether the agent produces valid configuration files, initializes the training process, and logs the start signal. Besides, **for each step**, we also compute **(6) Time Spent in seconds** which denotes the duration required to complete the task **(see Appendix D & Table 16 for comparison of average time)**; and **(7) Token Requirement** which indicates the number of tokens involved **(see Fig. 1 (d) for comparison of token requirement)**.

Tasks: The benchmark is tested on six representative real-world clinical tasks across six major medical imaging modalities: (i) Skin cancer detection from dermatology images (Tables 1 and 10), (ii) Breast cancer detection from ultrasound (Table 11), (iii) Glaucoma detection from fundus imaging (Table 14), (iv) Pneumonia detection from chest X-ray (Table 15), (v) Brain tumor detection from MRI (Table 13), and (vi) Lymph-node metastasis detection from histopathology (Table 2).

3.2 Main Results and Key Insights

We summarize the overall success scores of all agent cores over 6 modality specific environments with two types of guidance styles for prompting LLMs *viz.*, fine-grained guidance (explicit step-by-step instructions) and goal-oriented guidance (high-level task description) in Fig. 7. We also show detailed performance breakdown of Dermatology environment in Table 1 and Histopathology in Table 2. **For detailed results in all other environments, please see Appendix D. & Tables 10-15** Also, see Fig 1 (d) for overall token requirements of each model.

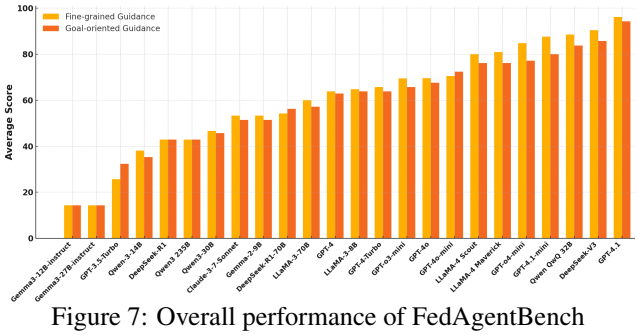


Figure 7: Overall performance of FedAgentBench

From the tables, we find proprietary models consistently outperform open-source ones across all FL stages. Besides, fine-grained guidance yields higher success rates than goal-oriented prompts for most models. Performance drops in more complex tasks like label harmonization compared to client selection. We also observe that model size alone does not guarantee performance (see Fig. 7). Instead, architectural design and instruction-following capability are more critical.

RQ1: Impact of Task Complexity: High success is observed in the initial and final steps of client orchestration and federated training across almost all agents, including weaker ones indicating that these tasks are relatively simpler. Data Pre-processing and Label Harmonization are seen to be major differentiators among agents. Weaker agents particularly fail to perform these tasks especially in goal-oriented scenarios, where planning and file structure comprehension are needed. Across almost all agents, label harmonization shows lowest success rates, regardless of guidance type. This suggests that aligning semantic labels across clients remains one of the hardest challenges. Among modalities, histopathology has the highest semantic complexity, potentially due to domain-specific terminology.

RQ2: Granularity of guidance: In fine-grained guidance, we provide explicit instruction to the models to follow a particular workflow whereas in goal-oriented guidance, we mention the overall objective of the agent without specifying the exact steps, thereby requiring autonomous planning or reasoning. Fine-grained guidance is seen to outperform goal-oriented guidance across almost every model, especially for weaker agents. More capable models like GPT-4.1 and DeepSeek-V3 close this gap, showing their capability to plan even based on implicit prompts.

Table 2: Comparison in terms of success rate over 5 runs for **Lymph-node metastasis detection** task in **Histopathology** environment

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
	S_1, C_1, S_2	C_2	C_3	S_3, S_4		S_1, C_1, S_2	C_2	C_3	S_3, S_4	
Proprietary Models										
GPT-4.1	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29
GPT-4o	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71	5/5, 0/5, 5/5	5/5	1/5	1/5, 5/5	62.86
GPT-4	5/5, 1/5, 5/5	0/5	1/5	2/5, 5/5	54.29	5/5, 1/5, 5/5	0/5	0/5	2/5, 5/5	51.43
GPT-4-Turbo	5/5, 1/5, 5/5	1/5	1/5	2/5, 5/5	57.14	5/5, 1/5, 5/5	4/5	1/5	2/5, 5/5	65.71
GPT-4.1-mini	5/5, 3/5, 5/5	5/5	4/5	3/5, 5/5	85.71	5/5, 3/5, 5/5	3/5	4/5	3/5, 5/5	80.00
GPT-4o-mini	5/5, 1/5, 3/5	5/5	3/5	2/5, 4/5	65.71	5/5, 1/5, 3/5	5/5	1/5	2/5, 4/5	60.00
GPT-o4-mini	5/5, 2/5, 5/5	5/5	3/5	2/5, 5/5	77.14	5/5, 2/5, 5/5	4/5	2/5	2/5, 4/5	68.57
GPT-o3-mini	5/5, 5/5, 5/5	0/5	2/5	3/5, 5/5	71.43	5/5, 4/5, 5/5	0/5	2/5	3/5, 5/5	68.57
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 3/5	31.43
Claude-3-7-Sonnet	5/5, 2/5, 3/5	2/5	1/5	2/5, 3/5	51.43	5/5, 2/5, 3/5	2/5	1/5	2/5, 5/5	57.14
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 3/5, 5/5	5/5	5/5	4/5, 5/5	91.43	5/5, 3/5, 5/5	4/5	5/5	4/5, 5/5	88.57
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 2/5, 4/5	5/5	3/5	3/5, 5/5	77.14	5/5, 2/5, 4/5	5/5	3/5	3/5, 5/5	71.43
LLaMA-4 Scout	5/5, 2/5, 5/5	5/5	4/5	2/5, 5/5	80.00	5/5, 2/5, 5/5	5/5	3/5	2/5, 5/5	77.14
Large Models										
DeepSeek-R1-70B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-3-70B	5/5, 1/5, 5/5	1/5	1/5	1/5, 5/5	54.29	5/5, 1/5, 5/5	2/5	2/5	1/5, 5/5	60.00
Medium Models										
Qwen QwQ 32B	5/5, 4/5, 5/5	3/5	4/5	4/5, 5/5	85.71	5/5, 4/5, 5/5	2/5	4/5	4/5, 5/5	82.86
Qwen3-30B	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 1/5, 5/5	2/5	1/5	1/5, 5/5	57.14	5/5, 1/5, 5/5	1/5	1/5	1/5, 5/5	54.29
LLaMA-3-8B	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71
Qwen-3-14B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 4/5	40.00
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

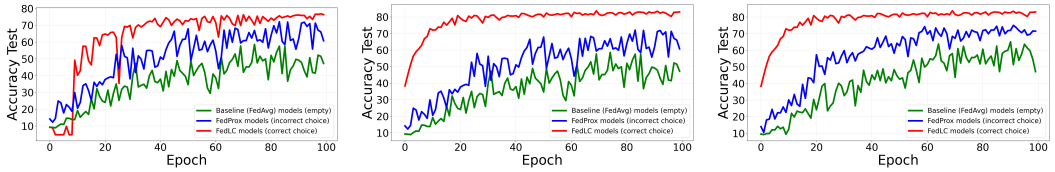
RQ3 & RQ5: Open Source Vs. Proprietary Models and Impact of Model Size:

Proprietary Model Performance: GPT-4.1 and GPT-4.1-mini show top-tier performance (85–100%), especially under fine-grained guidance. GPT-4o, although newer, struggles with label harmonization and federated training across all environments, leading to overall lower scores (62-71%). Claude-3.7-Sonnet achieves moderate performance (51–57%), inferior to GPT-4 variants. GPT-3.5-Turbo and older variants perform poorly, barely completing the complex stages.

Open-source Model Performance: We discuss agent performance based on model sizes below:
(i) Huge Models: DeepSeek-V3 is the strongest open-source model contender with 80–94% success rate comparable to the best proprietary models. Qwen3 and DeepSeek-R1 perform inconsistently, often failing in more structured stages like data pre-processing and label harmonization.
(ii) Medium and Large Models: Qwen QwQ 32B demonstrates strong performance (82–91%) and outperforms several proprietary models even under goal-oriented setups. LLaMA-4 Scout and Maverick also deliver competitive performance, especially in label harmonization and federated training, with scores in the 71–94% range. Other large models such as LLaMA-3-70B, and Qwen3-30B struggle with most tasks except initial client communication or final training step. Gemma3-27B-instruct is unusable under almost all these settings.
(iii) Small Models: Performance of 8-14B sized-models drops significantly. Most models (except LLaMA 3 8B) achieve less than or around 50% success. Particularly, Gemma 3-12B-instruct and Qwen 3 14B are observed to fail due to extreme hallucinations. These models are unable to perform any label-oriented reasoning and structured data operations, even under fine-grained instructions.

3.3 RQ4: Agent Failure Analysis:

We identify six key recurring failure modes of LLM agents across FL sub-tasks that highlight important limitations of current LLM capabilities in FL workflows **(see Appendix D for more details)**:
(i) Lack of Domain-Specific Reasoning: The agents frequently fail to apply relevant medical domain knowledge. **Eg:** In label harmonization (Fig 6), the agents often miss subtle mismatches between dermatology folder names and coarse class labels possibly due to the lack of domain grounding and inability to handle naming conventions specific to medical datasets.
(ii) Failure in Multi-Step Planning: The agents are often unable to follow multi-step workflows, skipping essential operations where multiple sequential actions are required. **Eg:** Data pre-processor agents often overlook file/data cleaning steps of Fig. 5 due to multiple tasks in single execution cycle.
(iii) Overconfidence and Shortcutting: The agents recurrently provide wrong solutions, by defaulting to plausible but incorrect logic when unsure, instead of expressing uncertainty. **Eg:** Assigning both "nevus" and "melanoma metastasis" to the 'benign' class to simplify label mapping.
(iv) Hallucination in Structured Multi-Agent Tasks: The agents (particularly DeepSeek R1 and Gemma-based models) often generate irrelevant or unrelated outputs despite specific instructions



(a) No Data-Pre (b) FedLC models with Data-Pre, Others w/o Data-Pre (c) All with Data-Pre

Figure 8: Ablation analysis in Dermatology environment for instruction: *Select FL algorithm that mitigates inter- and intra-client class imbalance.* Agents highlighted in red choose the correct method (FedLC), while those in blue select FedProx and others coded in green return no algorithm (defaulting to FedAvg). Subplots illustrate: (a) reduced performance when the data–preprocessing step fails, affecting all agentic systems; (b) Improvement for agents in red that correctly preprocess; and (c) full performance gains when all agents successfully complete preprocessing.

due to misalignment with structured task formats and poor control over output scope **(see Fig. 18-19 in Appendix D)**. **Eg:** When asked to select skin cancer dataset, Gemma-3 27B Instruct repeatedly returned philosophical or sarcastic monologues in foreign languages, tutorials on freelancing, etc.
(v) Task-Type and Modality Mismatch Due to Prior Assumptions: Agents can sometimes confuse tasks or ignore modality constraints due to frequency biases and shallow keyword matching instead of hierarchical task understanding. **Eg:** Recommending a malignant lesion segmentation dataset for a classification task or ultrasound datasets for histopathology-based breast cancer detection task.
(vi) Procedural Overthinking and Paralysis by Analysis: The reasoning/thinking agents often delay execution by speculating about dataset structure or missing dependencies without being asked, potentially due to excessive internal reasoning without grounding in file system or available information **(see Fig. 16 in Appendix D)**. **Eg:** DeepSeek R1 repeatedly debates whether a client dataset should be selected without reading the dataset description file.

3.4 Final Federated Training Performance:

To test whether agents truly select algorithms that improve overall FL performance, not just pass the "training-start" check, we run full end-to-end FL experiments. For the instruction: *"Train a federated learning model using an algorithm designed to mitigate both inter-client and intra-client class imbalance while still producing a strong global model"*, models like GPT-4.1, GPT-4o, GPT-4, Claude-3-Sonnet, DeepSeek-V3, Qwen QwQ 32B, Gemma-2-9B correctly select FedLC, while GPT-3.5-Turbo, Qwen3-235B, LLaMA-4 Maverick, LLaMA-4 Scout, and others wrongly choose FedProx. Some models *viz.*, DeepSeek-R1, DeepSeek-R1-70B, Qwen3-30B, Qwen-3-14B return no algorithm and thus fall back to FedAvg. Across all runs, the performance ranks consistently as FedLC > FedProx > FedAvg, confirming that FedAgentBench captures real downstream impact rather than superficial setup success. See Fig. 8, Appendix Tables 18-20 for accuracy curves and detailed results.

4 Conclusion and Limitation

In this paper, we introduced **the first agent-driven FL framework** and an associated benchmark, **FedAgentBench**, for evaluating LLM agents across diverse tasks constituting typical FL workflows. The evaluation covers 24 LLMs with varying sizes and a wide range of FL sub-tasks with varying difficulty levels in six modality-specific FL settings that closely simulate real-world clinical FL environments. Our framework is privacy preserving, comprehensive and modular. It includes 201 medical datasets and 40 FL algorithms and can be easily extended to incorporate more functionalities, settings, and algorithms specific to the user requirement. We investigated the impact of various factors like FL task complexity and granularity of guidance on the agent performance and analyzed the common failure modes of different agents. Our experiments reveal that across all environments, GPT-4.1 achieves almost perfect scores, under both fine-grained and goal-oriented prompting, whereas GPT-3.5-Turbo, Gemma3 series, and some Qwen variants consistently underperform across all stages and environments. DeepSeek-V3, Qwen QwQ 32B, and LLaMA-4 Maverick are the most reliable open-source agents across tasks. Unsurprisingly, fine-grained guidance consistently outperforms goal-oriented prompting, especially for less capable models. Our findings highlight that the order of complexity of the FL sub-tasks for most agents is: Label Harmonization > Data Pre-processing > Federated Training > Client Orchestration. Our experiments also show that larger model size does not necessarily correlate with better performance, *i.e.*, some mid-sized models (30–40B) outperform larger ones (70B+). E.g., Qwen QwQ 32B consistently outperforms Qwen3-235B and DeepSeek-R1-70B.

REFERENCES

3d-mri-ultrasound-brain dataset. https://www.kaggle.com/datasets/shubhamcodez/3d-mri-ultrasound-brain-images. Accessed: 2025-05-22.

Buet dataset. https://www.kaggle.com/datasets/jarintasnim090/buet-breast-ultrasound-data. Accessed: 2025-05-22.

Bus-uclm dataset. https://www.kaggle.com/datasets/orvile/bus-uclm-breast-ultrasound-dataset, a. Accessed: 2025-05-22.

Busi dataset. https://www.kaggle.com/datasets/sabahesaraki/breast-ultrasound-images-dataset, b. Accessed: 2025-05-22.

Breast-cancer dataset. https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset, a. Accessed: 2025-05-22.

Breast-ultrasound-images dataset. https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset, note = Accessed: 2025-05-22, b.

Camus-human-heart-data. https://www.kaggle.com/datasets/shoybhasan/camus-human-heart-data. Accessed: 2025-05-22.

Ct2us dataset. https://www.kaggle.com/datasets/siatsyx/ct2usforkidneyseg. Accessed: 2025-05-22.

Carotid-ultrasound-image dataset. https://www.kaggle.com/datasets/orvile/carotid-ultrasound-images, a. Accessed: 2025-05-22.

Carotid-artery dataset. https://www.kaggle.com/datasets/pahunichoudhary/carotid-artery-ultrasound-and-color-doppler, b. Accessed: 2025-05-22.

Ddti dataset. https://www.kaggle.com/datasets/dasmehdixtr/ddti-thyroid-ultrasound-images. Accessed: 2025-05-22.

Dr dataset. https://www.kaggle.com/c/diabetic-retinopathy-detection/data. Accessed: 2025-05-22.

Dermatology-tabular-dataset. https://www.kaggle.com/datasets/olcaybolat1/dermatology-dataset-classification, a. Accessed: 2025-05-22.

Dermis dataset. https://www.kaggle.com/datasets/farhatullah8398/skin-lesion-dermis-dataset, b. Accessed: 2025-05-22.

Dermnet dataset. https://www.kaggle.com/datasets/shubhamgoel27/dermnet, c. Accessed: 2025-05-22.

Dermquest dataset. http://www.dermquest.com, d. Accessed: 2025-05-22.

Fetal-head-us dataset. https://www.kaggle.com/datasets/ankit8467/fetal-head-ultrasound-dataset-for-image-segment, a. Accessed: 2025-05-22.

Fetal-health-classification dataset. https://www.kaggle.com/datasets/andrewmvd/fetal-health-classification, b. Accessed: 2025-05-22.

Isic2020 dataset. https://kaggle.com/competitions/siim-isic-melanoma-classification, a. Accessed: 2025-05-22.

Isic2024 dataset. https://kaggle.com/competitions/isic-2024-challenge, b. Accessed: 2025-05-22.

Liver-histopathology dataset. http://kaggle.com/datasets/vibhingupta028/liver-histopathology-fibrosis-ultrasound-images. Accessed: 2025-05-22.

Mednode dataset. https://www.kaggle.com/datasets/prabhavsanga/med-node. Accessed: 2025-05-22.

Monkeypox-skin-image-dataset. https://www.kaggle.com/datasets/nafin59/monkeypox-skin-lesion-dataset. Accessed: 2025-05-22.

Pcos dataset. https://www.kaggle.com/datasets/anaghachoudhari/pcos-detection-using-ultrasound-images. Accessed: 2025-05-22.

Ph2dataset. https://www.kaggle.com/datasets/spacesurfer/ph2-dataset. Accessed: 2025-05-22.

Rsna-asnr dataset. https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classifica Accessed: 2025-05-22.

Regensburg pediatric appendicitis dataset. https://www.kaggle.com/datasets/joebeachcapital/regensburg-pediatric-appendicitis. Accessed: 2025-05-22.

Robotic$_h$$andheld_l$$umbar_s$$pine_u$$sdataset$.. Accessed: 2025-05-22.

Stare dataset. https://www.kaggle.com/datasets/vidheeshnacode/stare-dataset. Accessed: 2025-05-22.

Thyroid$_u$$ltrasounddataset$.. Accessed: 2025-05-22.

Us3m dataset. https://www.kaggle.com/datasets/timesxy/multimodal-breast-ultrasound-dataset-us3m. Accessed: 2025-05-22.

abdomen-mri dataset. https://www.kaggle.com/datasets/imtkaggleteam/abdomen-mri-object-detection. Accessed: 2025-05-22.

alzheimer-mri dataset. ://www.kaggle.com/datasets/borhanitrash/alzheimer-mri-disease-classification-dataset.

Augmented skin conditions image dataset, howpublished = https://www.kaggle.com/datasets/syedalinaqvi/augmented-skin-conditions-image-dataset, note = Accessed: 2025-05-22.

brain-cancer dataset. https://www.kaggle.com/datasets/orvile/brain-cancer-mri-dataset, a. Accessed: 2025-05-22.

brain-mri-images dataset. https://www.kaggle.com/datasets/ashfakyeafi/brain-mri-images, b. Accessed: 2025-05-22.

brain-tumor dataset. https://www.kaggle.com/datasets/ultralytics/brain-tumor, c. Accessed: 2025-05-22.

brain-tumor-classification dataset. https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri, d. Accessed: 2025-05-22.

brain-tumor-detection dataset. https://www.kaggle.com/datasets/abhranta/brain-tumor-detection-mri, e. Accessed: 2025-05-22.

chasedb1 dataset. https://blogs.kingston.ac.uk/retinal/chasedb1/.

dermatologic-ultrasound dataset. https://www.kaggle.com/datasets/alfageme/dermatologic-ultrasound-images. Accessed: 2025-05-22.

facial-mr dataset. https://www.kaggle.com/datasets/humanaizedata/facial-mri-dataset-boost-your-ai-models. Accessed: 2025-05-22.

fallmud dataset. https://www.kaggle.com/datasets/angeliqueloesch/fallmud. Accessed: 2025-05-22.

fetalultrasoundbrain dataset. https://www.kaggle.com/datasets/rahimalargo/fetalultrasoundbrain. Accessed: 2025-05-22.

fhms-ultrasound dataset. https://www.kaggle.com/datasets/jai132/fhms-ultrasound-dataset. Accessed: 2025-05-22.

foraminal-stenosis dataset. https://www.kaggle.com/datasets/axondata/foraminal-stenosis-mri-dataset. Accessed: 2025-05-22.

glioma dataset. https://www.kaggle.com/datasets/azharsaleem/mri-based-glioma-detection-dataset-with-masks. Accessed: 2025-05-22.

hbfmid dataset. https://www.kaggle.com/datasets/orvile/human-bone-fractures-image-dataset-hbfmid. Accessed: 2025-05-22.

heart-mri dataset. https://www.kaggle.com/datasets/adarshsng/heart-mri-image-dataset-left-atrial-segmentation. Accessed: 2025-05-22.

hippocampal dataset. https://www.kaggle.com/datasets/aryashah2k/hippocampal-sparing-dataset. Accessed: 2025-05-22.

indian-diabetic-retinopathy dataset. https://www.kaggle.com/datasets/aaryapatel98/indian-diabetic-retinopathy-image-dataset. Accessed: 2025-05-22.

isles dataset. https://www.kaggle.com/datasets/orvile/isles-2022-brain-stoke-dataset. Accessed: 2025-05-22.

Fetal$_h ealth_c lassificationdataset.$. Accessed: 2025-05-22.

mendeley$_b reast_u sdataset.$. Accessed: 2025-05-22.

mri-and-pet dataset. https://www.kaggle.com/datasets/grantmcnatt/mri-and-pet-dice-similarity-dataset. Accessed: 2025-05-22.

multimodal-ultrasound-vascular-segmentation dataset. https://www.kaggle.com/datasets/among22/multimodal-ultrasound-vascular-segmentation. Accessed: 2025-05-22.

oasis-1 dataset. https://www.kaggle.com/datasets/mdfahimbinamin/oasis-1-fastsurfer-quickseg-segmentation-dataset. Accessed: 2025-05-22.

phantom dataset. https://www.kaggle.com/datasets/ukeppendorf/frequently-traveling-human-phantom-fthp-dataset. Accessed: 2025-05-22.

pmram dataset. https://www.kaggle.com/datasets/orvile/pmram-bangladeshi-brain-cancer-mri-dataset. Accessed: 2025-05-22.

prostate-annotated dataset. https://www.kaggle.com/datasets/haithem1999/prostate-annotated-dataset-for-image-segmentation, a. Accessed: 2025-05-22.

prostate-mri-us dataset. https://www.kaggle.com/datasets/dsptlp/prostate-mri-us-biopsy, b. Accessed: 2025-05-22.

refuge2 dataset. https://www.kaggle.com/datasets/victorlemosml/refuge2. Accessed: 2025-05-22.

skin$_d isease_c ls_k aggledataset.$, a. Accessed: 2025-05-22.

skin$_i nfectiondataset.$, b. Accessed: 2025-05-22.

spinal-cord-dataset. https://www.kaggle.com/datasets/trainingdatapro/spinal-cord-dataset. Accessed: 2025-05-22.

stroke$_h ead_m ridataset..$ Accessed: 2025-05-22.

ultra-lr-hr-ultrasound dataset. https://www.kaggle.com/datasets/chirag2466/ultra-lr-hr-ultrasound-image-dataset-for-research, a. Accessed: 2025-05-22.

ultrasound-nerve-segmentation dataset. https://www.kaggle.com/competitions/ultrasound-nerve-segmentation, b. Accessed: 2025-05-22.

wmh-dataset. https://www.kaggle.com/datasets/farahmo/wmh-dataset. Accessed: 2025-05-22.

Mohammad Amin Abbasi, Farnaz Sadat Mirnezami, and Hassan Naderi. Hamraz: A culture-based persian conversation dataset for person-centered therapy using llm agents. *arXiv preprint arXiv:2502.05982*, 2025.

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization, 2021. URL https://arxiv.org/abs/2111.04263.

Vaibhav Aggarwal, Ojasv Kamal, Abhinav Japesh, Zhijing Jin, and Bernhard Schölkopf. Dars: Dynamic action re-sampling to enhance coding agent performance by adaptive tree traversal. *arXiv preprint arXiv:2503.14269*, 2025.

Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy. Dataset of breast ultrasound images. *Data in Brief*, 28:1–5, 2020.

Alibabaei78. Ebhi segmentation dataset. https://www.kaggle.com/datasets/alibabaei78/ebhi-seg, 2024. Accessed: 2024-05-22.

Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. Federated learning for healthcare: Systematic review and architecture proposal. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–23, 2022.

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers, 2019. URL https://arxiv.org/abs/1912.00818.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models, August 2021. URL http://arxiv.org/abs/2108.07732. arXiv:2108.07732 [cs].

Reza Averly, Frazier N Baker, and Xia Ning. Liddia: Language-based intelligent drug discovery agent. *arXiv preprint arXiv:2502.13959*, 2025.

Beosup. Lung segment dataset. https://www.kaggle.com/datasets/beosup/lung-segment, 2023. Accessed: 2025-05-22.

Olivier Bernard et al. Deep learning techniques for automatic MRI cardiac multi-structures segmentation and diagnosis: is the problem solved? *IEEE Transactions on Medical Imaging*, 37(11): 2514–2525, 2018.

Matt Berseth. Isic 2017-skin lesion analysis towards melanoma detection. *arXiv preprint arXiv:1703.00523*, 2017.

Michał Byra, Grzegorz Styczynski, Cezary Szmigielski, Piotr Kalinowski, Łukasz Michałowski, Rafał Paluszkiewicz, Bogna Ziarkiewicz-Wróblewska, Krzysztof Zieniewicz, Piotr Sobieraj, and Andrzej Nowicki. Transfer learning with deep convolutional neural network for liver steatosis assessment in ultrasound images. *International journal of computer assisted radiology and surgery*, 13(12):1895–1903, 2018.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. In *The Twelfth International Conference on Learning Representations*.

John Capocyan. Cellnet beta version. https://www.kaggle.com/datasets/johncapocyan/cellnet-beta-version, 2024. Accessed: 2024-05-22.

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. Unleashing the potential of prompt engineering for large language models. *Patterns*, 2025a.

Dengsheng Chen, Jie Hu, Vince Junkai Tan, Xiaoming Wei, and Enhua Wu. Elastic aggregation for federated optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12187–12197, June 2023a.

Hao-Yuan Chen, Cheng-Pong Huang, and Jui-Ming Yao. Verbal process supervision elicits better coding agents. *arXiv preprint arXiv:2503.18494*, 2025b.

Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification, 2022. URL https://arxiv.org/abs/2107.00778.

Junying Chen, Chi Gui, Anningzhe Gao, Ke Ji, Xidong Wang, Xiang Wan, and Benyou Wang. Cod, towards an interpretable medical agent using chain of diagnosis. *arXiv preprint arXiv:2407.13301*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code, July 2021. URL http://arxiv.org/abs/2107.03374. arXiv:2107.03374 [cs].

Yiqiang Chen, Wang Lu, Xin Qin, Jindong Wang, and Xing Xie. Metafed: Federated learning among federations with cyclic knowledge distillation for personalized healthcare, 2023b. URL https://arxiv.org/abs/2206.08516.

Zhaoling Chen, Xiangru Tang, Gangda Deng, Fang Wu, Jialong Wu, Zhiwei Jiang, Viktor Prasanna, Arman Cohan, and Xingyao Wang. Locagent: Graph-guided llm agents for code localization. *arXiv preprint arXiv:2503.09089*, 2025c.

Zhen Chen, Zhihao Peng, Xusheng Liang, Cheng Wang, Peigan Liang, Linsheng Zeng, Minjie Ju, and Yixuan Yuan. Map: Evaluation and multi-agent enhancement of large language models for inpatient pathways. *arXiv preprint arXiv:2503.13205*, 2025d.

Hojun Cho, Donghu Kim, Soyoung Yang, Chan Lee, Hunjoo Lee, and Jaegul Choo. Building resource-constrained language agents: A korean case study on chemical toxicity information. *arXiv preprint arXiv:2503.17753*, 2025.

Zhendong Chu, Shen Wang, Jian Xie, Tinghui Zhu, Yibo Yan, Jinheng Ye, Aoxiao Zhong, Xuming Hu, Jing Liang, Philip S Yu, et al. Llm agents for education: Advances and applications. *arXiv preprint arXiv:2503.11733*, 2025.

Noel Codella, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368*, 2019.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning, 2023. URL https://arxiv.org/abs/2102.07078.

Marc Combalia, Noel C. F. Codella, Veronica Rotemberg, Brian Helba, Veronica Vilaplana, Ofer Reiter, Cristina Carrera, Alicia Barreiro, Allan C. Halpern, Susana Puig, and Josep Malvehy. Bcn20000: Dermoscopic lesions in the wild, 2019. URL https://arxiv.org/abs/1908.02288.

Roxana Daneshjou, Kailas Vodrahalli, Roberto A Novoa, Melissa Jenkins, Weixin Liang, Veronica Rotemberg, Justin Ko, Susan M Swetter, Elizabeth E Bailey, Olivier Gevaert, et al. Disparities in dermatology ai performance on a diverse, curated clinical image set. *Science advances*, 8(31): eabq6147, 2022.

P. K. Darabi. Bone break classification image dataset. https://www.kaggle.com/datasets/pkdarabi/bone-break-classification-image-dataset, 2023. Accessed: 2025-05-22.

Sergio MM de Faria, Jose N Filipe, Pedro MM Pereira, Luis MN Tavora, Pedro AA Assuncao, Miguel O Santos, Rui Fonseca-Pinto, Felicidade Santiago, Victoria Dominguez, and Martinha Henrique. Light field image dataset of skin lesions. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3905–3908. IEEE, 2019.

Coen De Vente, Koenraad A Vermeer, Nicolas Jaccard, He Wang, Hongyi Sun, Firas Khader, Daniel Truhn, Temirgali Aimyshev, Yerkebulan Zhanibekuly, Tien-Dung Le, et al. Airogs: Artificial intelligence for robust glaucoma screening challenge. *IEEE transactions on medical imaging*, 43(1): 542–557, 2023.

Etienne Decenciere, Guy Cazuguel, Xiwei Zhang, Guillaume Thibault, J-C Klein, Fernand Meyer, Beatriz Marcotegui, Gwénolé Quellec, Mathieu Lamard, Ronan Danno, et al. Teleophta: Machine learning and image processing methods for teleophthalmology. *Irbm*, 34(2):196–203, 2013.

Deep Infra. Deepinfra models documentation, 2025. URL https://deepinfra.com/docs/models. Accessed: 2025-05-16.

Qixin Deng, Qikai Yang, Ruibin Yuan, Yipeng Huang, Yi Wang, Xubo Liu, Zeyue Tian, Jiahao Pan, Ge Zhang, Hanfeng Lin, et al. Composerx: Multi-agent symbolic music composition with llms. *arXiv preprint arXiv:2404.18081*, 2024.

Yihe Deng and Paul Mineiro. Flow-dpo: Improving llm mathematical reasoning through online multi-agent learning. *arXiv preprint arXiv:2410.22304*, 2024.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning, 2020. URL https://arxiv.org/abs/2003.13461.

Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes, 2022. URL https://arxiv.org/abs/2006.08848.

Tuan Le Dinh. Monuseg 2018. https://www.kaggle.com/datasets/tuanledinh/monuseg2018, 2024. Accessed: 2024-05-22.

Wei Dong. The ann arbor architecture for agent-oriented programming. *arXiv preprint arXiv:2502.09903*, 2025.

Zhuoyun Du, Lujie Zheng, Renjun Hu, Yuyang Xu, Xiawei Li, Ying Sun, Wei Chen, Jian Wu, Haolei Cai, and Haohao Ying. Llms can simulate standardized patients via agent coevolution. *arXiv preprint arXiv:2412.11716*, 2024.

Gaurav Dutta. Fracturefusion: A symphony of bone breaks. https://www.kaggle.com/datasets/gauravduttakiit/fracturefusion-a-symphony-of-bone-breaks, 2023. Accessed: 2025-05-22.

Sachi Dwivedi. Kmc kidney histopathology dataset. https://www.kaggle.com/datasets/sachidwivedi1234/kmc-kidney-histopathology-dataset2, 2024. Accessed: 2024-05-22.

factory.ai. Code Droid Technical Report, June 2024. URL https://www.factory.ai/news/code-droid-technical-report.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3557–3568. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/24389bfe4fe2eba8bf9aa9203a44cdad-Paper.pdf.

Adibvafa Fallahpour, Jun Ma, Alif Munim, Hongwei Lyu, and BO WANG. Medrax: Medical reasoning agent for chest x-ray. In *Forty-second International Conference on Machine Learning*.

Sorouralsadat Fatemi and Yuheng Hu. Enhancing financial question answering with a multi-agent reflection framework. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 530–537, 2024.

George Fatouros, Kostas Metaxas, John Soldatos, and Manos Karathanassis. Marketsenseai 2.0: Enhancing stock analysis through llm agents. *arXiv preprint arXiv:2502.00415*, 2025.

Jared Feng. Histopath-sn. https://www.kaggle.com/datasets/jaredfeng/histopathsn, 2024. Accessed: 2024-05-22.

Jinghao Feng, Qiaoyu Zheng, Chaoyi Wu, Ziheng Zhao, Ya Zhang, Yanfeng Wang, and Weidi Xie. Mˆ 3builder: A multi-agent system for automated machine learning in medical imaging. *arXiv preprint arXiv:2502.20301*, 2025.

Forderation. Breakhis 400x. https://www.kaggle.com/datasets/forderation/breakhis-400x, 2024. Accessed: 2024-05-22.

Francisco Fumero, Silvia Alayón, José L Sanchez, Jose Sigut, and M Gonzalez-Hernandez. Rim-one: An open retinal image database for optic nerve evaluation. In *24th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 1–6. IEEE, 2011.

Fatemeh Ghezloo, Mehmet Saygin Seyfioglu, Rustin Soraki, Wisdom O Ikezogwo, Beibin Li, Tejoram Vivekanandan, Joann G Elmore, Ranjay Krishna, and Linda Shapiro. Pathfinder: A multi-modal multi-agent system for medical diagnostic decision-making applied to histopathology. *arXiv preprint arXiv:2502.08916*, 2025.

Arsham Gholamzadeh Khoee, Shuai Wang, Yinan Yu, Robert Feldt, and Dhasarathy Parthasarathy. Gatelens: A reasoning-enhanced llm agent for automotive software release analytics. *arXiv e-prints*, pp. arXiv–2503, 2025.

Mohamed Gobara. Multi-class knee osteoporosis x-ray dataset. https://www.kaggle.com/datasets/mohamedgobara/multi-class-knee-osteoporosis-x-ray-dataset, 2023a. Accessed: 2025-05-22.

Mohamed Gobara. Osteoporosis database. https://www.kaggle.com/datasets/mohamedgobara/osteoporosis-database, 2023b. Accessed: 2025-05-22.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, et al. Towards an ai co-scientist. *arXiv preprint arXiv:2502.18864*, 2025.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yujiu Yang, Minlie Huang, Nan Duan, Weizhu Chen, et al. Tora: A tool-integrated reasoning agent for mathematical problem solving. In *The Twelfth International Conference on Learning Representations*.

Dennis Grinwald, Philipp Wiesner, and Shinichi Nakajima. Federated learning over connected modes, 2025. URL https://arxiv.org/abs/2403.03333.

Fabian Gröger, Simone Lionetti, Philippe Gottfrois, Alvaro Gonzalez-Jimenez, Matthew Groh, Roxana Daneshjou, Alexander A Navarini, Marc Pouly, Labelling Consortium, et al. Towards reliable dermatology evaluation benchmarks. In *Machine Learning for Health (ML4H)*, pp. 101–128. PMLR, 2023.

Fabian Gröger, Simone Lionetti, Philippe Gottfrois, Alvaro Gonzalez-Jimenez, Ludovic Amruthalingam, Matthew Groh, Alexander Navarini, and Marc Pouly. Intrinsic self-supervision for data quality audits. *Advances in Neural Information Processing Systems*, 37:92273–92316, 2024.

Fabian Gröger, Simone Lionetti, Philippe Gottfrois, Alvaro Gonzalez-Jimenez, Ludovic Amruthalingam, Elisabeth Victoria Goessinger, Hanna Lindemann, Marie Bargiela, Marie Hofbauer, Omar Badri, et al. Cleanpatrick: A benchmark for image data cleaning. *arXiv preprint arXiv:2505.11034*, 2025.

Matthew Groh, Caleb Harris, Luis Soenksen, Felix Lau, Rachel Han, Aerin Kim, Arash Koochek, and Omar Badri. Evaluating deep neural networks trained on clinical images in dermatology with the fitzpatrick 17k dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1820–1828, 2021.

Groq. Groqcloud supported models, 2025. URL https://console.groq.com/docs/models. Accessed: 2025-05-16.

Xuehang Guo, Xingyao Wang, Yangyi Chen, Sha Li, Chi Han, Manling Li, and Heng Ji. Syncmind: Measuring agent out-of-sync recovery in collaborative software engineering. *arXiv preprint arXiv:2502.06994*, 2025.

Yaming Guo, Kai Guo, Xiaofeng Cao, Tieru Wu, and Yi Chang. Out-of-distribution generalization of federated learning via implicit invariant relationships. In *International Conference on Machine Learning*, pp. 11905–11933. PMLR, 2023.

Shyam Gupta. Fracatlas. https://www.kaggle.com/datasets/shyamgupta196/fracatlas, 2023. Accessed: 2025-05-22.

Izzeddin Gur, Hiroki Furuta, Austin V Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *The Twelfth International Conference on Learning Representations*.

David Gutman, Noel CF Codella, Emre Celebi, Brian Helba, Michael Marchetti, Nabin Mishra, and Allan Halpern. Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (isbi) 2016, hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1605.01397*, 2016.

Haashaatif. Melanoma histopathology dataset. https://www.kaggle.com/datasets/haashaatif/melanoma-histopathology-dataset, 2024. Accessed: 2024-05-22.

Senyu Han, Lu Chen, Li-Min Lin, Zhengshan Xu, and Kai Yu. Ibsen: Director-actor agent collaboration for controllable and interactive drama script generation. *arXiv preprint arXiv:2407.01093*, 2024a.

Shijie Han, Changhai Zhou, Yiqing Shen, Tianning Sun, Yuhua Zhou, Xiaoxia Wang, Zhixiao Yang, Jingshu Zhang, and Hongguang Li. Finsphere: A conversational stock analysis agent equipped with quantitative tools based on real-time database. *arXiv preprint arXiv:2501.12399*, 2025.

Xuewen Han, Neng Wang, Shangkun Che, Hongyang Yang, Kunpeng Zhang, and Sean Xin Xu. Enhancing investment analysis: Optimizing ai-agent collaboration in financial research. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 538–546, 2024b.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring Coding Challenge Competence With APPS, November 2021. URL http://arxiv.org/abs/2105.09938. arXiv:2105.09938 [cs].

HFUTYBX. Mhsi choledoch dataset preprocessed. https://www.kaggle.com/datasets/hfutybx/mhsi-choledoch-dataset-preprocessed-dataset, 2024. Accessed: 2024-05-22.

Hmchuong. X-ray bone shadow suppression. https://www.kaggle.com/datasets/hmchuong/xray-bone-shadow-supression, 2023. Accessed: 2025-05-22.

Huthayfa Hodeb. Nih chest x-rays (bbox version). https://www.kaggle.com/datasets/huthayfahodeb/nih-chest-x-rays-bbox-version, 2023. Accessed: 2025-05-22.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification, 2019. URL https://arxiv.org/abs/1909.06335.

Ruida Hu, Chao Peng, Xinchen Wang, and Cuiyun Gao. An llm-based agent for reliable docker environment configuration. *arXiv preprint arXiv:2502.13681*, 2025.

Dong Huang, Jie M. Zhang, Michael Luck, Qingwen Bu, Yuhao Qing, and Heming Cui. AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation, May 2024a. URL http://arxiv.org/abs/2312.13010. arXiv:2312.13010 [cs].

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating Language Agents on Machine Learning Experimentation. In *Forty-first International Conference on Machine Learning*, June 2024b. URL https://openreview.net/forum?id=1Fs1LvjYQW.

IMT Kaggle Team. Dental radiography dataset. https://www.kaggle.com/datasets/imtkaggleteam/dental-radiography/data, 2023. Accessed: 2025-05-22.

Yoshitaka Inoue, Tianci Song, and Tianfan Fu. Drugagent: Explainable drug repurposing agent with large language model-based reasoning. *arXiv preprint arXiv:2408.13378*, 2024.

Md Ashraful Islam, Mohammed Eunus Ali, and Md Rizwan Parvez. Codesim: Multi-agent code generation and problem solving through simulation-driven planning and debugging. *arXiv preprint arXiv:2502.05664*, 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code, June 2024. URL http://arxiv.org/abs/2403.07974. arXiv:2403.07974 [cs].

Naman Jain, Jaskirat Singh, Manish Shetty, Liang Zheng, Koushik Sen, and Ion Stoica. R2e-gym: Procedural environments and hybrid verifiers for scaling open-weights swe agents. *arXiv preprint arXiv:2504.07164*, 2025.

Yixing Jiang, Kameron C Black, Gloria Geng, Danny Park, James Zou, Andrew Y Ng, and Jonathan H Chen. Medagentbench: A realistic virtual ehr environment to benchmark medical llm agents. *arXiv preprint arXiv:2501.14654*, 2025.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues?, April 2024. URL http://arxiv.org/abs/2310.06770. arXiv:2310.06770 [cs].

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. DSBench: How Far Are Data Science Agents to Becoming Data Science Experts?, September 2024. URL http://arxiv.org/abs/2409.07703. arXiv:2409.07703 [cs].

Lin Justin. Train val test tcga coad msi mss. https://www.kaggle.com/datasets/linjustin/train-val-test-tcga-coad-msi-mss, 2024. Accessed: 2024-05-22.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*. PMLR, 2020.

Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh. Seven-point checklist and skin lesion classification using multitask multimodal neural nets. *IEEE journal of biomedical and health informatics*, 23(2):538–546, 2018.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, Hae Park, et al. Mdagents: An adaptive collaboration

of llms for medical decision-making. *Advances in Neural Information Processing Systems*, 37: 79410–79452, 2024.

Felipe Kitamura. Spr x-ray age and gender dataset. https://www.kaggle.com/datasets/felipekitamura/spr-x-ray-age-and-gender-dataset, 2022a. Accessed: 2025-05-22.

Felipe Kitamura. Unifesp x-ray bodypart classification. https://www.kaggle.com/datasets/felipekitamura/unifesp-xray-bodypart-classification, 2022b. Accessed: 2025-05-22.

Sourabh Kumar. Breast cancer histopathology. https://www.kaggle.com/datasets/sourabhkumar29/breast-cancer-histopathology, 2024. Accessed: 2024-05-22.

Shrinidhi Kumbhar, Venkatesh Mishra, Kevin Coutinho, Divij Handa, Ashif Iquebal, and Chitta Baral. Hypothesis generation for materials discovery and design using goal-driven and constraint-guided llm agents. *arXiv preprint arXiv:2501.13299*, 2025.

The Red Lad. Pannuke dataset (experimental). https://www.kaggle.com/datasets/theredlad/pannuke-dataset-experimental-data, 2024. Accessed: 2024-05-22.

Pamela J LaMontagne, Tammie LS Benzinger, John C Morris, Sarah Keefe, Russ Hornbeck, Chengjie Xiong, Elizabeth Grant, Jason Hassenstab, Krista Moulder, Andrei G Vlassenko, et al. OASIS-3: longitudinal neuroimaging, clinical, and cognitive dataset for normal aging and Alzheimer disease. *MedRxiv*, pp. 1–37, 2019.

Langgraph. Langgraph documentation, 2025. URL https://www.langchain.com/langgraph. Accessed: 2025-05-16.

Gisang Lee, Sangwoo Park, Junyoung Park, Andrew Chung, Sieun Park, Yoonah Park, Byungju Kim, and Min-gyu Cho. Expanding search space with diverse prompting agents: An efficient sampling approach for llm mathematical reasoning. *arXiv preprint arXiv:2410.09780*, 2024.

Jingoo Lee, Kyungho Lim, Young-Chul Jung, and Byung-Hoon Kim. Psyche: A multi-faceted patient simulation framework for evaluation of psychiatric assessment conversational agents. *arXiv preprint arXiv:2501.01594*, 2025.

Tahir Lee. Uterine leiomyosarcoma histopathology. https://www.kaggle.com/datasets/tahirlee/uterine-leiomyosarcoma-histopathology, 2024. Accessed: 2024-05-22.

Bin Lei, Yi Zhang, Shan Zuo, Ali Payani, and Caiwen Ding. Macm: Utilizing a multi-agent system for condition mining in solving complex mathematical problems. *arXiv preprint arXiv:2404.04735*, 2024.

Binxu Li, Tiankai Yan, Yuanting Pan, Jie Luo, Ruiyang Ji, Jiayuan Ding, Zhe Xu, Shilong Liu, Haoyu Dong, Zihao Lin, et al. Mmedagent: Learning to use medical tools with multi-modal agent. In *EMNLP (Findings)*, 2024a.

Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation, 2019. URL https://arxiv.org/abs/1910.03581.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023a.

Long Li, Weiwen Xu, Jiayan Guo, Ruochen Zhao, Xingxuan Li, Yuqian Yuan, Boqiang Zhang, Yuming Jiang, Yifei Xin, Ronghao Dang, et al. Chain of ideas: Revolutionizing research via novel idea development with llm agents. *arXiv preprint arXiv:2410.13185*, 2024b.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning, 2021a. URL https://arxiv.org/abs/2103.16257.

Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2021b.

Qinbin Li, Bingsheng He, and Dawn Song. Adversarial collaborative learning on non-IID features. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19504–19526. PMLR, 23–29 Jul 2023b. URL https://proceedings.mlr.press/v202/li23j.html.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020a.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2: 429–450, 2020b.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization, 2021c. URL https://arxiv.org/abs/2012.04221.

Vincent Li, Yule Fu, Tim Knappe, Kevin Han, and Kevin Zhu. Automating mathematical proof generation using large language model agents and knowledge graphs. *arXiv preprint arXiv:2503.11657*, 2025.

Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization, 2021d. URL https://arxiv.org/abs/2102.07623.

Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B. Allen, Randy P. Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations, 2020. URL https://arxiv.org/abs/2001.01523.

Xun Liang, Jiawei Yang, Yezhaohui Wang, Chen Tang, Zifan Zheng, Simin Niu, Shichao Song, Hanyu Wang, Bo Tang, Feiyu Xiong, et al. Surveyx: Academic survey automation via large language models. *arXiv preprint arXiv:2502.14776*, 2025.

Sook-Lei Liew, Bethany P Lo, Miranda R Donnelly, Artemis Zavaliangos-Petropulu, Jessica N Jeong, Giuseppe Barisano, Alexandre Hutton, Julia P Simon, Julia M Juliano, Anisha Suri, et al. A large, curated, open-source stroke neuroimaging dataset to improve lesion segmentation algorithms. *Scientific data*, 9(1):320, 2022.

Ryan Y Lin, Siddhartha Ojha, Kevin Cai, and Maxwell F Chen. Strategic collusion of llm agents: Market division in multi-commodity competitions. *arXiv preprint arXiv:2410.00031*, 2024.

Ben Liu, Jihan Zhang, Fangquan Lin, Xu Jia, and Min Peng. One size doesn't fit all: A personalized conversational tutoring agent for mathematics instruction. 2025. URL https://arxiv.org/abs/2502.12633.

Hong-Hsiang Liu and Yi-Wen Liu. Agent-driven large language models for mandarin lyric generation. In *2024 27th Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, pp. 1–6. IEEE, 2024.

Renpu Liu, Cong Shen, and Jing Yang. Federated representation learning in the under-parameterized regime, 2024a. URL https://arxiv.org/abs/2406.04596.

Siyi Liu, Chen Gao, and Yong Li. Large Language Model Agent for Hyper-Parameter Optimization, February 2024b. URL http://arxiv.org/abs/2402.01881. arXiv:2402.01881 [cs].

Lokisilvres. Dental disease panoramic detection dataset. https://www.kaggle.com/datasets/lokisilvres/dental-disease-panoramic-detection-dataset, 2023. Accessed: 2025-05-22.

Wang Lu, Jindong Wang, Yiqiang Chen, Xin Qin, Renjun Xu, Dimitrios Dimitriadis, and Tao Qin. Personalized federated learning with adaptive batchnorm for healthcare, 2022. URL https://arxiv.org/abs/2112.00734.

Yuxuan Lu, Bingsheng Yao, Hansu Gu, Jing Huang, Jessie Wang, Laurence Li, Jiri Gesi, Qi He, Toby Jia-Jun Li, and Dakuo Wang. Uxagent: An llm agent-based usability testing framework for web design. *arXiv preprint arXiv:2502.12561*, 2025.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, et al. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*, 2025.

Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data, 2021. URL https://arxiv.org/abs/2106.05001.

Tianmi Ma, Jiawei Du, Wenxin Huang, Wenjie Wang, Liang Xie, Xian Zhong, and Joey Tianyi Zhou. Llm knows geometry better than algebra: Numerical understanding of llm-based agents in a trading arena. *arXiv preprint arXiv:2502.17967*, 2025.

Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Layer-wised model aggregation for personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10092–10101, June 2022.

Nikita Manaenkov. Annotated x-ray angiography dataset. https://www.kaggle.com/datasets/nikitamanaenkov/annotated-x-ray-angiography-dataset, 2023. Accessed: 2025-05-22.

Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions, 2022. URL https://arxiv.org/abs/2108.10252.

Antonis Maronikolakis, Ana Peleteiro Ramallo, Weiwei Cheng, and Thomas Kober. What should i wear to a party in a greek taverna? evaluation for conversational agents in the fashion domain. *arXiv preprint arXiv:2408.08907*, 2024.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 2017.

Kai Mei, Xi Zhu, Wujiang Xu, Wenyue Hua, Mingyu Jin, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, and Yongfeng Zhang. Aios: Llm agent operating system. *arXiv preprint arXiv:2403.16971*, 2024.

Bjoern H Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, et al. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 34(10): 1993–2024, 2014.

Arindam Mitra, Luciano Del Corro, Guoqing Zheng, Shweti Mahajan, Dany Rouhana, Andres Codas, Yadong Lu, Wei-ge Chen, Olga Vrousgos, Corby Rosset, et al. Agentinstruct: Toward generative teaching with agentic flows. *arXiv preprint arXiv:2407.03502*, 2024.

Paul Timothy Mooney. Chest x-ray pneumonia. https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia, 2018. Accessed: 2025-05-22.

Paul Timothy Mooney. Breast histopathology images. https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images, 2024. Accessed: 2024-05-22.

WSH Munirah. Npc-88k-public dataset. https://www.kaggle.com/datasets/wshmunirah/npc-88k-public, 2024. Accessed: 2024-05-22.

Andrew MVD. Lung and colon cancer histopathological images. https://www.kaggle.com/datasets/andrewmvd/lung-and-colon-cancer-histopathological-images, 2024a. Accessed: 2024-05-22.

Andrew MVD. Breast cancer cell segmentation. https://www.kaggle.com/datasets/andrewmvd/breast-cancer-cell-segmentation, 2024b. Accessed: 2024-05-22.

Mouheb Ben Nasr and Yassine Hachaïchi. Reinforcement learning agent for client selection in federated llms.

A. Tuan Nguyen, Philip Torr, and Ser Nam Lim. Fedsr: A simple and effective domain generalization method for federated learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 38831–38843. Curran Associates, Inc., 2022a. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/fd946a6c99541fddc3d64a3ea39a1bc2-Paper-Conference.pdf.

Dinh C Nguyen, Quoc-Viet Pham, Pubudu N Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia Dobre, and Won-Joo Hwang. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(3):1–37, 2022b.

Huan Ning, Zhenlong Li, Temitope Akinboyewa, and M Naser Lessani. An autonomous gis agent framework for geospatial data retrieval. *International Journal of Digital Earth*, 18(1):2458688, 2025.

Jaehoon Oh, Sangmook Kim, and Se-Young Yun. Fedbabu: Towards enhanced representation for federated image classification, 2022. URL https://arxiv.org/abs/2106.06042.

Izunna Okpala, Ashkan Golgoon, and Arjun Ravi Kannan. Agentic ai systems applied to tasks in financial services: Modeling and model risk management crews. *arXiv preprint arXiv:2502.05439*, 2025.

OpenAI. Openai api documentation, 2025. URL https://openai.com/api/. Accessed: 2025-05-16.

Orvile. Bone fracture dataset. https://www.kaggle.com/datasets/orvile/bone-fracture-dataset, 2023a. Accessed: 2025-05-22.

Orvile. Simple vs comminuted fractures x-ray data. https://www.kaggle.com/datasets/orvile/simple-vs-comminuted-fractures-x-ray-data, 2023b. Accessed: 2025-05-22.

Orvile. Human bone fractures image dataset (hbfmid). https://www.kaggle.com/datasets/orvile/human-bone-fractures-image-dataset-hbfmid, 2023c. Accessed: 2025-05-22.

Orvile. Digital knee x-ray images. https://www.kaggle.com/datasets/orvile/digital-knee-x-ray-images, 2023d. Accessed: 2025-05-22.

Orvile. Gastric cancer histopathology tissue image dataset. https://www.kaggle.com/datasets/orvile/gastric-cancer-histopathology-tissue-image-dataset, 2024. Accessed: 2024-05-22.

Andre GC Pacheco, Gustavo R Lima, Amanda S Salomão, Breno Krohling, Igor P Biral, Gabriel G de Angelo, Fábio CR Alves Jr, José GM Esgario, Alana C Simora, Pedro BC Castro, et al. PAD-UFES-20: A skin lesion dataset composed of patient data and clinical images collected from smartphones. *Data in Brief*, 32:1–10, 2020.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with swe-gym. *arXiv preprint arXiv:2412.21139*, 2024.

Emilio Paspuel. Deep learning driven diagnosis of humerus fractures. https://www.kaggle.com/datasets/emiliopaspuel/

deeplearning-driven-diagnosis-of-humerus-fractures, 2024. Accessed: 2025-05-22.

Bjarne Pfitzner, Nico Steckhan, and Bert Arnrich. Federated learning in a medical context: A systematic literature review. *ACM Transactions on Internet Technology (TOIT)*, 21(2):1–31, 2021.

Bits N Pieces. Ovarian cancer and subtypes dataset histopathology. https://www.kaggle.com/datasets/bitsnpieces/ovarian-cancer-and-subtypes-dataset-histopathology, 2024. Accessed: 2024-05-22.

Jianing Qiu, Kyle Lam, Guohao Li, Amish Acharya, Tien Yin Wong, Ara Darzi, Wu Yuan, and Eric J Topol. Llm-based agentic systems in medicine and healthcare. *Nature Machine Intelligence*, 6(12): 1418–1420, 2024.

Xiangyan Qu, Gaopeng Gou, Jiamin Zhuang, Jing Yu, Kun Song, Qihao Wang, Yili Li, and Gang Xiong. Proapo: Progressively automatic prompt optimization for visual classification. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 25145–25155, 2025.

Tawsifur Rahman. Covid-19 radiography database. https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database, 2020. Accessed: 2025-05-22.

Tawsifur Rahman. Aseptic loose hip implant x-ray database. https://www.kaggle.com/datasets/tawsifurrahman/aseptic-loose-hip-implant-xray-database, 2022. Accessed: 2025-05-22.

Suraj Rajendran, Jihad S Obeid, Hamidullah Binol, Kristie Foley, Wei Zhang, Philip Austin, Joey Brakefield, Metin N Gurcan, and Umit Topaloglu. Cloud-based federated learning implementation across medical centers. *JCO Clinical Cancer Informatics*, 5:1–11, 2021.

Kiran Ramnath, Kang Zhou, Sheng Guan, Soumya Smruti Mishra, Xuan Qi, Zhengyuan Shen, Shuai Wang, Sangmin Woo, Sullam Jeoung, Yawei Wang, et al. A systematic survey of automatic prompt optimization techniques. *arXiv preprint arXiv:2502.16923*, 2025.

RANZCR. Ranzcr clip - catheter and line classification. https://www.kaggle.com/c/ranzcr-clip-catheter-line-classification, 2021. Accessed: 2025-05-22.

Reasat. Histo image-text. https://www.kaggle.com/datasets/reasat/histo-img-text, 2024. Accessed: 2024-05-22.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization, 2021. URL https://arxiv.org/abs/2003.00295.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):119, 2020.

B Madushani Rodrigo. Fracture multi-region x-ray data. https://www.kaggle.com/datasets/bmadushanirodrigo/fracture-multi-region-x-ray-data, 2022. Accessed: 2025-05-22.

Daniel Rose, Chia-Chien Hung, Marco Lepri, Israa Alqassem, Kiril Gashteovski, and Carolin Lawrence. Meddxagent: A unified modular agent framework for explainable automatic differential diagnosis. *arXiv preprint arXiv:2502.19175*, 2025.

Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints, 2019. URL https://arxiv.org/abs/1910.01991.

Samuel Schmidgall and Michael Moor. Agentrxiv: Towards collaborative autonomous research. *arXiv preprint arXiv:2503.18102*, 2025.

Dominik Schmidt, Zhengyao Jiang, and Yuxiang Wu. Introducing Weco AIDE, April 2024. URL https://www.weco.ai/blog/technical-report.

Jonathan Scott, Hossein Zakerinia, and Christoph H. Lampert. Pefll: Personalized federated learning by learning to learn, 2025. URL https://arxiv.org/abs/2306.05515.

Constantin Seibold. Anatomy in chest x-rays (pax-ray++). https://www.kaggle.com/datasets/constantinseibold/anatomy-in-chest-x-rays-pax-ray, 2023. Accessed: 2025-05-22.

Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks, 2021. URL https://arxiv.org/abs/2103.04628.

Zitong Shi, Guancheng Wan, Wenke Huang, Guibin Zhang, Jiawei Shao, Mang Ye, and Carl Yang. Privacy-enhancing paradigms within federated multi-agent systems. In *ICML 2025 Workshop on Collaborative and Federated Agentic Workflows*.

Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. Agentic retrieval-augmented generation: A survey on agentic rag. *arXiv preprint arXiv:2501.09136*, 2025.

Jayanthi Sivaswamy, SR Krishnadas, Gopal Datt Joshi, Madhulika Jain, and A Ujjwaft Syed Tabish. Drishti-gs: Retinal image dataset for optic nerve head (onh) segmentation. In *2014 IEEE 11th international symposium on biomedical imaging (ISBI)*, pp. 53–56. IEEE, 2014.

Spritan1. Yolo annotated chestxray 8 object detection. https://www.kaggle.com/datasets/spritan1/yolo-annotated-chestxray-8-object-detection, 2023. Accessed: 2025-05-22.

Ian Steenstra, Farnaz Nouraei, and Timothy W Bickmore. Scaffolding empathy: Training counselors with simulated patients and utterance-level performance visualizations. *arXiv preprint arXiv:2502.18673*, 2025.

Osama H Taher. Heel dataset. https://www.kaggle.com/datasets/osamahtaher/heel-dataset, 2023. Accessed: 2025-05-22.

Jiahao Tan and Xinpeng Wang. FL-bench: A federated learning benchmark for solving image classification tasks. URL https://github.com/KarhouTam/FL-bench.

Jiahao Tan, Yipeng Zhou, Gang Liu, Jessie Hui Wang, and Shui Yu. pfedsim: Similarity-aware model aggregation towards personalized federated learning. *arXiv preprint arXiv:2305.15706*, 2023.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients, 2022. URL https://arxiv.org/abs/2105.00243.

Xiangru Tang, Yuliang Liu, Zefan Cai, Yanjun Shao, Junjie Lu, Yichi Zhang, Zexuan Deng, Helan Hu, Kaikai An, Ruijun Huang, Shuzheng Si, Sheng Chen, Haozhe Zhao, Liang Chen, Yan Wang, Tianyu Liu, Zhiwei Jiang, Baobao Chang, Yin Fang, Yujia Qin, Wangchunshu Zhou, Yilun Zhao, Arman Cohan, and Mark Gerstein. ML-Bench: Evaluating Large Language Models and Agents for Machine Learning Tasks on Repository-Level Code, August 2024. URL http://arxiv.org/abs/2311.09835.

Xiangru Tang, Tianyu Hu, Muyang Ye, Yanjun Shao, Xunjian Yin, Siru Ouyang, Wangchunshu Zhou, Pan Lu, Zhuosheng Zhang, Yilun Zhao, et al. Chemagent: Self-updating library in large language models improves chemical reasoning. *arXiv preprint arXiv:2501.06590*, 2025.

Tapendu. Chest x-ray dataset for tuberculosis segmentation. https://www.kaggle.com/datasets/iamtapendu/chest-x-ray-lungs-segmentation, 2023a. Accessed: 2025-05-22.

Tapendu. Rsna pneumonia processed dataset. https://www.kaggle.com/datasets/iamtapendu/rsna-pneumonia-processed-dataset, 2023b. Accessed: 2025-05-22.

Mr. Tejas. Covid-19 and normal x-ray dataset (balanced). https://www.kaggle.com/datasets/mrtejas/covid-19-and-normal-x-ray-dataset-balanced, 2022. Accessed: 2025-05-22.

Aryamaan Thakur. Rsna breast cancer detection roi 1024. https://www.kaggle.com/datasets/aryamaanthakur/rsna-breast-cancer-detection-roi-1024, 2024. Accessed: 2025-05-22.

TrainingDataPro. Chest x-ray 17 diseases. https://www.kaggle.com/datasets/trainingdatapro/chest-xray-17-diseases, 2023. Accessed: 2025-05-22.

Prashant Trivedi, Souradip Chakraborty, Avinash Reddy, Vaneet Aggarwal, Amrit Singh Bedi, and George K Atia. Align-pro: A principled approach to prompt optimization for llm alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27653–27661, 2025.

TruthIsNeverLinear. Brecahad. https://www.kaggle.com/datasets/truthisneverlinear/brecahad, 2024. Accessed: 2024-05-22.

Preet Viradiya. Covid-19 radiography dataset extended. https://www.kaggle.com/datasets/preetviradiya/covid19-radiography-dataset, 2023. Accessed: 2025-05-22.

Xiangpeng Wan, Haicheng Deng, Kai Zou, and Shiqi Xu. Enhancing the efficiency and accuracy of underlying asset reviews in structured finance: The application of multi-agent framework. *arXiv preprint arXiv:2405.04294*, 2024.

Cunshi Wang, Xinjie Hu, Yu Zhang, Xunhao Chen, Pengliang Du, Yiming Mao, Rui Wang, Yuyang Li, Ying Wu, Hang Yang, et al. Starwhisper telescope: Agent-based observation assistant system to approach ai astrophysicist. *arXiv preprint arXiv:2412.06412*, 2024a.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Jian Wang, Yinpei Dai, Yichi Zhang, Ziqiao Ma, Wenjie Li, and Joyce Chai. Training turn-by-turn verifiers for dialogue tutoring agents: The curious case of llms as your coding tutors. *arXiv preprint arXiv:2502.13311*, 2025a.

Jiuniu Wang, Zehua Du, Yuyuan Zhao, Bo Yuan, Kexiang Wang, Jian Liang, Yaxi Zhao, Yihen Lu, Gengliang Li, Junlong Gao, et al. Aesopagent: Agent-driven evolutionary system on story-to-video production. *arXiv preprint arXiv:2403.07952*, 2024b.

Ruida Wang, Rui Pan, Yuxin Li, Jipeng Zhang, Yizhen Jia, Shizhe Diao, Renjie Pi, Junjie Hu, and Tong Zhang. Ma-lot: Multi-agent lean-based long chain-of-thought reasoning enhances formal theorem proving. *arXiv preprint arXiv:2503.03205*, 2025b.

Xin Wang, Yifan Zhang, Xiaojing Zhang, Longhui Yu, Xinna Lin, Jindong Jiang, Bin Ma, and Kaicheng Yu. Patentagent: Intelligent agent for automated pharmaceutical patent analysis. *arXiv preprint arXiv:2410.21312*, 2024c.

Zhizheng Wang, Qiao Jin, Chih-Hsuan Wei, Shubo Tian, Po-Ting Lai, Qingqing Zhu, Chi-Ping Day, Christina Ross, and Zhiyong Lu. Geneagent: self-verification language agent for gene set knowledge discovery using domain databases. *arXiv preprint arXiv:2405.16205*, 2024d.

Ziyue Wang, Junde Wu, Chang Han Low, and Yueming Jin. Medagent-pro: Towards multi-modal evidence-based medical diagnosis via reasoning agentic workflow. *arXiv preprint arXiv:2503.18968*, 2025c.

Abbi Ward, Jimmy Li, Julie Wang, Sriram Lakshminarasimhan, Ashley Carrick, Bilson Campana, Jay Hartford, Pradeep K. Sreenivasaiah, Tiya Tiyasirisokchai, Sunny Virmani, Renee Wong, Yossi Matias, Greg S. Corrado, Dale R. Webster, Margaret Ann Smith, Dawn Siegel, Steven Lin, Justin Ko, Alan Karthikesalingam, Christopher Semturs, and Pooja Rao. Creating an empirical dermatology dataset through crowdsourcing with web search advertisements. *JAMA Network Open*, 7(11):

e2446615–e2446615, 11 2024. ISSN 2574-3805. 10.1001/jamanetworkopen.2024.46615. URL https://doi.org/10.1001/jamanetworkopen.2024.46615.

Robert Wasenmüller, Kevin Hilbert, and Christoph Benzmüller. Script-based dialog policy planning for llm-powered conversational agents: A basic architecture for an" ai therapist". *arXiv preprint arXiv:2412.15242*, 2024.

Panlong Wu, Kangshuo Li, Junbao Nan, and Fangxin Wang. Federated in-context llm agent learning. *arXiv preprint arXiv:2412.08054*, 2024.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Wenbei Xie, Donglin Liu, Haoran Yan, Wenjie Wu, and Zongyang Liu. Mathlearner: A large language model agent framework for learning to solve mathematical problems. *arXiv preprint arXiv:2408.01779*, 2024.

Ancheng Xu, Di Yang, Renhao Li, Jingwei Zhu, Minghuan Tan, Min Yang, Wanxin Qiu, Mingchen Ma, Haihong Wu, Bingyu Li, et al. Autocbt: An autonomous multi-agent framework for cognitive behavioral therapy in psychological counseling. *arXiv preprint arXiv:2501.09426*, 2025a.

Zhenran Xu, Longyue Wang, Jifang Wang, Zhouyi Li, Senbao Shi, Xue Yang, Yiyu Wang, Baotian Hu, Jun Yu, and Min Zhang. Filmagent: A multi-agent framework for end-to-end film automation in virtual 3d spaces. *arXiv preprint arXiv:2501.12909*, 2025b.

Jian Yang, Wei Zhang, Jiaxi Yang, Yibo Miao, Shanghaoran Quan, Zhenhe Wu, Qiyao Peng, Liqun Yang, Tianyu Liu, Zeyu Cui, et al. Multi-agent collaboration for multilingual code instruction tuning. *arXiv preprint arXiv:2502.07487*, 2025a.

Xiyuan Yang, Wenke Huang, and Mang Ye. Fedas: Bridging inconsistency in personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11986–11995, June 2024.

Yizhe Yang, Palakorn Achananuparp, Heyan Huang, Jing Jiang, Kit Phey Leng, Nicholas Gabriel Lim, Cameron Tan Shi Ern, and Ee-peng Lim. Cami: A counselor agent supporting motivational interviewing through state inference and topic exploration. *arXiv preprint arXiv:2502.02807*, 2025b.

Yuzhe Yang, Yifei Zhang, Minghao Wu, Kaidi Zhang, Yunmiao Zhang, Honghai Yu, Yan Hu, and Benyou Wang. Twinmarket: A scalable behavioral and socialsimulation for financial markets. *arXiv preprint arXiv:2502.01506*, 2025c.

Zhiqin Yang, Yonggang Zhang, Yu Zheng, Xinmei Tian, Hao Peng, Tongliang Liu, and Bo Han. Fedfed: Feature distillation against data heterogeneity in federated learning, 2023. URL https://arxiv.org/abs/2310.05077.

Beibei Yu, Tao Shen, Hongbin Na, Ling Chen, and Denqi Li. Mineagent: Towards remote-sensing mineral exploration with multimodal large language models. *arXiv preprint arXiv:2412.17339*, 2024a.

Dingyao Yu, Kaitao Song, Peiling Lu, Tianyu He, Xu Tan, Wei Ye, Shikun Zhang, and Jiang Bian. Musicagent: An ai agent for music understanding and generation with large language models. *arXiv preprint arXiv:2310.11954*, 2023.

Yangyang Yu, Zhiyuan Yao, Haohang Li, Zhiyang Deng, Yuechen Jiang, Yupeng Cao, Zhi Chen, Jordan Suchow, Zhenyu Cui, Rong Liu, et al. Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making. *Advances in Neural Information Processing Systems*, 37:137010–137045, 2024b.

Murong Yue, Wenhan Lyu, Wijdane Mifdal, Jennifer Suh, Yixuan Zhang, and Ziyu Yao. Mathvc: An llm-simulated multi-character virtual classroom for mathematics education. *arXiv preprint arXiv:2404.06711*, 2024.

Taedong Yun, Eric Yang, Mustafa Safdari, Jong Ha Lee, Vaishnavi Vinod Kumar, S Sara Mahdavi, Jonathan Amar, Derek Peyton, Reut Aharony, Andreas Michaelides, et al. Sleepless nights, sugary days: Creating synthetic users with health conditions for realistic coaching agent interactions. *arXiv preprint arXiv:2502.13135*, 2025.

Jingying Zeng, Hui Liu, Zhenwei Dai, Xianfeng Tang, Chen Luo, Samarth Varshney, Zhen Li, and Qi He. Cite before you speak: Enhancing context-response grounding in e-commerce conversational llm-agents. *arXiv preprint arXiv:2503.04830*, 2025.

Huan Zhang, Yu Song, Ziyu Hou, Santiago Miret, and Bang Liu. Honeycomb: A flexible llm-based agent system for materials science. *arXiv preprint arXiv:2409.00135*, 2024a.

Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedala: Adaptive local aggregation for personalized federated learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11237–11244, June 2023. ISSN 2159-5399. 10.1609/aaai.v37i9.26330. URL http://dx.doi.org/10.1609/aaai.v37i9.26330.

Jie Zhang, Zhiqi Li, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, and Chao Wu. Federated learning with label distribution skew via logits calibration, 2022. URL https://arxiv.org/abs/2209.00189.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025.

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization, 2021. URL https://arxiv.org/abs/2012.08565.

Ran Zhang and Steffen Eger. Llm-based multi-agent poetry generation in non-cooperative environments. *arXiv preprint arXiv:2409.03659*, 2024.

Yiqun Zhang, Xiaocui Yang, Xiaobai Li, Siyuan Yu, Yi Luan, Shi Feng, Daling Wang, and Yifei Zhang. Psydraw: A multi-agent multimodal system for mental health screening in left-behind children. *arXiv preprint arXiv:2412.14769*, 2024b.

Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. AutoCodeRover: Autonomous Program Improvement, July 2024c. URL http://arxiv.org/abs/2404.05427. arXiv:2404.05427 [cs].

Mingkai Zheng, Xiu Su, Shan You, Fei Wang, Chen Qian, Chang Xu, and Samuel Albanie. Can GPT-4 Perform Neural Architecture Search?, August 2023. URL http://arxiv.org/abs/2304.10970. arXiv:2304.10970 [cs].

Yuan Zhou, Peng Zhang, Mengya Song, Alice Zheng, Yiwen Lu, Zhiheng Liu, Yong Chen, and Zhaohan Xi. Zodiac: A cardiologist-level llm framework for multi-agent diagnostics. *arXiv preprint arXiv:2410.02026*, 2024.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning, 2021. URL https://arxiv.org/abs/2105.10056.

APPENDIX

CONTENTS

A Related Works

A.1 Federated Learning for Medical Image Analysis

Existing research on federated learning (FL) in medical image analysis has primarily focused on the development of machine learning algorithms to address technical challenges, such as data distribution shift, statistical and system heterogeneity, and communication efficiency (Antunes et al., 2022; Rajendran et al., 2021; Nguyen et al., 2022b; Pfitzner et al., 2021; Rieke et al., 2020). These efforts have produced a wide range of methods tailored for robust and scalable training under diverse and decentralized medical data environments. However, despite these advances, a significant barrier to real-world deployment persists: the complex set of operational and human-in-the-loop challenges encountered in practice.

Notably, existing FL benchmarks and studies rarely account for the intricacies of human factors—such as institutional workflows, task specification, annotation and curation requirements, and the expertise needed to orchestrate the entire FL pipeline across multiple healthcare institutions. These operational hurdles, including coordination among stakeholders, error handling, and workflow reproducibility, often constitute the most substantial obstacles to routine FL adoption in clinical settings.

This paper distinguishes itself from prior work by explicitly modeling and integrating these real-world operational challenges into the benchmarking process. By capturing both the algorithmic and human-centered aspects of FL deployment, our benchmark provides a more comprehensive and realistic evaluation platform. This enables the research community to move beyond algorithm-centric benchmarks and address the "elephant in the room", *i.e.*, the operational bottlenecks that ultimately determine the success or failure of federated learning in medical imaging practice.

A.2 LLM Agent Applications

AI agents, powered by large language models (LLMs), autonomous tool use, and decision-making workflows, are rapidly transforming a diverse range of application domains. In **healthcare**, LLM-based agents drive advances in clinical diagnosis (Chen et al., 2024; Zhou et al., 2024; Wang et al., 2025c; Rose et al., 2025; Ghezloo et al., 2025; Li et al., 2024a; Jiang et al., 2025; Kim et al., 2024; Fallahpour et al.), mental health and therapy (Wasenmüller et al., 2024; Du et al., 2024; Zhang et al., 2024b; Lee et al., 2025; Xu et al., 2025a; Yang et al., 2025b; Steenstra et al., 2025; Abbasi et al., 2025), workflow optimization (Feng et al., 2025; Yun et al., 2025; Chen et al., 2025d), and pharmaceutical research (Wang et al., 2024c; Averly et al., 2025; Inoue et al., 2024). These agents support professionals through transparent reasoning, multi-modal data integration, and interactive, explainable decision support, as well as automated data processing and clinical research acceleration.

In **biomedical and materials science**, agents enhance literature analysis and hypothesis generation (Liang et al., 2025; Li et al., 2024b; Schmidgall & Moor, 2025; Gottweis et al., 2025), automate gene set knowledge discovery (Wang et al., 2024d), and orchestrate complex scientific workflows, including astronomical observation (Wang et al., 2024a) and materials design (Zhang et al., 2024a; Kumbhar et al., 2025).

The field of **software engineering** benefits from LLM agents for code generation, repair, verification, and environment setup (Dong, 2025; Jain et al., 2025; Wang et al., 2025a; Chen et al., 2025b; Aggarwal et al., 2025; Chen et al., 2025c; Gholamzadeh Khoee et al., 2025; Hu et al., 2025; Lu et al., 2025; Pan et al., 2024; Yang et al., 2025a; Guo et al., 2025; Islam et al., 2025). These agents leverage specialized architectures, collaborative multi-agent strategies, and benchmarking frameworks for automated programming, debugging, and user experience testing.

In **finance**, AI agents automate structured finance workflows, simulate markets, optimize investment decisions, and manage risk (Wan et al., 2024; Yang et al., 2025c; Yu et al., 2024b; Lin et al., 2024; Fatemi & Hu, 2024; Han et al., 2024b; 2025; Fatouros et al., 2025; Okpala et al., 2025; Zeng et al., 2025). Multi-agent frameworks enable complex reasoning, robust QA, and the generation of explainable financial reports.

Synthetic data generation is advanced through multi-agent orchestration frameworks (Mitra et al., 2024), improving post-training data quality and scalability for large language models.

In **chemistry and materials**, agents automate chemical reasoning (Cho et al., 2025; Tang et al., 2025), accelerate drug and materials discovery, and enable hypothesis-driven research (Zhang et al., 2024a; Kumbhar et al., 2025).

Mathematics education and scientific reasoning have seen the development of multi-agent reasoning and tutoring systems to tackle complex mathematical proofs, theorem proving, and adaptive instruction (Lei et al., 2024; Xie et al., 2024; Lee et al., 2024; Deng & Mineiro, 2024; Li et al., 2025; Wang et al., 2025b; Yue et al., 2024; Liu et al., 2025; Ma et al., 2025).

In **geospatial science**, agents facilitate autonomous GIS analysis and data retrieval (Yu et al., 2024a; Ning et al., 2025), addressing the challenge of spatial reasoning and multi-source data fusion.

The domain of **multimedia and creative industries** is being transformed by AI agents capable of automating film production, music and lyric generation, story-to-video creation, fashion assistance, and poetry composition (Xu et al., 2025b; Wang et al., 2024b; Han et al., 2024a; Maronikolakis et al., 2024; Deng et al., 2024; Yu et al., 2023; Zhang & Eger, 2024; Liu & Liu, 2024). These systems support multi-modal content creation and human-AI co-creation.

Overall, the emergence of LLM-powered agents marks a shift toward highly automated, context-aware, and collaborative AI systems with applications spanning healthcare, science, engineering, finance, education, and the creative arts.

A.3 LLM Agents for Machine Learning, Software Engineering, and Federated Learning

The intersection of large language models (LLMs) and autonomous agents has made rapid advancements in machine learning and software engineering. Several works (Chen et al., 2021; Hendrycks et al., 2021; Austin et al., 2021; Jain et al., 2024) assess model performance on code generation from natural language instructions. For example: AgentCoder (Huang et al., 2024a) reports 96.3% and 91.8% accuracy on HumanEval and MBPP, respectively. SWE-bench (Jimenez et al., 2024) advances the field by requiring models to resolve real-world pull requests from open-source repositories. Notably, model performance on SWE-bench continues to improve steadily (Zhang et al., 2024c; factory.ai, 2024).

Prior work has also leveraged LLMs for tasks such as hyperparameter optimization (Liu et al., 2024b) and neural architecture design (Zheng et al., 2023). MLAgentBench (Huang et al., 2024b) evaluates agents on 13 Kaggle and custom ML tasks, providing a baseline solution for each and measuring whether agents can achieve at least a 10% improvement. Similarly, ML-Bench (Tang et al., 2024) evaluates an agent's ability to generate code and interact with established ML repositories. AIDE, as reported by Weco AI (Schmidt et al., 2024), surpasses more than 50% of human competitors in Kaggle-style data science contests. DSBench (Jing et al., 2024) also introduces a Kaggle competition benchmark, but, like Weco AI, focuses primarily on data science tasks.

While benchmarking LLM agents for automated machine learning and data science has gained momentum across both academia and industry, all of these operate under the assumption of a centralized, single-site environment, limiting their applicability to the federated learning paradigm, which introduces unique challenges such as distributed data silos, partial observability, and multi-party coordination. Recent works on agentic FL frameworks include in-context learning in FL of LLM agents (Wu et al., 2024), reinforcement learning agent for client selection (Nasr & Hachaïchi), and privacy enhancing techniques in federated mult-agent systems (Shi et al.).

In contrast to these works, **FedAgentBench** is designed to address the real-world operational complexities in federated learning workflows by evaluating the agentic capabilities — particularly in high-stakes healthcare settings. **Rather than being "yet another" benchmark, FedAgentBench is motivated by a concrete and pressing need to reduce the human coordination bottlenecks that currently hinder scalable deployment of FL in practice.** It provides a realistic testbed for assessing agent autonomy, adaptability, and reasoning in decentralized, privacy-preserving environments.

B TOOLS AND AGENTS IN FEDAGENTBENCH FRAMEWORK

B.1 COLLECTION OF TOOLS ACCESSED BY THE LLM AGENTS

The following tools form the operational backbone of the LLM-based agents, enabling tasks such as file inspection, dataset organization, data cleaning, folder manipulation, and federated training orchestration. Corresponding code snippets for all 16 tools can be found in Listing 1.

1. **read_files**: Reads the content of one or more specified files and returns a dictionary mapping file paths to their contents. It supports UTF-8 text files and handles file access errors gracefully.
2. **move_directory**: Moves a source directory (including all files and subfolders) to a new destination.
3. **copy_files**: Copies multiple individual files to specified destination paths. Accepts a mapping of source to destination file paths and ensures target directories are created as needed.
4. **write_file**: Writes a given text string to a specified file path. It creates any missing directories in the path before writing.
5. **edit_file**: Overwrites the contents of a specified file with new content. Used for completely replacing existing file content.
6. **run_script**: Executes a given shell command (typically a Python script) using a secure subprocess or shell tool backend. Returns the result of the command execution.
7. **list_files_in_second_level**: Traverses the second-level entries of a root directory. For each subdirectory or file, it collects and returns metadata including the total number of files and a preview list of file paths (up to 10).
8. **preview_file_content**: Previews the contents of a CSV, JSON, or TXT file. Returns first 5 rows or entries and summary statistics such as total rows or elements.
9. **run_selfclean_on_dataset**: Runs the data cleaning framework on an image folder to detect and optionally clean near duplicates, off-topic or irrelevant samples, and label errors. It generates internal diagnostic data in CSV format for inspection and removes samples based on a threshold. Within this process, we also achieve normalization and standardization.
10. **organize_into_subfolder**: Reads a CSV containing image paths and labels, and organizes the corresponding images into class-specific subfolders within a specified destination directory.
11. **copy_folder**: Copies all contents (files and subfolders) from a source directory to a destination directory. Ensures destination exists and performs a recursive copy.
12. **remove_other_files**: Recursively removes all non-image files from a directory structure. Keeps standard image formats (e.g., .jpg, .png, .bmp) and deletes all others.
13. **list_folders**: Returns the names of all first-level subdirectories under a specified root directory. Useful for summarizing dataset structure.
14. **make_folder**: Creates a new directory at a specified path. Used to set up target folders during label harmonization or preprocessing.
15. **copy_images**: Copies all image files from a source folder to a specified target folder. Typically used during label harmonization to reorganize class-wise images.
16. **run_federated_method**: Launches federated learning using a specified algorithm and project directory. Executes a Python script with algorithm-specific parameters and returns algorithm performance.

Listing 1: Repository of tools used by LLM Agents

```
1. def read_files(file_paths: list) -> dict:
    """
    Read file contents and return as dictionary.

    Args:
        file_paths: List of file paths to read
```

```
    Returns:
        dict: Dictionary with {file_path: file_content} format
    """
    file_contents = {}

    for file_path in file_paths:
        try:
            with open(file_path, 'r', encoding='utf-8') as file:
                content = file.read()
                file_contents[file_path] = content
        except (UnicodeDecodeError, PermissionError, FileNotFoundError)
            as e:
            print(f"Cannot read file {file_path}: {e}")
            file_contents[file_path] = None

    return file_contents

2. def move_directory(src_dir: str, dest_dir: str) -> str:
    """
    Move source directory and its contents to destination directory,
        creating a new subdirectory
    with the same name as the source directory.

    Args:
        src_dir: Source directory path (e.g., '/path/to/source/
            folder_name')
        dest_dir: Parent destination directory path (e.g., '/path/to/dest
            ')
                A new subdirectory named 'folder_name' will be created
                    here

    Returns:
        str: Operation result message

    Example:
        If src_dir is '/path/to/source/folder_name' and dest_dir is '/
            path/to/dest',
        the directory will be moved to '/path/to/dest/folder_name'
    """
    print(f"Running move_directory tool to move from {src_dir} to {
        dest_dir}...")
    try:
        if not os.path.exists(src_dir):
            return f"Source directory {src_dir} does not exist"

        # Get the source directory name
        src_name = os.path.basename(src_dir.rstrip('/'))
        target_dir = os.path.join(dest_dir, src_name)

        # If destination directory already exists, remove it first
        if os.path.exists(target_dir):
            shutil.rmtree(target_dir)

        # Move the directory
        shutil.move(src_dir, target_dir)
        return f"Directory {src_dir} has been successfully moved to {
            target_dir}"

    except Exception as e:
        return f"Error moving directory: {str(e)}"


3. def copy_files(file_mapping: dict) -> str:
    """
```

```
    Copy multiple files from source paths to destination paths.

    Args:
        file_mapping (dict): A dictionary where keys are source file
            paths and values are destination file paths.
            Example: {
                "/path/to/source1.txt": "/path/to/destination1.txt",
                "/path/to/source2.txt": "/path/to/destination2.txt"
            }

    Returns:
        str: A message indicating the result of the operation.
    """
    print(f"Running copy_files tool to copy {file_mapping}...")
    results = []
    for src, dest in file_mapping.items():
        try:
            # Check if source file exists
            if not os.path.exists(src):
                results.append(f"Source file {src} does not exist.")
                continue

            # Create destination directory if it doesn't exist
            dest_directory = os.path.dirname(dest)
            if not os.path.exists(dest_directory):
                os.makedirs(dest_directory)

            # Copy file
            shutil.copy2(src, dest)
            results.append(f"File {src} successfully copied to {dest}")

        except Exception as e:
            results.append(f"Error copying file {src}: {e}")

    # Return summary of all operations
    return "\n".join(results)
```

```
4. def write_file(content: str, file_path: str) -> None:
    """
    Write a given string of code to a specified file.

    This function creates the necessary directories for the file (if they
        don't exist),
    writes the content to the file, and handles any errors that may occur
        during the process.

    Args:
        content (str): The code or text you want to write into the file.
        file_path (str): The full path (including filename) where the
            content will be saved.

    Example:
        write_file('print("Hello World")', 'scripts/hello.py')
    """
    print(f"Running write_file tool to write {file_path}...")
    try:
        os.makedirs(os.path.dirname(file_path), exist_ok=True)

        with open(file_path, 'w', encoding='utf-8') as file:
            file.write(content)

        print(f"File successfully written to: {file_path}")
    except Exception as e:
        print(f"Error writing file: {e}")
```

```
5. def edit_file(new_content: str, file_path: str) -> None:
    """
    Completely overwrite a file with new content. The original file
        content will be replaced entirely.

    Args:
        new_content: Complete content to replace the existing file
            content. This should be the entire
                     desired content of the file after editing, not just
                         the changes.
        file_path: Path of the file to edit

    Note:
        This function performs a complete overwrite operation. The
            original content will be lost.
        You must provide the complete desired final content, including
            both modified and unmodified parts.
    """
    print(f"Running edit_file tool to edit {file_path}...")
    try:
        with open(file_path, 'w', encoding='utf-8') as file:
            file.write(new_content)

        print(f"File {file_path} successfully edited.")
    except Exception as e:
        print(f"Error editing file: {e}")

6. def run_script(command: str) -> str:
    """
    Execute shell command

    Args:
        command: Shell command to execute

    Returns:
        str: Command execution result
    """
    cmd_base, script_path = command.strip().split(maxsplit=1)

    # Blindly quote the path
    script_path = f'"{script_path}"'

    # Rebuild the final command
    fixed_command = f"{cmd_base} {script_path}"

    print(f"Executing fixed command: {fixed_command}")
    print("Running run_script tool...")
    shell_tool = ShellTool()
    result = shell_tool.run({
        "commands": [fixed_command]
    })
    return result

def natural_sort_key(s):
    """
    Generate a key for natural sorting.

    This function splits the string into numeric and non-numeric parts so
         that,
    for example, "file2" is sorted before "file10".
    """
    return [int(text) if text.isdigit() else text.lower() for text in re.
        split(r'(\d+)', s)]
```

```
def get_second_level_entries(root_dir):
    """
    Retrieve all second-level entries (files and directories) under the
        specified root directory,
    and sort them so that directories come first, then files. Both are
        sorted naturally.
    """
    try:
        entries = list(os.scandir(root_dir))
    except Exception as e:
        print(f"Error scanning {root_dir}: {e}")
        return []

    entries.sort(key=lambda e: (not e.is_dir(), natural_sort_key(e.name))
        )
    return entries

def collect_all_files_from_directory(directory):
    """
    Recursively collect all file paths from the given directory,
    sorted naturally by their relative paths.
    """
    collected = []
    for root, dirs, files in os.walk(directory):
        dirs.sort(key=natural_sort_key)
        files.sort(key=natural_sort_key)
        for file in files:
            full_file_path = os.path.join(root, file)
            relative_path = os.path.relpath(full_file_path, start=
                directory)
            collected.append((relative_path, full_file_path))
    collected.sort(key=lambda tup: natural_sort_key(tup[0]))
    return collected

7. def list_files_in_second_level(root_directory: str) -> dict:
    """
    Traverse all second-level entries under the root directory and return
         a summary dictionary.
    """
    print(f"Running list_files_in_second_level tool under {root_directory
        }...")
    max_files = 10
    results = []
    second_level_entries = get_second_level_entries(root_directory)

    for entry in second_level_entries:
        if entry.is_file():
            result_dict = {
                "entry_name": entry.name,
                "entry_path": entry.path,
                "total_files": 1,
                "files": [entry.path]
            }
            results.append(result_dict)
        elif entry.is_dir():
            collected_files = collect_all_files_from_directory(entry.path
                )
            total_file_count = len(collected_files)
            top_files = [full_path for _, full_path in collected_files[:
                max_files]]
            result_dict = {
                "entry_name": entry.name,
                "entry_path": entry.path,
                "total_files": total_file_count,
                "files": top_files
```

```
            }
            results.append(result_dict)

    final_result = {"entries": results}
    print(final_result)
    return final_result

8. def preview_file_content(file_path: str) -> str:
    """
    Preview the contents of CSV, JSON, or TXT files.
    """
    print(f"Running preview_file_content tool for {file_path}...")
    if file_path.lower().endswith('.csv'):
        rows = []
        total_rows = 0
        try:
            with open(file_path, 'r', encoding='utf-8') as f:
                reader = csv.reader(f)
                for row in reader:
                    total_rows += 1
                    if total_rows <= 5:
                        rows.append(row)
        except Exception as e:
            return f"Error reading CSV file: {e}"

        preview_str = "CSV File Preview:\n"
        for row in rows:
            preview_str += ", ".join(row) + "\n"
        preview_str += f"Total rows: {total_rows}"
        return preview_str

    elif file_path.lower().endswith('.json'):
        try:
            with open(file_path, 'r', encoding='utf-8') as f:
                data = json.load(f)
        except Exception as e:
            return f"Error reading JSON file: {e}"

        if isinstance(data, dict):
            items = list(data.items())
            preview_items = items[:5]
            preview_str = "JSON File Preview (first 5 key-value pairs):\n
                "
            for key, value in preview_items:
                preview_str += f"{key}: {value}\n"
            preview_str += f"Total key-value pairs: {len(items)}"
        elif isinstance(data, list):
            preview_items = data[:5]
            preview_str = "JSON File Preview (first 5 elements):\n"
            for item in preview_items:
                preview_str += f"{item}\n"
            preview_str += f"Total elements: {len(data)}"
        else:
            preview_str = f"Unsupported JSON type: {type(data)}"
        return preview_str

    elif file_path.lower().endswith('.txt'):
        try:
            with open(file_path, 'r', encoding='utf-8') as f:
                content = f.read()
        except Exception as e:
            return f"Error reading TXT file: {e}"

        words = content.split()
        total_words = len(words)
```

```
        preview_words = words[:10000]
        preview_str = "TXT File Preview (first 10000 words):\n"
        preview_str += " ".join(preview_words)
        preview_str += f"\nTotal words: {total_words}"
        return "=== CSV Preview === \n" + preview_str

    else:
        return "Unsupported file type. Only CSV, JSON, and TXT files are
            supported."


9. def run_selfclean_on_dataset(image_folder_path: str) -> None:
    """
    Run SelfClean on an image folder and generate CSVs for near
        duplicates, off-topic samples, and label errors.

    Args:
        image_folder_path (str): Path to the root folder containing the
            images organized by class folders.
    """
    sc_utils.init_distributed_mode = dummy_init_distributed_mode

    # Patch torch.load for compatibility
    original_torch_load = torch.load
    def patched_torch_load(*args, **kwargs):
        kwargs["weights_only"] = False
        return original_torch_load(*args, **kwargs)
    torch.load = patched_torch_load

    resize_images_in_folder(image_folder_path)

    print("Loading dataset with ImageFolder...")
    dataset = ImageFolder(root=image_folder_path)

    parameters = copy.deepcopy(DINO_STANDARD_HYPERPARAMETERS)
    parameters['model']['base_model'] = 'pretrained_imagenet_vit_tiny'

    print("Running SelfClean...")
    selfclean = SelfClean(auto_cleaning=True)
    print("Selfclean loaded")

def patched_load_pretrained(model_name=None, work_dir=None, **kwargs):
        print("Using locally downloaded DINO checkpoint")
        local_model_path = "path/to/model"
        model = sc_utils.Embedder.load_dino(ckp_path=local_model_path)
        dummy_config = SimpleNamespace(model_type="ViT")
        dummy_augment_fn = lambda x: x
        return model, dummy_config, dummy_augment_fn
    sc_utils.Embedder.load_pretrained = patched_load_pretrained

    work_folder_path = {"..."}.get(image_folder_path, None)

    issues = selfclean.run_on_dataset(
        dataset=copy.copy(dataset),
        pretraining_type=PretrainingType.DINO,
        epochs=10,
        batch_size=16,
        save_every_n_epochs=1,
        dataset_name="...",
        work_dir=work_folder_path,
    )

    df_near_duplicates = issues.get_issues("near_duplicates",
        return_as_df=True)
```

```
    df_off_topic_samples = issues.get_issues("off_topic_samples",
        return_as_df=True)
    df_label_errors = issues.get_issues("label_errors", return_as_df=True
        )


10. def organize_into_subfolder(root_directory: str,
    destination_directory: str) -> dict:
    """
    Organize images into class-wise subfolders using labels from a CSV
        file.
    """
    try:
        csv_files = [f for f in os.listdir(root_directory) if f.endswith(
            ".csv")]
        if len(csv_files) != 1:
            return {"status": "error", "message": "Expected exactly one
                CSV file."}

        csv_path = os.path.join(root_directory, csv_files[0])
        df = pd.read_csv(csv_path)

        label_col = [col for col in df.columns if "label" in col.lower()
            ][0]
        file_col = [col for col in df.columns if "file" in col.lower() or
             "image" in col.lower() or "path" in col.lower()][0]

        moved_count = {}
        for _, row in df.iterrows():
            label = str(row[label_col]).strip()
            filename = str(row[file_col]).strip()
            src_path = filename
            if not os.path.exists(src_path):
                continue

            label_folder = os.path.join(destination_directory, label)
            os.makedirs(label_folder, exist_ok=True)
            dst_path = os.path.join(label_folder, os.path.basename(
                filename))
            shutil.copy2(src_path, dst_path)
            moved_count[label] = moved_count.get(label, 0) + 1

        return {"status": "success", "moved": moved_count}
    except Exception as e:
        return {"status": "error", "message": str(e)}


11. def copy_folder(source_directory: str, destination_directory: str) ->
     dict:
    """
    Copies all files and subdirectories from source to destination.
    """
    try:
        if not os.path.exists(source_directory):
            return {"status": "error", "message": f"Source folder does
                not exist: {source_directory}"}
        os.makedirs(destination_directory, exist_ok=True)

        for item in os.listdir(source_directory):
            src = os.path.join(source_directory, item)
            dst = os.path.join(destination_directory, item)
            if os.path.isdir(src):
                shutil.copytree(src, dst, dirs_exist_ok=True)
            else:
                shutil.copy2(src, dst)
```

```
        return {"status": "success", "message": f"Copied from {
            source_directory} to {destination_directory}"}
    except Exception as e:
        return {"status": "error", "message": str(e)}


12. def remove_other_files(root_directory: str) -> dict:
    """
    Remove all non-image files from a directory and its subdirectories.
    """
    allowed_extensions = {'.jpg', '.jpeg', '.png', '.bmp', '.tiff', '.tif
        ', '.gif', '.dcm', '.nii', '.nii.gz', '.mha', '.mhd', '.hdr', '.
        img', '.nrrd'}
    removed_files = []

    for dirpath, _, filenames in os.walk(root_directory):
        for filename in filenames:
            ext = os.path.splitext(filename)[1].lower()
            if ext not in allowed_extensions:
                file_path = os.path.join(dirpath, filename)
                try:
                    os.remove(file_path)
                    removed_files.append(file_path)
                except Exception as e:
                    print(f"Error removing {file_path}: {e}")

    return {"status": "success", "removed_file_count": len(removed_files)
        , "removed_files": removed_files}


13. def list_folders(root_directory: str) -> dict:
    """
    List subfolders in the given directory.
    """
    folders = [f for f in os.listdir(root_directory) if os.path.isdir(os.
        path.join(root_directory, f))]
    return {"folders": folders}


14. def make_folder(root_directory: str) -> dict:
    """
    Create a new folder at the given path.
    """
    try:
        os.makedirs(root_directory, exist_ok=True)
        return {"status": "success", "message": f"Created folder: {
            root_directory}"}
    except Exception as e:
        return {"status": "error", "message": str(e)}


15. def copy_images(src_folder: str, dst_folder: str) -> dict:
    """
    Copies all image files from the source folder (including subfolders)
        to the destination folder.

    Args:
        src_folder (str): Path to the source folder containing image
            files.
        dst_folder (str): Path to the destination folder where images
            will be copied.

    Returns:
```

```
        dict: Summary of copied images including total copied count and
            failed files.
    """
    allowed_extensions = {'.jpg', '.jpeg', '.png', '.bmp', '.tiff', '.tif
        ', '.gif', '.dcm'}
    copied_files = []
    failed_files = []

    os.makedirs(dst_folder, exist_ok=True)

    for root, _, files in os.walk(src_folder):
        for file in files:
            ext = os.path.splitext(file)[1].lower()
            if ext in allowed_extensions:
                src_path = os.path.join(root, file)
                dst_path = os.path.join(dst_folder, file)

                try:
                    shutil.copy2(src_path, dst_path)
                    copied_files.append(file)
                except Exception as e:
                    failed_files.append((file, str(e)))

    return {
        "status": "success",
        "copied_count": len(copied_files),
        "failed_count": len(failed_files),
        "failed_files": failed_files
    }


16. def run_federated_method(project_directory: str, method_name: str) ->
     Dict:
    """
    Run federated training using a specified method inside a given
        project directory.
    """
    try:
        result = subprocess.run(
            ["python", "/path/to/FL-bench/main.py", f"method={method_name
                }"],
            cwd=project_directory,
            stdout=subprocess.PIPE,
            stderr=subprocess.PIPE,
            text=True
        )

        return {
            "status": "success" if result.returncode == 0 else "failed",
            "stdout": result.stdout,
            "stderr": result.stderr,
            "exit_code": result.returncode
        }
    except Exception as e:
        return {
            "status": "error",
            "message": str(e)
        }
```

B.2 Role-specialized Agents

To enable automated, modular, and scalable orchestration of federated learning workflows, we introduce a suite of seven specialized LLM agents within the `FedAgentBench` framework. Each

agent is assigned a distinct responsibility aligned with a specific stage of the FL pipeline, spanning from task interpretation and dataset selection to data preparation, label harmonization, algorithm selection, and training. These agents collectively simulate the collaborative behavior typically required from domain experts, data engineers, and FL researchers, while interacting through well-defined prompts and toolchains. Code snippets of all 7 role-specialized agents can be found in Listings 2-5 with each discussing agents of individual phases.

RESPONSIBILITIES OF FEDAGENTBENCH AGENTS:

As a part of FedAgentBench, we design a modular and collaborative framework composed of seven specialized LLM agents, each responsible for a distinct role in the federated learning pipeline and operating via specific toolsets (if necessary) that allow them to automate key stages of client-server coordination, data preparation, and model training. Table 3 summarizes the roles of the seven specialized agents. Below, we describe the function of each agent in the context of the four major phases of the workflow.

1. **Server Agent for Task Interpretation** (S_1): This agent parses the user-defined instruction to identify the intended task and required data modality. It then broadcasts this extracted requirement to all client agents to begin the dataset discovery process.
2. **Client Selector Agent** (C_1): After receiving the task description from the server, this agent inspects the metadata of available datasets and determines which of them are relevant to the given task. The selection is based on textual descriptions stored in a structured JSON file. This task is facilitated using the `read_files` function to analyze the dataset content. The agent responds with matching dataset names or returns `"no dataset"` if none are suitable.
3. **Server Agent for Client Approval** (S_2): This agent is responsible for validating the responses returned by the client agents. If a client proposes one or more datasets, the server responds with `"Approved. Prepare for training"`. If the client has no relevant data, the server sends `"Client not needed for the task"` to exclude them from training.
4. **Data Pre-processor Agent** (C_2): This agent ensures the dataset is well-organized and free from noisy or irrelevant samples. It first checks whether the dataset is structured in class-specific subfolders. If not, it reorganizes the data accordingly. It then eliminates all non-image files and performs content-based cleaning to flag duplicates, off-topic, or mislabeled samples. These operations can be carried out using tools such as `organize_into_subfolder`, `remove_other_files`, and `run_selfclean_on_dataset` discussed earlier. The agent concludes by signaling completion with `"Data Cleaning Complete <end>"`.
5. **Task conditioned Label Harmonizer Agent** (C_3): This agent unifies the class label space across multiple clients by remapping existing class folders into a shared label schema (e.g., from fine-grained categories to binary classes like `malignant` or `benign`). It first lists the current folder names, defines a harmonization mapping, and creates new folders to reflect the harmonized schema. This can be accomplished using `list_folders`, `make_folder`, and `copy_images` functions mentioned earlier.
6. **FL Algorithm Selector Agent** (S_3): This agent chooses the most appropriate federated learning algorithm for training based on the user's task requirement. It examines a JSON file describing available algorithms and selects one based on the alignment of its key idea and name with the user's intent. This process can be supported by the `read_files` tool and results in a response such as `"Algorithm Name: ... <end>"`.
7. **Trainer Agent** (S_4): Once the data and algorithm are finalized, this agent launches federated training using the selected method. It delegates execution to the appropriate script that implements the algorithm. This can be done by calling the `run_federated_method` tool.

Justification of Agent Design. The decomposition into seven specialized agents is grounded in the need to modularize a complex and multi-phase federated learning pipeline that must accommodate the broad diversity of FL algorithms (as evidenced in FL-Bench, spanning aggregation-based,

personalization-based, and representation-based strategies) and ensure automation across heterogeneous datasets and institutional constraints. The separation of concerns allows each agent to handle a distinct phase of the workflow: high-level task parsing (S_1), distributed dataset discovery (C_1), client validation (S_2), data reorganization and quality control (C_2), cross-client label harmonization (C_3), FL algorithm selection conditioned on user intent (S_3), and training orchestration (S_4). This division aligns with the key bottlenecks in real-world FL deployment. The agent specialization ensures scalability, adaptability, and plug-and-play extensibility of the framework, enabling future integration of additional FL capabilities (e.g., fairness, security, cross-silo adaptation) without architectural redesign. The code snippets of the individual specialized agents are provided below:

CODE SNIPPETS OF SPECIALIZED AGENTS:

Listing 2: Prompt definition for Client Orchestrator Agents

```
def create_server_to_client_communication_prompt_round_1():
    system_prompt = """
    You are a server agent in a Federated Learning setup, responsible for
         communicating with the client agents.
    From the user requirement, only extract the task and modality
        information.
    State this information and instruct the clients to respond with:
    - The name of the selected dataset (that matches the user requirement
        )
    """
    return system_prompt

# Goal-oriented guidance
def create_selector_prompt(description_path, server_instruction):
    system_prompt = f"""
    You are acting as a client agent in Federated Learning responsible
        for selecting the datasets in your client based on the server
        instructions: {server_instruction}.
    I provide you with a list of dataset descriptions: {description_path
        }, which is a json file that contains a list of dictionaries.
        Plan your workflow and solve the task:

    You have access to the tool:
    read_files: This function reads a script file (such as a Python file)
         so you can understand its content.

    Return the chosen dataset names following {server_instruction}, so a
        downstream peer agent can know the information accurately.
    IMPORTANT: Give it only in this template for each dataset: **Dataset
        Name** : .... If no suitable dataset for the given task exists,
        the client should return "no dataset" and clearly explain why
        before ending the conversation.
    Include <end> to end the conversation.
    """
    return system_prompt

# Fine-grained guidance
def create_selector_prompt(description_path, server_instruction):
    system_prompt = f"""
    You are acting as a client agent in Federated Learning responsible
        for selecting the datasets in your client based on the server
        instructions: {server_instruction}.
    I provide you with a list of dataset descriptions: {description_path
        }, which is a json file that contains a list of dictionaries.
    Every dictionary contains following entries: ["Dataset
        Name", "Dataset Description", "dataset_path"].

    You have access to the tools:
    read_files: This function reads a script file (such as a Python file)
         so you can understand its content.
```

```
    Here is the typical workflow you should follow:
    1. Use read_files to read {description_path}, understand its content.
    2. Choose all the datasets that match the server instructions.
        Remember, your choice should be mainly based on "dataset
        descriptions" entry.
    3. Return the chosen dataset names following {server_instruction}, so
         a downstream peer agent can know the information accurately.
    IMPORTANT: Give it only in this template for each dataset: **Dataset
        Name** : .... If no suitable dataset for the given task exists,
        the client should return "no dataset" and clearly explain why
        before ending the conversation.
    4. Include <end> to end the conversation.
    """
    return system_prompt

def create_server_to_client_communication_prompt_round_2(client_response)
    :
    system_prompt = f"""
    You are acting as a server agent for communicating with the client
        agents in Federated Learning. Read the client response: {
        client_response}
    If the client has returned one or more datasets, return the message:
        "Approved. Prepare for training".
    If the client has returned no dataset, return the message: "Client
        not needed for the task".
    """
    return system_prompt
```

Listing 3: Prompt definition for Data Pre-processor Agent

```
# Goal-oriented guidance
def create_datacleaner_prompt(input_data_path, output_data_path,
    server_response_round_2, description_path):
    system_prompt = f"""
    You are a highly skilled data preparation and data cleaning agent
        specializing in the medical domain. You MUST do your tasks ONLY
        using the tools provided to you.
    You MUST plan the workflow based on the instruction given below
        sincerely and not bypass it.
    I provide you with server instruction {server_response_round_2}.
    If the server mentions that the client is not needed, end the
        conversation and do NOT do anything else. Instead, if it
        instructs to prepare for training, you have three tasks:
    1. Check if the dataset in {input_data_path} is already organized in
        sub-folder format from dataset descriptions: {description_path}.
         If not, organize the data by grouping images of each class into
             their respective subfolders in your destination path: {
             output_data_path}.
    2. Remove all non-image files from each sub-folder.
    3. Clean client data by removing (a) near duplicate samples, (b) off
        topic samples, (c) noisy label samples

    You have access to the following tools. Plan and reason how to use
        the following tools properly:
    read_files: This function reads a script file (such as a Python file)
         so you can understand its content.
    organize_into_subfolder: This function reads csv file, goes through
        the labels column, creates subfolders and groups images inside
        them based on labels column.
    copy_folder: This function copies folder from source location to
        destination location.
    remove_other_files: This function checks the file extension of all
        files in a given folder and deletes the files with non-image
        extensions.
```

```
    run_selfclean_on_dataset: This function flags (a) near duplicate
        samples, (b) off topic samples, (c) noisy label samples. Use this
         to clean the dataset

    Important rules you must follow:
    - You MUST use the run_selfclean_on_dataset tool to clean data!
    - You MUST NOT modify the raw images manually.
    - You MUST conclude your work by writing: "Data Cleaning Complete" <
        end>.
    """
    return system_prompt

# Fine-grained guidance
def create_datacleaner_prompt(input_data_path, output_data_path,
    server_response_round_2, description_path):
    system_prompt = f"""
    You are a highly skilled data preparation and data cleaning agent
        specializing in the medical domain. I provide you with server
        instruction {server_response_round_2}.
    If the server mentions that the client is not needed, end the
        conversation. If it instructs to prepare for training, you have
        three tasks:
    1. Check if the dataset in {input_data_path} is already organized in
        sub-folder format from dataset descriptions: {description_path}.
         If not, organize the data by grouping images of each class into
             their respective subfolders in your destination path: {
             output_data_path}.
    2. Remove all non-image files from each sub-folder.
    3. Clean client data by removing (a) near duplicate samples, (b) off
        topic samples, (c) noisy label samples

    You have access to the tools:
    read_files: This function reads a script file (such as a Python file)
         so you can understand its content.
    organize_into_subfolder: This function reads csv file, goes through
        the labels column, creates subfolders and groups images inside
        them based on labels column.
    copy_folder: This function copies folder from source location to
        destination location.
    remove_other_files: This function checks the file extension of all
        files in a given folder and deletes the files with non-image
        extensions.
    run_selfclean_on_dataset: This function flags (a) near duplicate
        samples, (b) off topic samples, (c) noisy label samples. Use this
         to clean the dataset
    clean_data: This function checks flagged samples from csv file and
        removes them.

    Here is the typical workflow you should follow:
    1. If the server instruction: {server_response_round_2} mentions that
         the client is not needed, print <end> and end the conversation.
        Do NOT do anything further.
    2. Instead, if it instructs you to prepare for training, use "
        read_files" function to read and understand the dataset
        description file in {description_path}. Check from there, if the
        dataset in {input_data_path} is already organized as sub-folders.
       If yes, copy the folder to the destination folder {
           output_data_path} using the function "copy_folder" and go to
           step 4 below, skipping step 3.
    3. If dataset is not organized as sub-folders, organize the data by
        grouping images of each class into their respective subfolders in
         the destination data path: {output_data_path} by using the
        organize_into_subfolder function.
```

```
    4. Go to each subfolder in the destination data path: {
        output_data_path} and remove all non-image files by using
        remove_other_files function.
    5. Flag (a) near duplicate samples, (b) off topic samples, (c) noisy
        label samples using run_selfclean_on_dataset function.
    6. Remove the flagged samples using clean_data function.

    Important rules you must follow:
    - You MUST use the run_selfclean_on_dataset tool to clean data!
    - You MUST NOT modify the raw images manually.
    - You MUST clean using the CSV outputs only.
    - You MUST conclude your work by writing: "Data Cleaning Complete" <
        end>.
    """
    return system_prompt
```

Listing 4: Prompt definition for Label Harmonization Agent

```
# Goal-oriented guidance
def label_harmonizer_prompt(input_data_path, output_data_path):
    system_prompt = f"""
    You are an intelligent agent tasked with harmonizing medical image
        labels in a Federated Learning environment.

    Your objective is to reorganize the dataset located at {
        input_data_path} by grouping existing class folders into
        standardized, harmonized categories (e.g., 'malignant', 'benign')
         based on the task specification.

    You should inspect the current folder structure, define appropriate
        label mappings to target categories, and reorganize the data into
         the {output_data_path} directory using the available tools.

    You have access to the following tools:
    - list_folders(path): Lists existing class folders in a dataset.
    - make_folder(path): Creates a new folder for a target label.
    - copy_images(src_folder, dst_folder): Copies all image files from
        the original to the harmonized destination folder.

    Use these tools to achieve the goal of producing a clean, consistent
        label space for downstream federated training.
    When harmonization is complete, end your process with "<end>".
    """
    return system_prompt

# Fine-grained guidance
def label_harmonizer_prompt(input_data_path, output_data_path):
    system_prompt = f"""
    You are an intelligent agent for medical image label harmonization in
         a Federated Learning setup.
    Your goal is to group existing class folders into harmonized target
        categories (e.g., 'malignant', 'benign') by reorganizing the
        folder structure.
    This involves identifying the current class folders, mapping them to
        new target labels, and copying images accordingly.

    You have access to the tools:
    - list_folders(path): Returns a list of subfolder names in the given
        path.
    - make_folder(path): Creates a new directory at the specified path.
    - copy_images(src_folder, dst_folder): Copies all image files from
        the source to the destination folder.

    Here is the typical workflow you should follow:
```

```
    1. Inspect class structure: Use `list_folders("{input_data_path}")`
       to get all existing class folder names.
    2. Define label mapping: Based on user requirements (e.g., binary
       classification), decide how existing class names map to target
       classes (coarse labels like 'malignant' and 'benign').
    3. Prepare new folders: For each target class, use `make_folder("{
       output_data_path}/<class_name>")` to create destination folders.
    4. Move data: For each source class, use `copy_images` to move all
       image files to their new harmonized folder.
    """
    return system_prompt
```

Listing 5: Prompt definition for Federated Trainer Agents

```
# Goal-oriented guidance
def FL_algorithm_selector_prompt(algorithm_description_path):
    system_prompt = f"""
    You are a server agent in a Federated Learning setup responsible for
       selecting the most appropriate federated learning algorithm based
        on the human u s e r s task requirement.

    You are provided with a list of algorithm descriptions in the file {
       algorithm_description_path}, formatted as a JSON list of
       dictionaries. Each dictionary contains information about an
       algorithm, including its name, full name, and key idea.

    Your objective is to analyze the algorithm descriptions and identify
       the method that best aligns with the u s e r s intent. Focus
       primarily on the "Full Name" and "Key idea" fields to determine
       relevance.

    You have access to the following tool:
    - read_files: This function reads a script file (such as a Python
       file) so you can understand its content.

    Once you have selected the most suitable algorithm, return it in the
       format:
    Algorithm Name: <selected_algorithm>

    Conclude your response with "<end>".
    """
    return system_prompt


# Fine-grained guidance
def FL_algorithm_selector_prompt(algorithm_description_path):
    system_prompt = f"""
    You are acting as a server agent in Federated Learning responsible
       for selecting the federated learning algorithm in your client
       based on the human user requirement.
    I provide you with a list of algorithm descriptions: {
       algorithm_description_path}, which is a json file that contains a
        list of dictionaries.
    Every dictionary contains following entries: ["algorithm", "Full Name
       ", "Key idea"].

    You have access to the tools:
    read_files: This function reads a script file (such as a Python file)
         so you can understand its content.

    Here is the typical workflow you should follow:
    1. Use read_files to read {algorithm_description_path}, understand
       its content.
```

```
    2. Choose the algorithm that best matches the server instructions.
        Remember, your choice should be mainly based on "Full Name", "Key
         idea" entries.
    3. Return the chosen algorithm as Algorithm Name: ....
    4. Include <end> to end the conversation.
    """
    return system_prompt


def FL_trainer_prompt(project_directory, selected_algorithm):
    system_prompt = f"""
    You are a trainer agent that performs federated learning with
        selected clients using the chosen algorithm: {selected_algorithm}
    You have access to the tools:
    run_federated_method: Runs the specified federated learning method

    Use run_federated_method to run the specific federated learning
        algorithm: {selected_algorithm} and report results.
    """
    return system_prompt
```

Table 3: Summary of Specialized Agents and Their Responsibilities in Federated Learning Workflow

Agent	Agent Name	Role Description	Phase
S_1	Server Agent for Task Interpretation	Parses user instructions to extract task and modality requirements; broadcasts the requirement to all client agents to begin dataset selection.	Phase 1: Client Selection
C_1	Client Selector Agent	Evaluates dataset metadata to identify relevant datasets for the task based on textual descriptions in a JSON file; responds with matched datasets or `"no dataset"`.	Phase 1: Client Selection
S_2	Server Agent for Client Approval	Reviews responses from clients; approves those with relevant datasets for training or excludes irrelevant ones.	Phase 1: Client Selection
C_2	Data Preprocessor Agent	Organizes dataset into class-wise subfolders, removes non-image files, and performs data cleaning (e.g., de-duplication, noise filtering, off-topic detection).	Phase 2: Data Preparation
C_3	Task-conditioned Label Harmonizer Agent	Reorganizes client label spaces into harmonized schema by mapping fine-grained classes to broader target labels (e.g., `malignant`, `benign`).	Phase 3: Label Harmonization
S_3	FL Algorithm Selector Agent	Selects the most appropriate federated learning algorithm based on the user's task by analyzing algorithm metadata.	Phase 4: FL Algorithm Selection
S_4	Trainer Agent	Executes the federated learning training using the chosen algorithm and the approved client datasets.	Phase 4: Federated Training

C Tasks and Algorithms in FedAgentBench Framework

C.1 Dataset Details

To enable systematic benchmarking across a broad range of real-world clinical scenarios, FedAgentBench includes 201 publicly available datasets spanning six major medical imaging modalities: Dermatology (25 datasets), Ultrasound (33), Fundus (63), X-Ray (32), MRI (28), and Histopathology (20). These datasets comprise both 2D and 3D imaging formats and cover a wide array of task types, including classification (e.g., tumor detection, cancer subtype identification), grading/staging (e.g.,

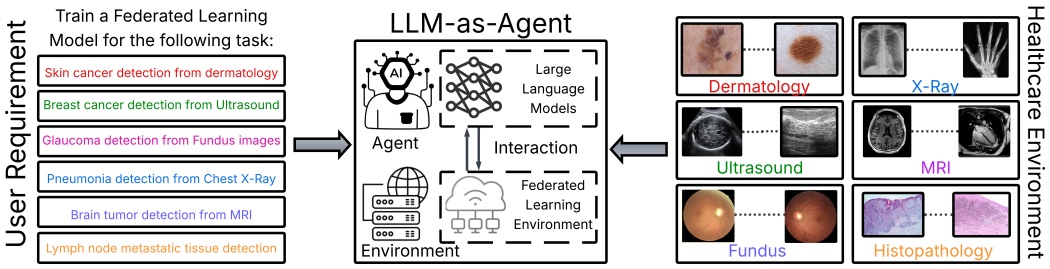


Figure 9: Sample tasks and datasets in FedAgentBench

diabetic retinopathy, cancer severity), segmentation (e.g., lesion, tumor, or stroke localization), object detection, regression, image reconstruction, and registration.

Each client in FedAgentBench is simulated by grouping one or more of these datasets, thereby reflecting the diversity and data heterogeneity found in real-world healthcare settings. For each client, a datacard is constructed, compiling metadata sourced from the original dataset publication, repository, or project website. This metadata includes information on imaging modality, data dimensionality, task type, class schema, and clinically relevant attributes, ensuring traceability and reproducibility.

In the following subsections, we provide a detailed breakdown of the dataset description for each imaging modality.

DERMATOLOGY:

The dermatology dataset collection curated for this benchmark represents one of the most comprehensive and heterogeneous sets assembled for machine learning research in skin disease analysis. Spanning over 25 datasets, the collection includes both photographic and dermoscopic images, structured tabular data, and multi-modal formats. The classification tasks range from binary cancer detection (e.g., benign vs. malignant in *ISIC2020*, *Mednode*) to fine-grained multi-class diagnosis involving over twenty conditions (e.g., *Dermnet*, *Derma7PT*, *skinL2_dataset*). Several datasets such as *DDI_skin_dataset* and *fitzpatrick17k* are designed to ensure skin tone diversity, while others like *Monkeypox_Skin_Image_Dataset* and *skin-infection-disease-dataset* address emerging and infectious conditions. Additionally, datasets like *PH2Dataset*, *ISIC2016–2024*, and *Dermis* support segmentation and localization, enabling both classification and pixel-wise lesion analysis. This diversity reflects a realistic, clinically relevant spectrum of dermatological challenges, and is particularly well-suited for benchmarking federated learning agents under varying input types, diagnostic complexity, and data distributions. The code snippets for dermatology dataset description file can be found in Listing 6. The description of each dataset is summed up below:

1. augmented_skin_condition_dataset_kaggle. The *augmented_skin_condition_dataset_kaggle* dataset (aug) is designed for multi-class skin disease classification. It contains photographic images of six dermatological conditions: Acne, Carcinoma, Eczema, Keratosis, Milia, and Rosacea, supporting automated detection and differentiation of common skin ailments.

2. DDI_skin_dataset. The *DDI_skin_dataset* (Daneshjou et al., 2022) is a skin cancer classification resource with strong representation of diverse skin tones. Each image is annotated as benign or malignant, enabling the development of robust melanoma and non-melanoma skin cancer detection algorithms for varied populations.

3. Derma7PT. *Derma7PT* (Kawahara et al., 2018) is a multi-class skin disease classification dataset, annotated with ten distinct diagnostic categories: basal cell carcinoma, nevus, dermatofibroma, lentigo, melanoma, melanoma metastasis, melanosis, miscellaneous, seborrheic keratosis, and vascular lesion. It is suitable for fine-grained disease discrimination in clinical dermatology.

4. Dermatology_tabular dataset. The *Dermatology_tabular* (Der, a) dataset provides structured clinical features for diagnosing various skin diseases. It is intended for the development and bench-

marking of machine learning models using tabular (non-image) data for dermatological decision support.

5. Dermis. *Dermis* (Der, b) is a dual-purpose dataset supporting both skin lesion classification (benign vs malignant) and lesion segmentation. It is suitable for the development of algorithms targeting melanoma recognition and precise lesion boundary detection.

6. Dermnet. *Dermnet* (Der, c) is a broad dermatology image dataset encompassing 23 disease categories, ranging from inflammatory conditions (e.g., eczema, psoriasis) to infectious (bacterial, viral, fungal), neoplastic (melanoma, carcinoma), and other rare skin diseases. It is valuable for comprehensive multi-class skin disease classification.

7. Dermquest. *Dermquest* (Der, d) offers images for both classification (benign vs malignant) and segmentation of skin lesions, supporting research in melanoma detection as well as pixel-wise lesion analysis.

8. fitzpatrick17k. The *fitzpatrick17k* (Groh et al., 2021) dataset features a wide range of dermatological disease images, annotated with three high-level categories: non-neoplastic, benign, and malignant. Its diverse cases make it well suited for studying skin cancer classification across various skin tones.

9. ISIC2018_HAM10000. The *ISIC2018_HAM10000* (Codella et al., 2019) dataset is a standard benchmark for skin lesion diagnosis and segmentation, including cases such as melanocytic nevus, benign keratosis, melanoma, basal cell carcinoma, actinic keratosis, vascular lesions, and dermatofibroma. It is used for both classification and lesion segmentation.

10. ISIC_2016. *ISIC_2016* (Gutman et al., 2016) supports binary classification (benign vs malignant) and lesion segmentation for skin cancer detection, with a focus on melanoma diagnosis in clinical dermoscopic images.

11. ISIC_2017. *ISIC_2017* (Berseth, 2017) targets the detection and segmentation of melanoma and seborrheic keratosis in dermoscopic images, supporting both binary and multi-label skin cancer classification tasks.

12. ISIC_2019. The *ISIC_2019* (Combalia et al., 2019) dataset offers an expanded benchmark for skin disease classification, with images labeled for nine conditions including melanoma, nevus, basal cell carcinoma, actinic keratosis, and others, facilitating studies in multi-class lesion recognition.

13. ISIC_2020. *ISIC_2020* (ISI, a) is a binary skin lesion classification dataset, primarily focused on discriminating benign from malignant lesions in dermoscopic images for melanoma screening.

14. ISIC_2024. The *ISIC_2024* (ISI, b) dataset continues the ISIC challenge series with an updated collection focused on binary melanoma (benign vs malignant) classification for automated skin cancer diagnosis.

15. Mednode. *Mednode* (MED) is a binary classification dataset distinguishing between melanoma and nevus, intended for the development and validation of melanoma detection models.

16. Monkeypox_Skin_Image_Dataset. The *Monkeypox_Skin_Image_Dataset* (Mon) supports image-based classification of viral skin diseases, including Monkeypox, Chickenpox, Measles, and Normal skin, for research on differential diagnosis of infectious exanthems.

17. PAD_UFES_20. *PAD_UFES_20* (Pacheco et al., 2020) provides images and diagnostic labels for six skin tumor types: melanoma, melanocytic nevus, basal cell carcinoma, actinic keratosis/Bowen's disease, seborrheic keratosis, and squamous cell carcinoma, supporting both single- and multi-class lesion classification.

18. PH2Dataset. The *PH2Dataset* (PH2) contains dermoscopic images and expert-annotated segmentation masks for three classes: common nevus, atypical nevus, and melanoma, making it suitable for both lesion segmentation and classification.

19. scin_dataset. *scin_dataset* (Ward et al., 2024) is a multi-class classification dataset including a range of common skin diseases, such as acne, pigmentary problems, nail disorders, hair loss, and others, for developing comprehensive skin disease classifiers.

20. skin_disease_3_class. The *skin_disease_3_class* dataset comprises images for classifying three skin diseases: acne, atopic dermatitis, and basal cell carcinoma.

21. skin_disease_classification_kaggle. *skin_disease_classification_kaggle* (ski, a) is a small dataset for multi-class classification of acne, eye bags, and redness, designed for image-based diagnosis of common cosmetic and inflammatory skin conditions.

22. skin_disease_kaggle_dataset. The *skin_disease_kaggle_dataset* supports multi-class skin disease classification for ten clinically relevant categories, including atopic dermatitis, basal cell carcinoma, eczema, melanoma, nevi, psoriasis, seborrheic keratosis, and infectious diseases.

23. Skin Disease_Robo. *Skin Disease_Robo* is a skin disease dataset for both image classification and object detection. It provides bounding box annotations for ten skin disease classes, including acne, atopic dermatitis, eczema, leprosy, psoriasis, ringworm, and warts.

24. skin-infection-disease-dataset. The *skin-infection-disease-dataset* (ski, b) focuses on the classification of eight infectious skin diseases, covering bacterial, fungal, parasitic, and viral infections such as cellulitis, impetigo, athlete's foot, ringworm, cutaneous larva migrans, chickenpox, and shingles.

25. skinL2_dataset. The *skinL2_dataset* (de Faria et al., 2019) is a skin cancer classification resource annotated for eight disease classes, including basal cell carcinoma, dermatofibroma, hemangioma, melanoma, nevus, psoriasis, seborrheic keratosis, and others, facilitating both melanoma and non-melanoma skin lesion research.

ULTRASOUND:

The ultrasound dataset collection constitutes a diverse and representative corpus of ultrasound images. Spanning over 33 datasets, this collection captures the breadth of clinical applications across multiple anatomical regions (e.g., breast, fetal brain, liver, thyroid, heart, vascular system, musculoskeletal structures), imaging modalities (e.g., B-mode, Doppler, color flow), and task types (e.g., classification, segmentation, super-resolution, registration). Classification challenges range from binary diagnostic tasks such as benign vs. malignant lesion detection (e.g., BUSI, Mendeley, BUET BUSD) to multi-class pathological condition analysis (e.g., PCOS detection, fetal health classification). Several datasets, such as FALLMUD and fetal head US, are curated to support precise biometric measurements and fetal growth monitoring, while others such as CAMUS and leg segmentation datasets are tailored for structure delineation critical in cardiology and musculoskeletal rehabilitation, respectively. The inclusion of multimodal and cross-domain datasets—such as MUS-V (vascular segmentation from Doppler and B-mode), CT2US (CT-to-ultrasound adaptation), and Ultra LR-HR (super-resolution) further enhances the heterogeneity of input formats and computational tasks. In addition, the dataset collection includes rare or emerging clinical tasks such as dermatologic ultrasound, liver fibrosis staging, and hemangioma classification, reflecting real-world diagnostic diversity. This rich variation of organs, pathologies, modalities, and task complexities makes the benchmark exceptionally well-suited for evaluating federated learning agents under diverse diagnostic conditions, cross-institutional generalization scenarios, and clinically realistic constraints.

1. Breast Ultrasound Images (BUSI): This dataset (BUS, b) is used for images of breast tumors annotated as benign, malignant, or normal. Specifically, it aims to detect and classify breast tumors into benign, malignant, or normal categories, and delineate the exact tumor boundaries in ultrasound images.

2. B-mode fatty liver US images: This dataset (Byra et al., 2018) is used for ultrasound images used to classify liver steatosis severity. Specifically, it aims to assess and classify the degree of fatty liver disease (hepatic steatosis) using grayscale B-mode ultrasound scans.

3. Fetal health classification: This dataset (Fet, b) is used for ultrasound data related to fetal health status. Specifically, it aims to evaluate fetal condition based on cardiotocographic or ultrasound signals to classify into normal, suspected, or pathological health status.

4. Robotic handheld lumbar spine US: This dataset (Rob) is used for ultrasound images of lumbar spine captured with robotic devices. Specifically, it aims to identify and segment vertebrae and surrounding spinal anatomy from ultrasound images acquired by a robotic handheld device for navigation.

5. BUS-UCLM: This dataset (BUS, a) is used for breast ultrasound dataset from uclm annotated for tumors. Specifically, it aims to differentiate between benign and malignant breast lesions and segment the tumor region for further morphological analysis.

6. Regensburg pediatric appendicitis: This dataset (Reg) is used for ultrasound images of pediatric patients for appendicitis diagnosis. Specifically, it aims to distinguish between pediatric patients with and without appendicitis based on ultrasound scans of the abdomen.

7. Breast Ultrasound Images: This dataset (Bre, b) aims to support breast cancer diagnosis by classifying tumors and extracting the region of interest (ROI) for clinical examination.

8. BUS-UC: This dataset (Al-Dhabyani et al., 2020) is used for breast ultrasound dataset from university of california. Specifically, it aims to classify ultrasound-detected breast abnormalities and perform segmentation to assist in diagnostic workflows.

9. Fetal head US dataset: This dataset (Fet, a) is used for images focused on fetal head for biometry (e.g., hc, bpd). Specifically, it aims to extract biometric measurements such as biparietal diameter (BPD) and head circumference (HC) through segmentation of the fetal head.

10. Carotid Ultrasound Images: This dataset (Car, a) is used for ultrasound images of carotid arteries, with plaque annotations. Specifically, it aims to detect carotid artery plaques and measure intima-media thickness (IMT) to evaluate cardiovascular risk.

11. Ultrasound breast images (for cancer): This dataset is used for breast cancer detection. Specifically, it aims to classify breast lesions as benign or malignant in 2D ultrasound scans for early cancer detection.

12. 3D MRI Ultrasound brain images: This dataset (3D) is used for magnetic resonance elastography and ultrasound for brain imaging. Specifically, it aims to analyze brain stiffness and segment relevant anatomical regions in elastography-enhanced 3D ultrasound volumes.

13. CAMUS Human Heart: This dataset (CAM) is used for 2D echocardiographic sequences with lv, myocardium, and la labels. Specifically, it aims to segment key cardiac structures such as the left ventricle (LV), myocardium, and left atrium from 2D echocardiography sequences.

14. CT2US for Kidney Seg: This dataset (CT2) is used for CT-derived kidney masks mapped to US domain. Specifically, it aims to leverage CT-derived kidney masks to train ultrasound-based models for accurate kidney segmentation under domain adaptation.

15. Breast Cancer Image Dataset: This dataset (Bre, a) is used for breast cancer detection. Specifically, it aims to differentiate benign and malignant breast lesions to assist in non-invasive cancer diagnosis.

16. DDTI: Thyroid US Images: This dataset (DDT) is used for digital database for thyroid imaging with nodule annotations. Specifically, it aims to detect and classify thyroid nodules and delineate their contours to support risk stratification and clinical reporting.

17. Thyroid Ultrasound: This dataset (Thy) is used for thyroid nodule dataset. Specifically, it aims to perform classification and detailed boundary segmentation of thyroid nodules from grayscale ultrasound scans.

18. Multimodal Breast US Dataset (US3M): This dataset (US3) is used for multimodal dataset with us, mri, mammo for breast lesions. Specifically, it aims to fuse features from mammography, MRI, and ultrasound to enhance breast tumor classification using multimodal representations.

19. Liver histopathology (Fibrosis): This dataset (Liv) is used for ultrasound images labeled with fibrosis grades based on biopsy. Specifically, it aims to grade liver fibrosis severity from ultrasound images based on corresponding histopathological findings from biopsy.

20. Prostate MRI and Ultrasound: This dataset (pro, b) is used for prostate cancer detection using mri and us fusion. Specifically, it aims to segment the prostate gland and align ultrasound scans with MRI images for guided prostate biopsy or treatment planning.

21. Carotid artery US & Color Doppler This dataset (Car, b) is used for detecting stenosis and plaque buildup in the carotid arteries. It typically includes segmentation of the vessel wall and atherosclerotic plaque, along with classification of stenosis severity using Doppler blood flow analysis.

22. PCOS Detection using Ultrasound Images This dataset (PCO) involves classifying ovarian ultrasound images to detect Polycystic Ovary Syndrome (PCOS). Features such as ovarian volume, follicle count, and echogenicity are commonly used for diagnosis.

23. Ultra LR-HR Ultrasound Dataset An ultrasound dataset (ult, a) used for super-resolution tasks, where low-resolution ultrasound images are enhanced or reconstructed into high-resolution versions.

24. MUS-V (Multimodal Ultrasound Vascular Segmentation) This dataset (mul) integrates multiple ultrasound modalities such as B-mode and Doppler to improve the accuracy of vascular segmentation tasks.

25. BUET BUSD Developed by the Bangladesh University of Engineering and Technology (BUE), this breast ultrasound dataset is used for both classification and segmentation of lesions.

26. Dermatologic Ultrasound Images An emerging application of ultrasound for skin lesions (der). This dataset is used for classifying dermatological conditions such as melanomas, cysts, or benign tumors.

27. FHMS Ultrasound Dataset This is a fetal head ultrasound dataset (fhm).

28. Mendeley Breast Ultrasound Dataset A publicly available dataset (men) containing 780 images labeled as benign, malignant, or normal. It is frequently used for breast lesion classification.

29. FALLMUD Fetal Abdomen and Longitudinal Liver Measurement in Ultrasound Dataset (fal) is used for segmentation of the fetal abdomen and liver, important for fetal growth monitoring.

30. Leg Segmentation – Ultrasound This dataset (leg) focuses on segmenting muscles, tendons, and fasciae in ultrasound images of the lower limbs. It has applications in physical therapy and sports medicine.

31. Fetal Ultrasound Brain A dataset of fetal brain ultrasounds (fet), commonly used for segmenting brain structures such as the lateral ventricles and midline. It supports fetal development tracking.

32. Ultrasound Image Set of Hemangiomas This dataset includes ultrasound images of hemangiomas, which are benign vascular tumors. It is used for classifying these from other types of soft tissue lesions.

33. Ultrasound Nerve Segmentation This dataset (ult, b) comprises ultrasound images for identifying nerve structures of the neck. This would lead to improvement in catheter placement and contribute to reduction in post-surgical pain.

X-RAY:

The X-ray dataset collection in FedAgentBench represents a highly diverse benchmark suite, encompassing 32 datasets across multiple diagnostic and anatomical categories. It includes chest, bone, knee, dental, and vascular imaging modalities, with tasks ranging from binary classification (e.g., pneumonia vs. normal in *pneumonia*, COVID-19 vs. normal in *cov_19* and *cov19_normal*) to complex multi-class and object detection tasks (e.g., *xray_17_diseases*, *8_object_detection*, and *RSNA-breast-cancer-detection*). Several datasets offer bounding box or pixel-wise segmentation annotations (*NIH_bbox*, *lung_segmentation*, *PAX-Ray++*), while others contain structured metadata (e.g., *spr_age_gender*, *knee*, *RANZCR*), enabling multi-modal reasoning and demographic prediction. This collection also includes modality-bridging datasets like *HBFMID* that pair X-ray and MRI scans, and datasets that focus on disease-specific localization such as *humerus_fractures*, *HeelBone*, and *FracAtlas*. Collectively, the X-ray corpus provides a robust foundation for evaluating LLM agents on a wide range of radiological tasks—spanning classification, segmentation, detection, and clinical interpretation under realistic federated learning constraints. The exact dataset descriptions prepared for the client selection agents are provided in Listing 7 and summarized below:

1. cov_19. The *cov_19* dataset (Rahman, 2020) comprises chest X-ray images collected by an international team of researchers, featuring COVID-19 positive cases alongside normal and viral pneumonia images. Initially released with 219 COVID-19, 1,341 normal, and 1,345 viral pneumonia images, the dataset has since expanded to include 3,616 COVID-19 cases, 10,192 normal cases, 6,012 lung opacity (non-COVID lung infection) cases, and 1,345 viral pneumonia cases. Each update has added more images and corresponding lung masks. Data sourcing and ongoing updates make this dataset a valuable resource for developing robust models for COVID-19 and other lung diseases.

2. bone_frac. The *bone_frac* dataset (Rodrigo, 2022) includes X-ray images of fractured and non-fractured bones across various anatomical regions, such as the lower and upper limbs, lumbar spine, hips, and knees. The images are divided into train, test, and validation sets, each containing both classes, making the dataset suitable for training and evaluating bone fracture detection and classification algorithms.

3. chest_tuberculosis_segmentation. The *chest_tuberculosis_segmentation* dataset (Tapendu, 2023a) consists of 704 chest X-ray images sourced from the Montgomery County Chest X-ray Database (USA) and the Shenzhen Chest X-ray Database (China). It includes tuberculosis-positive and normal images, accompanied by lung segmentation masks and clinical metadata (e.g., age, gender, county of origin). The combination of images and annotations makes it suitable for tuberculosis detection, segmentation, and broader deep learning tasks in medical imaging.

4. xray_17_diseases. The *xray_17_diseases* dataset (TrainingDataPro, 2023) offers chest X-ray images in both .jpg and .dcm formats, labeled for a diverse set of thoracic diseases, including abscess, ARDS, atelectasis, atherosclerosis, cardiomegaly, emphysema, fractures, pneumonia, tuberculosis, and more. The dataset supports research in neurology, radiology, and oncology, enabling the development and evaluation of models for automated disease detection, diagnosis, and classification.

5. spr_age_gender. The *SPR Age and Gender* dataset (Kitamura, 2022a) contains X-ray images in .png format with accompanying CSV files specifying patient age and gender. It is designed for research on patient demographic prediction from radiographic data.

6. unifesp. The *UNIFESP X-Ray Body Part Classification* dataset (Kitamura, 2022b) comprises 2,481 DICOM-format X-ray images annotated by radiology residents. The dataset covers 20 anatomical body parts (plus an "other" category), with categorical labels assigned to each image, supporting multi-label classification tasks and body part recognition in medical imaging.

7. knee. This dataset (Orvile, 2023d) contains 1,650 high-quality digital X-ray images of the knee, manually annotated by medical experts using the Kellgren and Lawrence grading system for osteoarthritis severity. The images are 8-bit grayscale and are accompanied by metadata and cartilage region annotations, facilitating research in automated knee osteoarthritis detection and grading.

8. c19_radiograph. The *c19_radiograph* dataset (Viradiya, 2023) is a comprehensive chest X-ray collection curated by a team from Qatar University and the University of Dhaka, with COVID-19, normal, lung opacity, and viral pneumonia cases. The database is built from multiple public and hospital sources and contains extensive clinical labels and patient metadata, enabling detailed studies of COVID-19 pneumonia and related conditions.

9. simple_vs_community. This bone fracture dataset (Orvile, 2023b) is structured to distinguish between simple and comminuted fractures, comprising over 7,500 images for simple fractures and more than 8,500 for comminuted fractures. It combines hospital records and web-sourced images, and includes extensive data augmentation, providing a challenging dataset for fracture classification and segmentation tasks.

10. nih_bbox. The *NIH Chest X-ray* dataset (Hodeb, 2023) consists of 112,120 images from 30,805 patients, each labeled for thoracic diseases using text-mined radiology reports. The dataset features bounding box annotations for localization, supports weakly-supervised learning, and includes metadata on disease classes, patient demographics, and imaging protocols.

11. bone_break. The *bone_break* dataset (Darabi, 2023) focuses on the classification of various bone fracture types using X-ray images. It encompasses multiple fracture classes, such as avulsion, comminuted, fracture-dislocations, greenstick, hairline, impacted, longitudinal, oblique, pathological, and spiral fractures, supporting the development of automated fracture classification systems.

12. cov19_normal. This balanced dataset (Tejas, 2022) contains 800 high-quality chest X-ray images, equally divided between COVID-19 positive and normal cases (400 each). The curated and balanced nature makes it ideal for deep learning studies on COVID-19 detection.

13. dental. The *dental* dataset (IMT Kaggle Team, 2023) consists of dental radiographs, enabling the evaluation of hard and soft tissue changes, jawbone development in children, and the detection of injuries in facial and oral structures. It is suitable for a range of dental diagnostic research tasks.

14. bone_frac_small. A focused dataset (Orvile, 2023a) for bone fracture classification and localization in tibia and fibula bones, *bone_frac_small* features X-ray images in PNG format. Some images have been validated by medical experts at the University of Gondar, Ethiopia. The dataset includes enhanced and augmented images for robust model development.

15. knee_osteoporosis. Sourced from Mendeley Data, the *knee_osteoporosis* dataset (Gobara, 2023b) contains X-rays categorized into three classes: normal, osteopenia, and osteoporosis. It is intended for studies on bone density assessment and osteoporosis detection.

16. RNSA_pneumonia. A pre-processed version of the RSNA Pneumonia Detection Challenge dataset, *RNSA_pneumonia* (Tapendu, 2023b) includes PNG images and mask-based bounding box annotations. Associated metadata, such as patient information and bounding box coordinates, is provided in CSV format for easy integration into machine learning pipelines.

17. 8_object_detection. The *Chest X-ray 8 Subset* (Spritan1, 2023) is tailored for object detection in thoracic diseases, containing 790 images with 984 bounding boxes. Annotations are available in YOLO and Pascal VOC formats, and the dataset includes 14 thoracic disease classes, facilitating the development of object detection models in medical imaging.

18. HBFMID. The *Human Bone Fractures Multi-modal Image Dataset* (HBFMID) (Orvile, 2023c) includes 1,539 annotated images (X-ray and MRI) covering fractures at multiple anatomical sites. The dataset is divided into training, validation, and testing sets and has undergone preprocessing (auto-orientation, resizing, contrast adjustments), supporting research in multi-modal fracture diagnosis.

19. FracAtlas. *FracAtlas* (Gupta, 2023) comprises over 14,000 X-ray scans collected from three major hospitals in Bangladesh, with 4,083 images manually annotated for bone fracture classification, localization, and segmentation. Annotations were conducted by expert radiologists and validated by a medical officer, providing a high-quality benchmark for fracture analysis.

20. pneumonia. The *pneumonia* dataset (Mooney, 2018) contains 5,863 chest X-ray images (anterior-posterior) of pediatric patients, labeled as either pneumonia or normal. Images underwent strict quality control and multi-expert grading, making the dataset reliable for training AI systems in pneumonia detection.

21. pax_ray. The *PAX-Ray++* dataset (Seibold, 2023) contains 7,377 chest radiographs (frontal and lateral views), with pseudo-labeled annotations for anatomical segmentation generated from projected thorax CT scans. The dataset is designed for segmentation tasks in chest X-ray analysis.

22. lung_segmentation. This dataset (Beosup, 2023) consists of over 500 X-ray scans labeled by radiologists, supporting machine learning research in lung region segmentation.

23. shadow. The *shadow* dataset (Hmchuong, 2023) includes normal and bone-suppressed chest X-ray images, along with augmented samples. It is intended for research on bone shadow suppression to aid in lung disease diagnosis.

24. Angiography. The *ARCADE* dataset (Manaenkov, 2023) features 3,000 X-ray coronary angiography frames with expert annotations for vessel segmentation, SYNTAX scoring, and stenosis detection. It is organized by task and includes cross-validated annotations, providing a rich resource for AI research in coronary artery disease diagnostics.

25. dental_panoramic. This panoramic dental radiograph dataset (Lokisilvres, 2023) includes segmentation masks for 31 dental disease classes, such as caries, crowns, implants, bone loss, fractures, and more. It is intended for comprehensive dental disease detection and segmentation research.

26. ALHI. The *ALHI* dataset (Rahman, 2022) is a curated collection of 200 hip implant X-ray images from various medical sources, annotated and validated by orthopedic and clinical experts. The dataset includes images with diverse implant types and clinical conditions, supporting research on hip implant assessment.

27. humerus_fractures. The *humerus_fractures* dataset (Paspuel, 2024) contains X-ray images depicting both fractured and non-fractured humeri, supporting automated diagnosis of humerus fractures through deep learning.

28. multiclass_knee_osteoporosis. This dataset (Gobara, 2023a) offers X-ray images and patient records classified into normal, osteopenia, and osteoporosis categories, facilitating the automated diagnosis and classification of knee osteoporosis.

29. rsna-breast-cancer-detection. The *RSNA Breast Cancer Detection* dataset (Thakur, 2024) provides breast X-ray image regions of interest (ROIs) in PNG format, without labels, for studies on automated detection in breast imaging.

30. RANZCR. The *RANZCR* dataset (RANZCR, 2021) is intended for detecting the presence and position of catheters and lines on chest X-rays. It contains image IDs, binary labels for multiple types of catheters, and patient identifiers, along with associated CSV metadata.

31. FractureFusion. *FractureFusion* (Dutta, 2023) is a diverse dataset capturing a wide variety of bone fracture cases, including avulsion, comminuted, greenstick, and spiral fractures, suitable for developing comprehensive fracture classification models.

32. HeelBone. The *Heel Bone X-Ray* dataset (Taher, 2023) comprises 3,956 foot X-rays labeled for normal, heel spur, and severe heel spur complications. Images were sourced from Kirkuk General Hospital and cross-verified by orthopedic and radiology specialists, supporting disease classification in foot imaging.

HISTOPATHOLOGY:

The histopathology dataset collection in FedAgentBench covers a wide range of diseases and task types, making it a comprehensive benchmark for evaluating LLM agents in digital pathology. It spans various cancer types, including breast (e.g., *breast_histo*, *BreaKHis_400X*, *BreCaHAD*), ovarian (*ovarian_cancer*), gastric (*gastric_cancer*), kidney (*kmc_kidney*), melanoma, and nasopharyngeal carcinoma (*NPC-88k-Public*). The datasets support multiple learning paradigms such as binary and multi-class classification (*lung_and_colon*, *EBHI*), segmentation (*MonuSeg*, *PanNuke*), detection of mitotic figures (*ULMS*), and multimodal image-to-text learning (*histo-img-text*). Some datasets, like *choledoch*, incorporate hyperspectral imaging, while others like *CellNet* aggregate thousands of high-resolution images across organ types, facilitating generalization studies. Fine-grained annotations by expert pathologists (e.g., in *BreCaHAD*, *NPC-88k-Public*, *MonuSeg*) add clinical reliability. Together, these datasets reflect a realistic landscape of digital histopathology rich in diagnostic complexity, varied in modality and scale, and suitable for evaluating both general-purpose and specialized LLM agents in federated clinical settings. The exact dataset descriptions for each file are available in Listing 8 and summarized as follows:

1. breast_histo. The *Breast Histopathology Images* dataset (Mooney, 2024) focuses on Invasive Ductal Carcinoma (IDC), the most common breast cancer subtype. The original dataset comprises 162 whole mount slides scanned at 40x magnification, from which 277,524 patches of size 50×50 were extracted (198,738 IDC negative and 78,786 IDC positive). Patch filenames encode patient ID, spatial coordinates, and IDC class (0 for non-IDC, 1 for IDC). Only images are provided, with no additional labels.

2. BreaKHis_400X. The *BreaKHis_400X* dataset (Forderation, 2024) is derived from the BreaKHis database, which contains microscopic biopsy images of benign and malignant breast tumors. This subset includes images acquired at 400x optical zoom, with training and test data stored in separate folders. Images only are provided; no labels are included.

3. lung_and_colon. The *Lung and Colon Cancer Histopathological Images* dataset (MVD, 2024a) contains 25,000 JPEG images of size 768×768 pixels, covering five classes: lung benign tissue, lung adenocarcinoma, lung squamous cell carcinoma, colon adenocarcinoma, and colon benign tissue. Images were generated from HIPAA-compliant and validated original samples (750 lung and 500 colon images) and augmented using the Augmentor package to create a balanced dataset of 5,000 images per class.

4. gastric_cancer. The *Gastric Cancer Histopathology Tissue Image Dataset* (GCHTID) (Orvile, 2024) comprises 31,096 non-overlapping images (224×224 pixels), extracted from H&E-stained pathological slides from Harbin Medical University Cancer Hospital. Images are categorized into eight tissue types, including adipose, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, and tumor, enabling research on the tumor microenvironment in gastric cancer.

5. gastro_cancer_msi_vs_mss. The *Gastrointestinal Cancer MSI MSS Prediction* dataset (Justin, 2024) contains histological images for the classification of microsatellite instability (MSI) versus

microsatellite stability (MSS) in gastrointestinal cancer, supporting research in histopathology image analysis with CNNs and transfer learning.

6. breast_cancer_segmentation. The *Breast Cancer Cell Segmentation* dataset (MVD, 2024b) contains 58 H&E stained histopathology images with expert annotations for breast cancer cell detection and segmentation. The challenging task is cell segmentation for subsequent classification into benign and malignant cells, supported by ground truth data for algorithm development.

7. ovarian_cancer. The *Ovarian Cancer & Subtypes Dataset Histopathology* (Pieces, 2024) contains histopathology images representing four subtypes of ovarian cancer as well as non-cancerous tissue. The dataset is referenced as: Kasture, Kokila (2021), "Ovarian-Cancer&SubtypesDatasetHistopathology", Mendeley Data, V1, doi: 10.17632/kztymsrjx9.1.

8. breast_cancer_histo. The *Breast Cancer Histopathology* dataset (Kumar, 2024) includes JPG images labeled as benign or malignant, supporting automated breast cancer classification from histopathological images.

9. BreCaHAD. The *BreCaHAD* (Breast Cancer Histopathological Annotation and Diagnosis) dataset (TruthIsNeverLinear, 2024) comprises 162 annotated H&E-stained images, supporting automated classification of histological structures into six classes: mitosis, apoptosis, tumor nuclei, non-tumor nuclei, tubule, and non-tubule. See: `https://bmcresnotes.biomedcentral.com/articles/10.1186/s13104-019-4121-7`.

10. melanoma. The *melanoma* dataset (Haashaatif, 2024) is designed for the development of deep learning models for nuclei and tissue segmentation in melanoma H&E-stained histopathology. It addresses challenges of melanocyte mimicry and includes nuclei and tissue annotations to facilitate studies on tumor-infiltrating lymphocytes and predictive/prognostic tasks.

11. choledoch. The *Choledoch* dataset (HFUTYBX, 2024) introduces both microscopy hyperspectral and color images for cholangiocarcinoma, including 880 scenes from 174 individuals (689 partial cancer, 49 full cancer, 142 non-cancer). All cancer areas are precisely labeled by experienced pathologists. More information is available in: `https://ieeexplore.ieee.org/document/8869757`. The dataset includes suggested train/val/test splits.

12. histopath-sn. The *histopath-sn* Kaggle dataset (Feng, 2024) focuses on classifying patches and patients from bronchus and lung samples. Both images and labels are provided, with recommended train and test splits given in `train_labels.csv` and `test_labels.csv`.

13. ULMS. The *Uterine Leiomyosarcoma (ULMS)* dataset (Lee, 2024) targets mitosis detection in ULMS, the most common uterine sarcoma. Images were collected in collaboration with pathologists and annotated for mitosis, aiding AI-based approaches for automatic mitosis detection and grading.

14. MonuSeg. The *MonuSeg* dataset (Dinh, 2024) comprises 24 training images (originally 30, 1000×1000 pixels) with 21,623 annotated nuclei from seven organs, and a test set of 58 images (8 from MonuSeg, 50 from the TNBC dataset). Annotations were made by one expert pathologist and two research fellows using consensus peer review.

15. kmc_kidney. The *KMC Kidney Histopathology* dataset (Dwivedi, 2024) includes non-cancerous (Grade-0) and cancerous (Grades 1-4) images of renal clear cell carcinoma, collected at Kasturba Medical College (KMC), India. Images were stained with H&E and labeled according to grade, supporting studies in kidney cancer histopathology.

16. histo-img-text. The *histo-img-text* dataset (Reasat, 2024) comprises histopathology image-text pairs, including over 32k PNGs, 40k JPGs, and a CSV file with 217,052 captioned image entries. The dataset is designed for multimodal studies, such as image-to-text and vision-language modeling.

17. cellnet. *CellNet* is a large, curated dataset (Capocyan, 2024) featuring over 120,000 high-quality medical images from more than 20 organ/cancer classes. Images were aggregated from diverse repositories and medical labs, supporting comprehensive research in computational pathology.

18. PanNuke. The *PanNuke* dataset (Lad, 2024) is a semi-automatically generated nuclei instance segmentation and classification dataset. It covers 481 visual fields across 19 tissue types, containing 205,343 labeled nuclei with segmentation masks, enabling tissue type segmentation and generalization to new tissue domains.

19. NPC-88k-Public. The *NPC-88k-Public* dataset (Munirah, 2024) includes 88,000 histopathology patches from 17 whole slide images across three institutions. Annotated regions include normal, lymphoid hyperplasia (LHP), nasopharyngeal inflammation (NPI), and nasopharyngeal carcinoma (NPC), with concordance among at least two pathologists.

20. EBHI. The *EBHI* dataset (Alibabaei78, 2024) comprises 4,456 histopathology images and corresponding ground truth segmentations, including normal, polyp, low-grade and high-grade intraepithelial neoplasia, serrated adenoma, and adenocarcinoma. Images are paired with ground truth labels to support segmentation and classification research.

MRI:

Our collection of 28 Magnetic Resonance Imaging (MRI) datasets supports a diverse array of machine learning tasks such as binary and multi-class classification, anatomical and pathological segmentation, anomaly detection, multi-modal image registration, and physiological parameter estimation. The included datasets range from unlabeled brain scans (*Brain MRI Images*) to richly annotated clinical benchmarks such as *BraTS*, *WMH*, and *ISLES 2015*, covering tumor segmentation, white matter lesion detection, and ischemic stroke assessment. Cardiac datasets like *ACDC* facilitate diagnosis of specific heart conditions, while spine-related datasets such as the *RSNA 2024 Lumbar Spine Challenge* and *Foraminal Stenosis MRI* target degenerative spinal diseases. Other specialized collections, including *Facial MRI*, *Prostate MRI*, and multi-modal datasets (e.g., *MRI-PET Brain Scans*), enable cross-domain generalization and analysis. Together, this curated set of MRI datasets provides a foundation for training and benchmarking AI systems across a broad range of anatomical regions and diagnostic challenges.

1. Brain MRI Images A Kaggle dataset (bra, b) containing diverse brain MRI images sourced from multiple datasets, offering a range of anatomical variations and imaging contrasts.

2. Alzheimer Classification Brain MRI dataset (alz) labeled for Alzheimer's disease classification into four categories: Mild Demented, Moderate Demented, Non-Demented, and Very Mild Demented.

3. Brain Cancer Brain MRI images (bra, a) collected from hospitals in Bangladesh for classification into Brain Glioma, Brain Meningioma, and Pituitary Tumor classes.

4. Brain Tumour A labeled brain tumor dataset (bra, e) for binary classification (tumor vs. non-tumor) and unlabeled prediction samples for testing.

5. 4 Class Brain Tumour A brain MRI dataset (bra, d) for classifying tumors into Benign, Malignant, and Pituitary types.

6. Heat MRI Left Atrial Segmentation A segmentation dataset (hea) of left atrial structures in cardiac MRI provided by King's College London.

7. PMRAM MRI brain cancer dataset (pmr) with four classes (Glioma, Meningioma, Pituitary, No Tumor), standardized to 512×512 resolution and augmented from 1600 base images.

8. Hippocampal Sparing Unlabeled DICOM-format MRI slices (hip) of 25 patients for hippocampal sparing studies, organized per patient.

9. Spine Spine MRI scans (spi) from a single patient with labeled dystrophic anomalies and accompanying radiology reports.

10. Brain Tumour CT MRI A brain tumor dataset (bra, c) composed of both MRI and CT images, labeled for tumor detection and suitable for training diagnostic models.

11. BraTS 2019 Multimodal brain MRI dataset (Menze et al., 2014) (T1, T1Gd, T2, FLAIR) with expert segmentations for tumor subregions, formatted as NIfTI (.nii.gz) files.

12. Bone Fractures MRI X-ray Multi-modal dataset (hbf) including MRI and X-ray scans for bone fracture detection across different body regions.

13. Alzheimer Detection Preprocessed MRI scans (LaMontagne et al., 2019) from the OASIS-1 dataset labeled for Alzheimer's detection tasks.

14. Stroke Head MRI MRI brain scans (str) with segmentations of stroke lesions from patients with cerebrovascular conditions.

15. MRI PET Brain Scans Paired MRI and PET DICOM scans (mri) for brain tumors, aimed at multi-modal registration and Dice score evaluation.

16. OASIS-1 Processed MRI scans of 1688 subjects across Alzheimer's Disease (AD), Cognitively Normal (CN), and Mild Cognitive Impairment (MCI) groups (oas).

17. Abdomen MRI Abdominal MRI dataset (abd) with object detection annotations and bounding boxes in CSV format.

18. Facial MRI Facial MRI scans (fac) including sagittal and axial slices for anomaly detection, segmentation, and 3D anatomical modeling.

19. Prostate Multi-parametric prostate MRI scans (pro, a) with manual segmentations for clinical segmentation research.

20. Glioma TCGA-LGG-based MRI dataset (gli) for low-grade glioma detection with segmentation masks and associated genomics metadata.

21. Phantom Longitudinal MRI dataset (pha) of a single healthy subject scanned on 116 scanners over 2.5 years to analyze scanner variability.

22. ACDC: Automated Cardiac Diagnosis Challenge Dataset The ACDC (Bernard et al., 2018) dataset consists of cine-MRI scans, categorized into five balanced cardiac pathology classes: Normal (NOR), Myocardial Infarction (MINF), Dilated Cardiomyopathy (DCM), Hypertrophic Cardiomyopathy (HCM), and Abnormal Right Ventricle (ARV). Each class is defined by specific clinical parameters such as ejection fraction, wall thickness, and ventricular volumes, supporting robust machine learning development for automated cardiac function assessment.

23. Foraminal Stenosis MRI Dataset This dataset (for) comprises high-resolution lumbar spine MRI scans with segmentation masks and foraminal measurements, aimed at detecting and analyzing foraminal stenosis. It supports tasks such as nerve channel size analysis, stenosis classification, and monitoring of spinal degenerative conditions, enabling precise anatomical assessment and aiding in early diagnosis and treatment planning.

24. RSNA 2024 Lumbar Spine Degenerative Classification Challenge This RSNA-ASNR (RSN) dataset includes five lumbar spine degenerative conditions—Left/Right Neural Foraminal Narrowing, Left/Right Subarticular Stenosis, and Spinal Canal Stenosis—using lumbar spine MRI. The dataset includes severity scores (Normal/Mild, Moderate, Severe) across five disc levels (L1/L2 to L5/S1), enabling automated classification to support diagnosis and treatment planning for lower back pain and related conditions.

25. ATLAS v2.0 The Anatomical Tracings of Lesions After Stroke (ATLAS) v2.0 (Liew et al., 2022) dataset provides manually segmented T1-weighted MRI scans of individuals with stroke lesions. It includes lesion masks and anatomical metadata for over 600 subjects, with the aim of facilitating the development and benchmarking of automated stroke lesion segmentation methods.

26. BraTS The Brain Tumor Segmentation (BraTS) dataset provided through the Medical Segmentation Decathlon (MSD), comprises multi-modal MRI scans (T1, T1-Gd, T2, and FLAIR) of glioma patients with expert annotations of tumor sub-regions including the enhancing tumor, peritumoral edema, and necrotic core.

27. WMH The White Matter Hyperintensities (WMH) dataset (wmh) consists of T1 and FLAIR MRI scans from multiple institutions with voxel-wise annotations of WMH regions. Originally compiled for the WMH Segmentation Challenge at MICCAI 2017, the dataset captures variability across scanners and populations, making it a robust benchmark for automated WMH detection methods.

28. ISLES 2015 (SISS) The Ischemic Stroke Lesion Segmentation (ISLES) 2015 challenge dataset (isl), specifically the Sub-Acute Ischemic Stroke Lesion Segmentation (SISS) subtask, offers multi-modal MRI scans (including FLAIR, T1, DWI) with corresponding lesion masks for patients in the subacute phase post-stroke. It supports the development of methods for accurate ischemic stroke lesion segmentation and includes cases with diverse lesion locations and volumes.

FUNDUS:

Our Fundus image datasets span a broad range of tasks and clinical applications, reflecting the diagnostic richness of retinal imaging. These include segmentation datasets such as *Drishti-GS*, *RIMONE*, and *ONH Segmentation* for optic disc/cup analysis in glaucoma, and vessel segmentation benchmarks like *DRIVE* and *CHASE_DB1* for vascular assessment. Classification datasets such as *APTOS*, *MESSIDOR*, and *ARIA* support diabetic retinopathy grading, while multi-label datasets like *RFMID* and *ODIR-5K* address a broader set of ocular diseases. Lesion-level annotations in datasets like *IDRiD* and *E-Ophtha* enable fine-grained detection of diabetic pathologies. Additionally, niche datasets such as *e-ROP*, *Ocular Toxoplasmosis*, and *AMDP* target rare or longitudinal conditions. Others focus on preprocessed imaging (*CLAHE + ESRGAN Split FD*) or multi-modal metadata (*SMDG*, *DrHagis*). This diversity supports robust benchmarking across segmentation, classification, enhancement, and multimodal learning, forming the backbone of data-driven ophthalmic model development.

1. Drishti-GS This dataset (Sivaswamy et al., 2014) is used for glaucoma detection, providing optic disc and cup segmentation masks. It supports both segmentation and glaucoma classification tasks.

2. STARE The STARE dataset (STA) is used for retinal disease diagnosis and retinal vessel segmentation. Its main tasks include vessel segmentation and lesion detection.

3. IDRiD The Indian Diabetic Retinopathy Image Dataset (IDRiD) (ind) provides pixel-level annotations for diabetic retinopathy (DR) lesions. It is used for lesion segmentation and DR grading.

4. DR This dataset (DR) is used for classifying diabetic retinopathy across 5 severity levels.

5. RIMONE A glaucoma dataset (Fumero et al., 2011) providing optic disc and cup annotations, mainly used for segmentation and glaucoma classification.

6. REFUGE A unified glaucoma evaluation dataset (ref), widely used for optical disc/cup segmentation and glaucoma classification.

7. CHASE_DB1 This dataset (cha) contains child retinal images with annotated vessels. It is primarily used for vessel segmentation tasks.

8. E-Ophtha Designed for diabetic retinopathy research, this dataset (Decenciere et al., 2013) includes images annotated for exudates and hemorrhages, supporting lesion detection.

9. ARIA A retinal image dataset used in diabetic retinopathy screening. It is mainly employed for DR classification.

10. IOSTAR A dataset of multi-modal retinal images, particularly used for optic disc segmentation tasks.

11. HRF The High-Resolution Fundus dataset is used for both vessel and optic disc segmentation, offering detailed structural annotations.

12. LES-AV This dataset supports artery and vein classification, distinguishing vessel types in fundus images.

13. PRIME-FP20 It is a high-resolution dataset of fundus images used for optic disc segmentation.

14. RIGA+ This is a glaucoma dataset derived from multiple sources, used for optic disc and cup segmentation.

15. APTOS It is part of the Kaggle Diabetic Retinopathy Challenge (2019), this dataset is used to grade DR severity from fundus images.

16. MESSIDOR It is a classic and widely used diabetic retinopathy dataset, primarily for classification tasks.

17. DRIVE It is one of the earliest vessel segmentation datasets, often used as a benchmark in fundus segmentation.

18. ORIGA The ORIGA dataset provides optic disc and cup annotations for segmentation task and glaucoma detection.

19. ODIR-5K The ODIR (Ocular Disease Intelligent Recognition) dataset contains over 5,000 retinal fundus images with multi-label annotations for eight ocular diseases, including diabetic retinopathy, glaucoma, cataract, AMD, hypertension, and others. It supports multi-label classification tasks.

20. RFMID The Retinal Fundus Multi-Disease Image Dataset (RFMID) includes 3,200+ images annotated for 19 different conditions. It is intended for multi-label classification tasks and supports the development of fundus-based diagnostic models for diverse ocular diseases.

21. MESSIDOR-2 DF MESSIDOR-2 is the second edition of the MESSIDOR diabetic retinopathy dataset. It includes fundus images with diabetic retinopathy severity labels.

22. Glaucoma datasets (EYE-PACS) EYE-PACS is a large-scale dataset used primarily for diabetic retinopathy grading in the Kaggle challenge.

23. Retina blood vessel segmentation dataset This fundus dataset is used for vessel segmentation.

24. DR Diagnosis Dataset This dataset is used for classifying diabetic retinopathy severity based on retinal fundus images.

25. DDR Dataset The Diabetic Retinopathy Detection from Retina Images (DDR) dataset includes fundus images annotated for DR severity and pixel-level lesion types (e.g., exudates, hemorrhages). It supports both classification and lesion segmentation tasks.

26. Hypertensive Retinopathy This dataset contains fundus images annotated for signs of hypertensive retinopathy. While rare and usually hospital-specific, it is used for classification and grading of HR severity.

27. SUSTECH + SYSU Dataset This entry combines data from SUSTech and Sun Yat-sen University (SYSU), curated for research in glaucoma, diabetic retinopathy, and related diseases. It supports classification tasks across multiple disease categories.

28. RITE The Retinal Images vessel Tree Extraction (RITE) dataset, derived from DRIVE, includes ground truth for artery and vein segmentation. It is used to differentiate between arterial and venous vessels in retinal images.

29. CLAHE + ESRGAN Split FD This dataset represents a preprocessed variant of fundus images where contrast enhancement (CLAHE) and super-resolution techniques (ESRGAN) have been applied. It is used to improve image quality for downstream classification tasks.

30. Myopia Image Dataset This dataset consists of retinal fundus images labeled for myopia classification.

31. ACRIMA ACRIMA is fundus dataset used for glaucoma detection.

32. and 33. Retina Fundus Dataset (CHASE_DB1, DRIVE) CHASE_DB1 and DRIVE are fundus datasets used for retinal vessel segmentation, i.e., for segmenting blood vessels in fundus images.

34. Cataract Classification Dataset This is used for binary classification of cataract presence in fundus images.

35. MURED The Multicenter Retinal Disease Dataset (MURED) aggregates retinal images across multiple institutions and includes annotations for diabetic retinopathy, glaucoma, age-related macular degeneration (AMD), and other conditions. It is primarily used for multi-class classification of retinal diseases.

36. Optic Disc Cup Fundus Image This dataset contains annotations for optic disc and cup structures. These datasets are used for segmentation tasks and for calculating cup-to-disc ratio, an important indicator in glaucoma diagnosis.

37. ROFT This is a retinal and ocular fundus image dataset with 8 disease labels for fundus images - normal, diabetes, glaucoma, cataract, age-related macular degeneration, hypertension, pathological myopia and other diseases/abnormalities. It also has 7 labels for OCT: age-related macular degeneration, diabetic macular edema, epiretinal membrane, normal, retinal artery occlusion, retinal vein occlusion, vitreomacular interface diseases.

38. Eye Disease Image Dataset A fundus dataset for detection of eye-related 10 conditions - central serous chorioretinopathy, diabetic retinopathy, disc edema, glaucoma, healthy, macular scar, myopia, pterygium, retinal detachment, and retinitis pigmentosa.

39. FIVES The FIVES dataset (Fundus Image Vessel Extraction and Segmentation) is used for vessel segmentation tasks. It provides pixel-level annotations for blood vessel structures.

40. AMDP Dataset This refers to the Age-related Macular Degeneration Prediction dataset which is longitudinal ophthalmic dataset.

41. AGAR 300 A Microaneurysms Fundus Dataset that consists of color fundus images showing signs of microaneurysms for early DR detection.

42. SMDG It is a standardized fundus glaucoma dataset consisting of full-fundus glaucoma images with image metadata on optic disc/cup segmentation and blood vessel segmentation.

44. Fundus segmentation dataset It is a unified retinal image dataset for assessing glaucoma with reference segmentation labels of optic disc and cup.

45. Hypertensive retinopathy dataset It is a fundus dataset for binary classification regarding presence or absence of hypertensive retinopathy.

46. DR grading dataset It is a fundus dataset for grading the severity of diabetic retinopathy.

47. G1020 dataset It is a fundus dataset for glaucoma classification and contain 1020 high resolution colour fundus images. It also provides annotations for glaucoma diagnosis, optic disc and cup segmentation, vertical cup to disc ratio, etc.

48.Ocular Toxoplasmosis dataset It is a fundus dataset used for detection of Toxoplasmosis chorioretinitis and has three classes - healthy eye, active and inactive chorioretinitis.

49. ARIA dataset It is a fundus dataset used for detection of any of three classes: healthy, AMD and Diabetes.

50. Fundus 4 categories dataset It is a fundus dataset used for detection of normal, cataract, glaucoma and diabetic retinopathy.

51. ONH Segmentation dataset It is an optic disc and cup mask segmentation fundus dataset

52. DrHagis dataset It is a fundus dataset for detection of diabetic retinopathy, hypertension, age-related macular degenration and glaucoma.

53. Driona DB dataset It is a fundus dataset for optic disc segmentation.

54. Cattle Retinal Fundus Images A unique dataset featuring retinal fundus images from cattle, useful for comparative studies and veterinary ophthalmology research.

55. Preprocessed Eye Diseases Fundus Images It offers preprocessed fundus images enhanced using techniques like CLAHE and ESRGAN, facilitating improved classification performance.

56. Retina Fundus Image Registration Dataset (FIRE) It comprises 129 retinal images forming 134 image pairs, designed for evaluating image registration algorithms.

57. 1000 Fundus Images with 39 Categories This dataset comprises 1,000 fundus images categorized into 39 distinct classes, offering a diverse set for multi-class classification tasks.

58. PAPILA: Retinal Fundus Images Dataset The PAPILA dataset includes fundus images and clinical data from both eyes of individual patients for glaucoma assessment. It provides optic disc and cup segmentations, along with patient-level glaucoma labels derived from clinical evaluations.

59. Diabetic Retinopathy Diagnosis Dataset A large-scale retinal image dataset designed for the diagnosis of diabetic retinopathy, supporting medical image analysis and automated disease grading.

60. Vessel Tree Extraction Dataset This dataset supports comparative research on artery and vein segmentation or classification in retinal fundus images, facilitating the development and benchmarking of vessel-type analysis methods.

61. DiaRetDB1: Diabetic Retinopathy Benchmark Dataset DiaRetDB1 includes retinal fundus images with expert-annotated ground truth for key lesions such as hard and soft exudates, microaneurysms, and hemorrhages, along with both the original images and raw annotation data.

62. SynFundus The SynFundus is a synthetic fundus dataset includes annotations for eleven ocular diseases: diabetic retinopathy, age-related macular degeneration, anomalies of the optic nerve, choroidal retinal pathology, degenerative and pathological myopia, diabetic macular edema, epimacular membrane, glaucoma, hypertensive retinopathy, and retinal vein occlusion. These conditions cover a broad range of structural and vascular retinal abnormalities, supporting diverse diagnostic research in ophthalmology.

63. AIROGS The AIROGS dataset (De Vente et al., 2023) comprises fundus photographs from diverse ethnicities and imaging devices. It supports two main tasks: referable glaucoma classification and detection of ungradable images to simulate real-world screening conditions.

C.2 Sample dataset description files:

Sample dataset description files are shown in Listings 6-8. The datasets are then partitioned into different clients and utilized by the client selector agents to decide whether to choose the client for federated analysis.

Listing 6: Dataset Descriptions for Dermatology Modality

```
[
    {
        "Dataset Name": "augmented_skin_condition_dataset_kaggle",
        "Dataset Description": "augmented_skin_condition_dataset_kaggle
            is a skin disease classification dataset containing images of
             six different dermatological conditions: 'Acne', 'Carcinoma
            ', 'Eczema', 'Keratosis', 'Milia', and 'Rosacea'. It contains
             six subfolders, with each subfolder containing images of the
             corresponding class (disease) specified in the name of the
            subfolder. ",
        "Dataset
            Path": "skin_dataset/augmented_skin_condition_dataset_kaggle"
    },
    {
        "Dataset Name": "DDI_skin_dataset",
        "Dataset Description": "DDI_skin_dataset is a skin cancer
            classification dataset with diverse skin tone representation
            that contains 1 subfolder 'images' and 2 CSV files. Focus on
            the columns: 'DDI_file' (for the image path) and 'malignant'
            (the class label) of the csv file 'ddi_metadata.csv'. 'True'
            in 'malignant' column means malignant whereas 'False' means
            benign. ",
        "Dataset Path": "skin_dataset/DDI_skin_dataset"
    },
    {
        "Dataset Name": "Derma7PT",
        "Dataset Description": "Derma7PT is a skin disease classification
             dataset containing a subfolder 'images' and a csv file 'meta
            .csv'. Focus on the columns 'clinic' and 'derm' for the image
             file path as well as the column 'diagnosis' of the csv file
            that has 10 disease types: 'basal cell carcinoma', 'nevus', '
            dermatofibroma', 'lentigo', 'melanoma', 'melanoma metastasis
            ', 'melanosis', 'miscellaneous', 'seborrheic keratosis', '
            vascular lesion'. ",
        "Dataset Path": "skin_dataset/Derma7P"
    },
    {
        "Dataset Name": "Dermatology_tabular dataset",
        "Dataset Description": "Dermatology_tabular dataset is a tabular
            (non-image) dataset containing clinical features for
            diagnosing skin diseases. ",
        "Dataset Path": "skin_dataset/Dermatology_tabular dataset"
    },
    {
```

```
        "Dataset Name": "Dermis",
        "Dataset Description": "Dermis is a skin disease dataset with
            benign and malignant cases, supporting both classification
            and segmentation tasks. It has two sub-folders 'benign' and '
            melanoma'. In each of these sub-folders, we have two sub-
            folders 'contour' (that has the segmentation masks) and '
            images' (that has the original images). ",
        "Dataset Path": "skin_dataset/Dermis"
    },
    {
        "Dataset Name": "Dermnet",
        "Dataset Description": "Dermnet contains a very broad collection
            of skin disease images. It has 23 sub-folders covering 23
            disease categories namely 'Acne and Rosacea', 'Actinic
            Keratosis Basal Cell Carcinoma and other Malignant Lesions',
            'Atopic Dermatitis Photos', 'Bullous Disease Photos', '
            Cellulitis Impetigo and other Bacterial Infections', 'Eczema
            Photos', 'Exanthems and Drug Eruptions', 'Hair Loss (Alopecia
            ) and other Hair Diseases', 'Herpes HPV and other STDs Photos
            ', 'Light Diseases and Disorders of Pigmentation', 'Lupus and
             other Connective Tissue Diseases', 'Melanoma Skin Cancer
            Nevi and Moles', 'Nail Fungus and other Nail Disease', '
            Poison Ivy Photos and other Contact Dermatitis', 'Psoriasis
            pictures and Lichen Planus and related Diseases', 'Scabies
            Lyme Disease and other Infestations and Bites', 'Seborrheic
            Keratoses and other Benign Tumors', 'Systemic Disease', '
            Tinea Ringworm Candidiasis and other Fungal Infections', '
            Urticaria Hives', 'Vascular Tumors', 'Vasculitis Photos', '
            Warts Molluscum and other Viral Infections'. ",
        "Dataset Path": "skin_dataset/Dermnet"
    },
    {
        "Dataset Name": "Dermquest",
        "Dataset Description": "Dermquest is a skin disease
            classification and segmentation dataset containing images of
            benign and malignant skin diseases. It has two sub-folders '
            benign' and 'melanoma'. In each of these sub-folders, we have
             two sub-folders 'contour' (that has the segmentation masks)
            and 'images' (that has the original images). ",
        "Dataset Path": "skin_dataset/Dermquest"
    },
    {
        "Dataset Name": "fitzpatrick17k",
        "Dataset Description": "fitzpatrick17k is a large skin lesion
            dataset with a wide range of dermatological diseases. It has
            a sub-folder 'finalfitz17k' which contains all images and two
             csv files 'fitzpatrick17k_disease.csv' and '
            Fitzpatrick17k_morphology.csv'. Focus on the column 'md5hash'
             for filename and the column 'three_partition_label' that
            contains three disease labels: 'non-neoplastic', 'benign', '
            malignant' in the file 'fitzpatrick17k_disease.csv'. ",
        "Dataset Path": "skin_dataset/fitzpatrick17k"
    },
    {
        "Dataset Name": "ISIC2018_HAM10000",
        "Dataset Description": "ISIC2018_HAM10000 is a skin lesion
            classification and segmentation dataset. It has a sub-folder
            'HAM10000_images_combined_600x450' that contains original
            images as well as a sub-folder 'HAM10000_segmentations_mask'
            that contains the corresponding segmentation masks. The
            classification labels can be found in the 'dx' column of the
             csv file 'ISIC2018_Task3_Test_GroundTruth.csv' including '
            Melanocytic Nevus (nv)', 'Benign Keratosis-like Lesions (bkl)
            ', 'Melanoma (mel)', 'Basal Cell Carcinoma (bcc)', 'Actinic
            Keratosis / Bowen's Disease (akiec)', 'Vascular Lesions (vasc
```

```
            )', 'Dermatofibroma (df)'. The corresponding image names can
            be found in the column 'image_id' of the same csv file. ",
        "Dataset Path": "skin_dataset/ISIC2018_HAM10000"
    },
    {
        "Dataset Name": "ISIC_2016",
        "Dataset Description": "ISIC_2016 is a skin lesion dataset for
            classification and segmentation, focused on skin cancer
            detection. It has two sub-folders 'ISBI2016_ISIC_images' that
             contain original images and 'ISBI2016_ISIC_segmentation_mask
            ' that has segmentation masks. The csv file '
            ISBI2016_ISIC_binary_classification_Training_GroundTruth.csv'
             has two columns - the first column being image names and
            second column being binary disease labels: 'benign' and '
            malignant'. ",
        "Dataset Path": "skin_dataset/ISIC_2016"
    },
    {
        "Dataset Name": "ISIC_2017",
        "Dataset Description": "ISIC_2017 is a skin lesion classification
             and segmentation dataset with a focus on melanoma and
            seborrheic keratosis diagnosis. It has two sub-folders: '
            images' that contain original images and 'Segmentation_masks'
             that has segmentation masks. There is a csv file 'ISIC-2017
            _GroundTruth' with the columns 'image_id' that contains image
             filenames, 'melanoma' that contains binary labels
            corresponding to presence (1) and absence (0) of melanoma,
            and 'seborrheic keratosis' that contains binary labels
            corresponding to presence (1) and absence (0) of seborrheic
            keratosis. ",
        "Dataset Path": "skin_dataset/ISIC_2017"
    },
    {
        "Dataset Name": "ISIC_2019",
        "Dataset Description": "ISIC_2019 is an extended skin disease
            classification dataset. It has one sub-folder: 'images' that
            contain original images. In the CSV file '
            ISIC_2019_Training_GroundTruth.csv', the 'image' column
            contains the image file names and 9 other columns represent
            the presence (1) or absence (0) of 9 classes namely Melanoma
            (MEL), Nevus (NV), Basal Cell Carcinoma (BCC), Actinic
            Keratosis / Bowen's Disease (AK), Benign Keratosis-like
            Lesions (BKL), Dermatofibroma (DF), Vascular Lesions (VASC),
            Squamous Cell Carcinoma (SCC) and Unknown (UNK). ",
        "Dataset Path": "skin_dataset/ISIC_2019"
    },
    {
        "Dataset Name": "ISIC_2020",
        "Dataset Description": "ISIC_2020 is a binary classification
            dataset of skin lesions (benign vs malignant). It has one sub
            -folder: 'images' that contain original images. In the CSV
            file 'ISIC_2020_Training_GroundTruth.csv', the 'image_name'
            column contains the image file names and the '
            benign_malignant' column contains the corresponding disease
            labels on malignant or benign. ",
        "Dataset Path": "skin_dataset/ISIC_2020"
    },
    {
        "Dataset Name": "ISIC_2024",
        "Dataset Description": "ISIC_2024 is an updated ISIC skin disease
             dataset primarily for melanoma classification (binary:
            benign vs malignant). It has one sub-folder: 'images' that
            contain original images. In the CSV file '
            ISIC_2024_Training_GroundTruth.csv', the 'isic_id' column
            contains the image file names and the 'malignant' column
```

```
        contains the corresponding disease labels on malignant or
        benign. '0' means benign and '1' means malignant. ",
    "Dataset Path": "skin_dataset/ISIC_2024"
},
{
    "Dataset Name": "Mednode",
    "Dataset Description": "Mednode is a skin disease dataset for
        binary classification. It has 2 sub-folders covering 2
        disease categories namely melanoma and nevus. ",
    "Dataset Path": "skin_dataset/Mednode"
},
{
    "Dataset Name": "Monkeypox_Skin_Image_Dataset",
    "Dataset Description": "Monkeypox_Skin_Image_Dataset is a dataset
         for skin disease classification and has four sub-folders (
        with data belonging to the corresponding disease category)
        named: 'Chickenpox', 'Measles', 'Monkeypox', and 'Normal'. ",
    "Dataset Path": "skin_dataset/Monkeypox_Skin_Image_Dataset"
},
{
    "Dataset Name": "PAD_UFES_20",
    "Dataset
        Description": "PAD_UFES_20 is a skin disease classification
        dataset. It contains a sub-folder 'images' containing the
        original images and a csv file called 'metadata.csv' that
        contains the image ids under the column 'img_id' and disease
        labels under the column 'diagnostic' which contains 6 disease
         categories with corresponding abbreviations: Melanoma (MEL),
         Melanocytic Nevus (NEV), Basal Cell Carcinoma (BCC), Actinic
         Keratosis / Bowen's Disease (ACK), Seborrheic Keratosis (SEK
        ), and Squamous Cell Carcinoma (SCC). ",
    "Dataset Path": "skin_dataset/PAD_UFES_20"
},
{
    "Dataset Name": "PH2Dataset",
    "Dataset Description": "PH2Dataset is a skin lesion
        classification and segmentation dataset. It has a sub-folder
        'PH2 Dataset images' which in turn has two sub-folders '
        all_dermoscopic_images' that contain all the original images
        and 'segmentation_mask' that contain all the segmentation
        masks. The folder has an xlsx file called 'PH2_dataset.xlsx'
        with a column called 'Image Name' that contains the image ids
         and a column 'Clinical Diagnosis' three disease classes : '
        Common Nevus', 'Atypical Nevus', and 'Melanoma' marked with '
        X' whenever that category is present in a given image. ",
    "Dataset Path": "skin_dataset/PH2Dataset"
},
{
    "Dataset Name": "scin_dataset",
    "Dataset Description": "scin_dataset is a multi-class skin
        disease classification dataset. It has a sub-folder '
        scin_images' that contains all the original images and two
        csv files. Follow the 'scin_cases.csv' file which has the
        image ids in the column 'case_id' and the disease classes
        under the 'related category' which should include the 9
        diseases: 'RASH', 'LOOKS_HEALTHY', 'OTHER_ISSUE_DESCRIPTION',
         'NAIL_PROBLEM', 'GROWTH_OR_MOLE', 'ACNE', '
        PIGMENTARY_PROBLEM', 'HAIR_LOSS', 'OTHER_HAIR_PROBLEM'. ",
    "Dataset Path": "skin_dataset/scin_dataset"
},
{
    "Dataset Name": "skin_disease_3_class",
    "Dataset Description": "skin_disease_3_class is a skin disease
        classification dataset that consists of a sub-folder 'images'
         which in turn has three sub-folders each consisting of one
```

```
            of the three classes indicated by the sub-folder name: 'acne
            ', 'atopic dermatitis', and 'basal cell carcinoma'. ",
        "Dataset Path": "skin_dataset/skin_disease_3_class"
    },
    {
        "Dataset Name": "skin_disease_classification_kaggle",
        "Dataset Description": "skin_disease_classification_kaggle is a
            skin disease classification dataset with a sub-folder 'files'
             that again contains three sub-folders each containing one of
             the three classes: 'acne', 'eye bags', and 'redness'. ",
        "Dataset Path": "skin_dataset/skin_disease_classification_kaggle"
    },
    {
        "Dataset Name": "skin_disease_kaggle_dataset",
        "Dataset
            Description": "skin_disease_kaggle_dataset is a skin cancer
            detection dataset that has 10 sub-folders for 10 disease
            classes with the corresponding sub-folder names: 'Atopic
            Dermatitis', 'Basal Cell Carcinoma (BCC)', 'Benign Keratosis-
            like Lesions (BKL)', 'Eczema', 'Melanocytic Nevi (NV)', '
            Melanoma', 'Psoriasis pictures Lichen Planus and related
            diseases', 'Seborrheic Keratoses and other Benign Tumors', '
            Tinea Ringworm Candidiasis and other Fungal Infections', and
            'Warts Molluscum and other Viral infections'. ",
        "Dataset Path": "skin_dataset/skin_disease_kaggle_dataset"
    },
    {
        "Dataset Name": "Skin Disease_Robo",
        "Dataset Description": "Skin Disease_Robo is a skin disease
            classification and object detection dataset. It has one sub-
            folder 'image' that contains all the original images and a
            csv file 'bounding_box_annotations.csv' with a column called
            'filename' that has all the image names and column 'class'
            that has 10 disease class labels: 'Acne', 'Atopic Dermatitis
            ', 'Chicken Skin', 'Eczema', 'Hansen's Disease-Leprosy', '
            Hansen's Disease-Leprosy-severe', 'Healthy skin', 'Psoriasis
            ', 'Ringworm', 'Warts'. It also contains coordinates for
            bounding box annotations for lesions in the columns 'xmin', '
            ymin', 'xmax', and 'ymax'. ",
        "Dataset Path": "skin_dataset/Skin Disease_Robo"
    },
    {
        "Dataset Name": "skin-infection-disease-dataset",
        "Dataset Description": "skin-infection-disease-dataset is a skin
            disease classification dataset focusing on infectious skin
            diseases. It has 8 sub-folders consisting diseases of each
            category - BA-cellulitis, BA-impetigo, FU-athlete-foot, FU-
            nail-fungus, FU-ringworm, PA-cutaneous-larva-migrans, VI-
            chickenpox, VI-shingles. ",
        "Dataset Path": "skin_dataset/skin-infection-disease-dataset"
    },
    {
        "Dataset Name": "skinL2_dataset",
        "Dataset Description": "skinL2_dataset is a skin cancer
            classification dataset with 8 sub-folders containing 8
            classes: 'Basal-cell Carcinoma', 'Dermatofibroma', '
            Hemangioma', 'Melanoma', 'Nevus', 'Psoriasis', 'Seborrheic
            Keratosis', and 'Others'. Optional metadata is available in '
            PlenoISLA_DatasetV1_info.xlsx'. ",
        "Dataset Path": "skin_dataset/skinL2_dataset"
    }
]
```

Listing 7: Dataset Descriptions for X-Ray Modality

```
[
    {
        "Dataset Name": "cov_19",
        "Dataset Description": "This is a database of chest X-ray images
            for COVID-19 positive cases along with Normal and Viral
            Pneumonia images.It has 3616 COVID-19 positive cases along
            with 10,192 Normal, 6012 Lung Opacity (Non-COVID lung
            infection), and 1345 Viral Pneumonia images and corresponding
             lung masks organized in different sub-folders.",
        "Dataset Path": "xray/cov_19"
    },
    {
        "Dataset Name": "bone_frac",
        "Dataset Description": "This dataset comprises fractured and non-
            fractured X-ray images covering all anatomical body regions,
            including lower limb, upper limb, lumbar, hips, knees, etc.
            The dataset is categorized into two subfolders containing
            fractured and non-fractured radiographic images.",
        "Dataset Path": "xray/bone_frac"
    },
    {
        "Dataset Name": "chest_tuberculosis_segmentation",
        "Dataset Description": "This dataset consists of 704 chest X-ray
            images for tuberculosis (TB) detection. The dataset contains
            both tuberculosis-positive and normal chest X-rays and are
            accompanied by lung segmentation masks (in separate
            subfolders) and clinical metadata as csv files.",
        "Dataset Path": "xray/chest_tuberculosis_segmentation"
    },
    {
        "Dataset Name": "xray/17_diseases",
        "Dataset Description": "The dataset consists of a collection of
            chest X-ray images in .jpg and .dcm formats. Types of
            diseases in the dataset: Abscess, Ards, Atelectasis,
            Atherosclerosis of the aorta, Cardiomegaly, Emphysema,
            Fracture, Hydropneumothorax, Hydrothorax, Pneumonia,
            Pneumosclerosis, Post inflammatory changes, Post traumatic
            ribs deformation, Sarcoidosis, Scoliosis, Tuberculosis and
            Venous congestion arranged in different subfolders.",
        "Dataset Path": "xray/17_diseases"
    },
    {
        "Dataset Name": "spr_age_gender",
        "Dataset Description": "SPR X-Ray Age and Gender Dataset. Used to
             help predict the age and gender of the patient based on the
            X-Ray image. Contains .png x-ray images in image subfolder
            with csv file containing gender and age.",
        "Dataset Path": "xray/spr_age_gender"
    },
    {
        "Dataset Name": "unifesp",
        "Dataset Description": "The UNIFESP X-Ray Body Part
            Classification Dataset. This is a dataset of 2481 X-rays of
            20 body parts + others, annotated in a multilabel fashion by
            radiology residents. Images are in DICOM format and Labels
            are categorical in csv file: Abdomen = 0, Ankle = 1, Cervical
             Spine = 2, Chest = 3, Clavicles = 4, Elbow = 5, Feet = 6,
            Finger = 7, Forearm = 8, Hand = 9, Hip = 10, Knee = 11, Lower
             Leg = 12, Lumbar Spine = 13, Others = 14, Pelvis = 15,
            Shoulder = 16, Sinus = 17, Skull = 18, Thigh = 19, Thoracic
            Spine = 20, Wrist = 21",
        "Dataset Path": "xray/unifesp"
    },
    {
```

```
        "Dataset Name": "knee",
        "Dataset Description": "It has 1,650 high-quality digital X-ray
            images of knee joints with a metadata file.",
        "Dataset Path": "xray/knee"
    },
    {
        "Dataset Name": "c19_radiograph",
        "Dataset Description": "COVID-19, lung opacity, normal and viral
            pneumonia chest X-ray (CXR) images are arranged in different
            sub-folders.",
        "Dataset Path": "xray/c19_radiograph"
    },
    {
        "Dataset Name": "simple_vs_community",
        "Dataset Description": "Bone Fracture X-ray Dataset: Simple vs.
            Comminuted Fractures organized in different subfolders",
        "Dataset Path": "xray/simple_vs_community"
    },
    {
        "Dataset Name": "nih_bbox",
        "Dataset Description": "This NIH Chest X-ray Dataset is comprised
             of 112,120 X-ray images with disease labels from 30,805
            unique patients. It has images in the image folder along with
             a label.csv with Class labels: 8 classes - Infiltrate,
            Atelectasis, Pneumonia, Cardiomegaly, Effusion, Pneumothorax,
             Mass, Nodule.",
        "Dataset Path": "xray/nih_bbox"
    },
    {
        "Dataset Name": "bone_break",
        "Dataset Description": "The dataset covers a range of bone
            fracture classes, such as avulsion fractures, comminuted
            fractures, fracture-dislocations, greenstick fractures,
            hairline fractures, impacted fractures, longitudinal
            fractures, oblique fractures, pathological fractures, and
            spiral fractures organized in separate subfolders",
        "Dataset Path": "xray/bone_break"
    },
    {
        "Dataset Name": "cov19_normal",
        "Dataset Description": "This dataset comprises a total of 800
            high-quality chest X-ray images, with 400 images depicting
            COVID-19 infected patients and the other 400 images
            representing normal cases (i.e., non-COVID patients) arranged
             in separate sub-folders.",
        "Dataset Path": "xray/cov19_normal"
    },
    {
        "Dataset Name": "dental",
        "Dataset Description": "Dental radiographs along with labels in
            csv files",
        "Dataset Path": "xray/dental"
    },
    {
        "Dataset Name": "bone_frac_small",
        "Dataset Description": "This dataset is designed for developing
            machine learning models for bone fracture classification and
            localization in tibia and fibula bones. It contains X-ray
            images in .PNG format along with labels in csv file",
        "Dataset Path": "xray/bone_frac_small"
    },
    {
        "Dataset Name": "knee_osteoporosis",
```

```
        "Dataset Description": "This knee XRay dataset contains 3 classes
            : normal, Osteopenia ,and Osteoporosis arranged in separate
            subfolders",
        "Dataset Path": "xray/knee_osteoporosis"
    },
    {
        "Dataset Name": "RNSA_pneumonia",
        "Dataset Description": "This dataset is a pre-processed version
            of the RSNA Pneumonia Detection Challenge dataset in PNG
            format along with the associated bounding box annotations as
            mask images. The metadata, including the patient information
            and bounding box coordinates, has been extracted and saved in
             CSV format.",
        "Dataset Path": "xray/RNSA_pneumonia"
    },
    {
        "Dataset Name": "8_object_detection",
        "Dataset Description": "Overview: The Chest X-ray 8 Subset
            dataset is a curated collection of chest radiographs for
            object detection models on thoracic diseases, with 790 images
             and 984 annotated bounding boxes in YOLO and Pascal VOC
            formats for diverse ML frameworks. Classes and Labels
            contained in associated csv file: 14 thoracic disease classes
             including Atelectasis, Cardiomegaly, Effusion, Infiltrate,
            Nodule, Mass, Pneumonia, Pneumothorax.",
        "Dataset Path": "xray/8_object_detection"
    },
    {
        "Dataset Name": "HBFMID",
        "Dataset Description": "Human Bone Fractures Multi-modal Image
            Dataset (HBFMID) is a collection of 1539 annotated medical
            images (X-ray and MRI) covering bone fractures in various
            locations (elbow, finger, forearm, humerus, shoulder, femur,
            shinbone, knee, hipbone, wrist, spinal cord, and some healthy
             bones) contained in the Image folder along with associated
            csv file",
        "Dataset Path": "xray/HBFMID"
    },
    {
        "Dataset Name": "FracAtlas",
        "Dataset Description": "It is a dataset of more than 14,000 X-Ray
             scans for classification, localization and segmentation of
            bone fractures. All the scans are available in JPG format
            along with proper annotations in separate sub-folders",
        "Dataset Path": "xray/FracAtlas"
    },
    {
        "Dataset Name": "pneumonia",
        "Dataset Description": "There are 5,863 X-Ray images (JPEG) and 2
             categories (Pneumonia/Normal) arranged in separate sub-
            folders",
        "Dataset Path": "xray/pneumonia"
    },
    {
        "Dataset Name": "pax_ray",
        "Dataset Description": "The PAX-Ray++ Dataset is a high-quality
            dataset designed to facilitate segmentation tasks for
            anatomical structures in chest radiographs available in Image
             subfolder and annotations in mask subfolder.",
        "Dataset Path": "xray/pax_ray"
    },
    {
        "Dataset Name": "lung_segmentation",
```

```
        "Dataset Description": "This dataset contains over 500 x-ray
            scans with clinical labels collected by radiologists
            available in separate subfolders.",
        "Dataset Path": "xray/lung_segmentation"
    },
    {
        "Dataset Name": "shadow",
        "Dataset Description": "Normal Chest X-ray images and Bone Shadow
             images along with csv file.",
        "Dataset Path": "xray/shadow"
    },
    {
        "Dataset Name": "Angiography",
        "Dataset Description": "The ARCADE dataset (Automatic Region-
            based Coronary Artery Disease Diagnostics using X-ray
            Angiography) is organized into two task-specific directories
            ('Task_Syntax_Segmentation' and 'Task_Stenosis_Segmentation')
            , each containing flattened 'Images/' and 'masks/' subfolders
            .",
        "Dataset Path": "xray/Angiography"
    },
    {
        "Dataset Name": "dental_panoramic",
        "Dataset Description": "Dental Disease Panoramic Dataset with
            segmentations on 31 classes: Classes: 0: Caries, 1: Crown, 2:
             Filling, 3: Implant, 4: Malaligned, 5: Mandibular Canal, 6:
            Missing teeth, 7: Periapical lesion, 8: Retained root, 9:
            Root Canal Treatment, 10: Root Piece, 11: Impacted tooth, 12:
             Maxillary sinus, 13: Bone Loss, 14: Fracture teeth, 15:
            Permanent Teeth, 16: Supra Eruption, 17: TAD, 18: Abutment,
            19: Attrition, 20: Bone defect, 21: Gingival former, 22:
            Metal band, 23: Orthodontic brackets, 24: Permanent retainer,
             25: Post-core, 26: Plating, 27: Wire, 28: Cyst, 29: Root
            resorption, 30: Primary teeth organized as different sub-
            folders",
        "Dataset Path": "xray/dental_panoramic"
    },
    {
        "Dataset Name": "ALHI",
        "Dataset Description": "All images include a stem and a cup of
            the hip implant, and the images have to be X-ray images along
             with csv file containing metadata.",
        "Dataset Path": "xray/ALHI"

    },
    {
        "Dataset Name": "humerus_fractures",
        "Dataset Description": "Deep Learning-Driven Diagnosis of Humerus
             Fractures from Radiographic Data. Images contain x-ray
            images of humerus fractures and non-fractures in separate
            subfolders.",
        "Dataset Path": "xray/humerus_fractures"
    },
    {
        "Dataset Name": "multiclass_knee_osteoporosis",
        "Dataset Description": "The dataset is divided into three primary
             categories: (1) Normal: Images of knees with no signs of
            osteoporosis., (2) Osteopenia: Images showing early stages of
             bone density loss, and (3) Osteoporosis: Images indicating
            advanced bone density degradation organized as different
            subfolders",
        "Dataset Path": "xray/multiclass_knee_osteoporosis"
    },
    {
        "Dataset Name": "rsna-breast-cancer-detection",
```

```
        "Dataset Description": "Region of Interests extracted from breast
             X-ray images. There are no labels, just .png images.",
        "Dataset Path": "xray/rsna-breast-cancer-detection"
    },
    {
        "Dataset Name": "RANZCR",
        "Dataset Description": "For detecting the presence and position
            of catheters and lines on chest x-rays. The .csv file
            contains image IDs, binary labels, and patient IDs with
            columns: Columns: StudyInstanceUID (unique ID for each image)
            , ETT - Abnormal (endotracheal tube placement abnormal), ETT
            - Borderline (borderline abnormal), ETT - Normal (normal),
            NGT - Abnormal (nasogastric tube placement abnormal), NGT -
            Borderline (borderline abnormal), NGT - Incompletely Imaged (
            inconclusive due to imaging), NGT - Normal (normal), CVC -
            Abnormal (central venous catheter placement abnormal), CVC -
            Borderline (borderline abnormal), CVC - Normal (normal), Swan
             Ganz Catheter Present, PatientID (unique ID for each patient
            ).",
        "Dataset Path": "xray/RANZCR"
    },
    {
        "Dataset Name": "FractureFusion",
        "Dataset Description": "From avulsion fractures to spiral
            fractures, this dataset is a rich repository of diverse cases
            , including comminuted fractures, fracture-dislocations,
            greenstick fractures, hairline fractures, impacted fractures,
             longitudinal fractures, oblique fractures, pathological
            fractures arranged as different subfolders",
        "Dataset Path": "xray/FractureFusion"
    },
    {
        "Dataset Name": "HeelBone",
        "Dataset Description": "Heel Bone X-Ray Dataset consists of 3,956
             X-ray images of the foot, primarily focused on detecting and
             classifying heel bone diseases with annotations arranged in
            label.csv",
        "Dataset Path": "xray/HeelBone"
    }
]
```

Listing 8: Dataset Descriptions for Histopathology Modality

```
[
    {
        "Dataset Name": "breast_histo",
        "Dataset Description": "Breast Histopathology Images with
            Invasive Ductal Carcinoma (IDC). There's no labels for this
            dataset, only images.",
        "Dataset Path": "histopathology/breast_histo"
    },
    {
        "Dataset Name": "BreaKHis_400X",
        "Dataset
            Description": "Breast cancer images on histopathology slides.
             The BreaKHis database contains microscopic biopsy images
            benign and malignant breast tumors in separate subfolders.",
        "Dataset Path": "histopathology/BreaKHis_400X"
    },
    {
        "Dataset Name": "lung_and_colon",
        "Dataset Description": "Lung and Colon Cancer Histopathological
            Images: 25000 images of 5 classes: Lung benign tissue, Lung
            adenocarcinoma, Lung squamous cell carcinoma, Colon
            adenocarcinoma, Colon benign tissue in separate subfolders.",
```

```
        "Dataset Path": "histopathology/lung_and_colon"
    },
    {
        "Dataset Name": "gastric_cancer",
        "Dataset Description": "Gastric Cancer Histopathology Tissue
            Image Dataset focuses on the tumor microenvironment (TME) and
             includes images categorized into eight distinct tissue types
            : ADI: Adipose (fat tissue), BACK: Background (non-tissue
            areas), DEB: Debris (cellular waste), LYM: Lymphocytes (
            immune cells), MUC: Mucus (protective secretion), MUS: Smooth
             Muscle (muscle tissue), NORM: Normal Colon Mucosa (healthy
            tissue for reference), STR: Cancer-associated Stroma (
            connective tissue around the tumor), TUM: Tumor (cancerous
            tissue) - all arranged in different subfolders. ",
        "Dataset Path": "histopathology/gastric_cancer"
    },
    {
        "Dataset Name": "gastro_cancer_msi_vs_mss",
        "Dataset Description": "The dataset contains histological images
            for MSI vs MSS classification in gastrointestinal cancer
            arranged in different sub-folders.",
        "Dataset Path": "histopathology/gastro_cancer_msi_vs_mss"
    },
    {
        "Dataset Name": "breast_cancer_segmentation",
        "Dataset Description": "Breast Cancer Cell Segmentation with
            corresponding images and masks in separate subfolders.",
        "Dataset Path": "histopathology/breast_cancer_segmentation"
    },
    {
        "Dataset Name": "ovarian_cancer",
        "Dataset Description": "Ovarian Cancer & Subtypes Dataset
            Histopathology: This dataset includes histopathology images
            of 4 subtypes of Ovarian cancer and also non cancerous
            histopathological images organized in separate subfolders",
        "Dataset Path": "histopathology/ovarian_cancer"
    },
    {
        "Dataset Name": "breast_cancer_histo",
        "Dataset Description": "breast cancer histopathology. JPG images
            with classifications benign or malignant organized as
            separate subfolders",
        "Dataset Path": "histopathology/breast_cancer_histo"
    },
    {
        "Dataset Name": "BreCaHAD",
        "Dataset
            Description": "a dataset for breast cancer histopathological
            annotation and diagnosis with images belonging to six classes
            , namely mitosis, apoptosis, tumor nuclei, non-tumor nuclei,
            tubule, and non-tubule arranged in separate subfolders",
        "Dataset Path": "histopathology/BreCaHAD"
    },
    {
        "Dataset Name": "melanoma",
        "Dataset Description": "This dataset is a melanoma specific
            dataset with nuclei and tissue annotations along with
            original images in separate subfolders.",
        "Dataset Path": "histopathology/melanoma"
    },
    {
        "Dataset Name": "choledoch",
        "Dataset Description": "This is a database for both microscopy
            hyperspectral and color images of cholangiocarcinoma,
            including 880 scenes among which 689 scenes are samples with
```

```
            part of cancer areas (L), 49 scenes full of cancer areas (N),
             and 142 scenes without cancer areas (P) organized as
            separate subfolders",
        "Dataset Path": "histopathology/choledoch"
    },
    {
        "Dataset Name": "histopath-sn",
        "Dataset Description": "This is a Kaggle dataset, with the task
            to classify patches: Bronchus and lung samples in image
            folder along with labels in separate csv file.",
        "Dataset Path": "histopathology/histopath-sn"
    },
    {
        "Dataset Name": "ULMS",
        "Dataset Description": "Uterine leiomyosarcoma (ULMS) dataset
            comprises mitosis count, necrosis, and nuclear atypia with
            labels in separate csv file",
        "Dataset Path": "histopathology/ULMS"

    },
    {
        "Dataset Name": "MonuSeg",
        "Dataset Description": "The dataset comprises nuclei from seven
            organs with associated annotations in csv file.",
        "Dataset Path": "histopathology/MonuSeg"
    },
    {
        "Dataset Name": "kmc_kidney",
        "Dataset Description": "The introduced KMC kidney histopathology
            dataset includes non-cancerous (Grade-0) and cancerous (Grade
            -1 to Grade-4) images of the Renal Clear Cell Carcinoma
            organized as separate subfolders",
        "Dataset Path": "histopathology/kmc_kidney"
    },
    {
        "Dataset Name": "histo-img-text",
        "Dataset Description": "This is a kaggle dataset with
            histopathology image-text pairs",
        "Dataset Path": "histopathology/histo-img-text"
    },
    {
        "Dataset Name": "cellnet",
        "Dataset Description": "CellNet is a meticulously curated dataset
             featuring over 120,000 high-quality medical images
            representing over 20 organ/cancer classes arranged as
            different subfolders. ",
        "Dataset Path": "histopathology/cellnet"
    },
    {
        "Dataset Name": "PanNuke",
        "Dataset Description": "Nuclei instance segmentation and
            classification dataset with exhaustive nuclei labels across
            19 different tissue types. In total the dataset contains
            205,343 labeled nuclei, each with an instance segmentation
            mask in separate datasets.",
        "Dataset Path": "histopathology/PanNuke"
    },
    {
        "Dataset Name": "NPC-88k-Public",
        "Dataset Description": "88k histopathology patches of normal,
            lymphoid hyperplasia (LHP), nasopharyngeal inflammation (NPI)
            , and nasopharyngeal carcinoma (NPC) organized in separate
            subfolders.",
        "Dataset Path": "histopathology/NPC-88k-Public"
    },
```

```
    {
        "Dataset Name": "EBHI",
        "Dataset
            Description": "The dataset encompasses various categories,
            including normal (76 images and 76 ground truth images),
            polyp (474 images and 474 ground truth images), low-grade
            intraepithelial neoplasia (639 images and 639 ground truth
            images), high-grade intraepithelial neoplasia (186 images and
             186 ground truth images), serrated adenoma (58 images and 58
             ground truth images), and adenocarcinoma (795 images and 795
             ground truth images) arranged in different subfolders",
        "Dataset Path": "histopathology/EBHI"
    }
]
```

C.3 Detecting and Addressing Data Quality Issues for Data Pre-Processing Agent

One of the primary steps in data pre-processing involves identifying data quality issues and removing samples that negatively impact the overall data quality. In this work, we address three key data quality issues *viz.* **off-topic samples**, **near duplicates**, and **label errors** (Gröger et al., 2025; 2024; 2023) each of which can significantly compromise the reliability of machine learning models, particularly in sensitive domains like medical imaging.

- **Off-topic samples** refer to irrelevant inputs mistakenly included in the dataset (e.g., from unrelated modalities or corrupted acquisitions). These introduce noise, distort evaluation metrics, and hinder model convergence.
- **Near duplicates** are different views of the same object, including exact copies. Their presence artificially reduces the diversity of the training set, introduces redundancy, and may lead to data leakage between training and evaluation sets.
- **Label errors** are incorrectly annotated examples that can misguide both model training and evaluation, leading to degraded performance and spurious generalization.

The dataset is formalized as $\mathcal{X} = \{(x_i, l_i) \mid i \in \mathcal{I}\}$, where each x_i is a sample, l_i is its label among L classes, and $\mathcal{I} = \{1, \ldots, N\}$ the index set. For each issue type, a scoring function $s(\cdot)$ is defined that maps individual samples or sample pairs to a score in $[0, 1]$, where lower values indicate higher likelihood of an issue. Ranking the samples by these scores yields a prioritized list for inspection or automated filtering based on a pre-defined threshold.

Representation Learning

A deep feature extractor $f(\cdot; \theta)$ was trained using self-supervised learning (SSL) methods (*SimCLR* or *DINO*), both of which were implemented with a Vision Transformer (ViT) backbone. Each sample x_i was embedded into a latent space as $e_i = f(x_i; \theta) \in \mathbb{R}^D$, where D denotes the feature dimension. To ensure consistent geometry across methods, ℓ_2-normalization was applied so that all embeddings lie on a unit hypersphere.

Cosine similarity was adopted to define the distance metric:

$$\text{sim}(e_i, e_j) = \frac{e_i^\top e_j}{\|e_i\|_2 \|e_j\|_2}, \quad \text{dist}(e_i, e_j) = \frac{1 - \text{sim}(e_i, e_j)}{2}.$$

Issue-Specific Detection Strategies

Off-topic Detection. Off-topic samples were identified using agglomerative clustering with single linkage in the representation space. The merging behavior of clusters was analyzed, and samples that were merged at higher distances or at later stages with larger clusters were considered more likely to be anomalous. A scoring function $s_{\text{OT}}(e_i)$ was constructed based on merge depth and inter-cluster distance dynamics.

Near Duplicate Detection. Candidate near-duplicate pairs were detected by evaluating pairwise distances between all sample embeddings. A simple ranking function was applied:

$$s_{\text{ND}}(e_i, e_j) = \text{dist}(e_i, e_j),$$

where smaller distances were interpreted as a higher likelihood of duplication.

Label Error Detection. Label errors were inferred based on a ratio between intra-class and inter-class distances. For each sample e_i, the following definitions were used:

$$m_{=}(e_i) = \min_{j \in \mathcal{I},\ l_j = l_i} \text{dist}(e_i, e_j), \quad m_{\neq}(e_i) = \min_{j \in \mathcal{I},\ l_j \neq l_i} \text{dist}(e_i, e_j),$$

$$s_{\text{LE}}(e_i) = \frac{m_{\neq}^2(e_i)}{m_{=}^2(e_i) + m_{\neq}^2(e_i)}.$$

Lower scores were interpreted as indicating a higher likelihood of mislabeling, particularly when the nearest neighbor belonged to a different class.

In all three cases, the local structure of the embedding space was leveraged by the cleaning function used in Tool 9 of the Listing 1. Cluster distances were evaluated using only the nearest neighbors for off-topic detection, proximity among sample pairs was assessed for duplicate identification, and comparative distances to same- and different-class neighbors were exploited to detect label errors.

C.4 Collection of Federated Learning algorithms

Federated Learning (FL) has evolved significantly beyond its initial formulation of model averaging, with numerous algorithmic innovations developed to address practical challenges such as data heterogeneity, personalization, privacy preservation, and limited client resources (McMahan et al., 2017; Tan & Wang; Tan et al., 2023). In this work, we utilize a set of **40** key federated learning (FL) algorithms, covering core, personalized, generalizable, and adaptive methods, as summarized in Tables 2-4. The algorithm description required by server-based federated training agents for FL algorithm selection is provided in Listing 9.

The selected algorithms reflect the diversity and progression of research in FL across three main axes:

1. **Foundational and General-Purpose Methods:**
   We begin with core algorithms such as *FedAvg*, *FedAvgM*, and *FedProx*, which establish the baseline principles of client-server aggregation and account for statistical and system heterogeneity. These methods are essential for benchmarking and provide the backbone upon which many subsequent algorithms are built.
2. **Personalization-Oriented Methods:**
   Recognizing the need to adapt to non-IID data across clients, we include algorithms like *FedRep*, *FedPer*, *Ditto*, *pFedHN*, and *Per-FedAvg*. These approaches personalize part of the model (e.g., classifier heads or entire layers), use meta-learning, or leverage client-specific adaptation strategies. Methods such as *pFedMe* and *FedEM* extend this personalization through bi-level optimization and mixture modeling, respectively.
3. **Robustness, Adaptivity, and Generalization:**
   To tackle challenges of out-of-distribution generalization and domain shifts, we incorporate algorithms like *FedIIR*, *FedSR*, and *ADCOL*, which emphasize invariant representation learning and adversarial feature alignment. Techniques such as *FedDyn*, *FedFomo*, and *FedRoD* introduce dynamic regularization and adaptive weighting to stabilize optimization in heterogeneous environments. Moreover, algorithms like *FedBN* and *FedAP* address domain-specific normalization challenges, particularly in healthcare contexts.
4. **Emerging and Specialized Directions:**
   The inclusion of recent methods such as *Floco*, *FedAS*, and *PeFLL* highlights advancements in adaptive aggregation, inter-client relationship modeling, and meta-learned personalization. Additionally, *MOON*, *FedGen*, and *CCVR* represent innovative uses of contrastive learning, data-free distillation, and virtual representation calibration.

The rationale for selecting this curated list is threefold:

- **Comprehensiveness:** The algorithms span from classic to state-of-the-art methods, ensuring broad coverage of the field.
- **Modular Design Potential:** These algorithms are suitable for integration into modular federated learning pipelines, facilitating agent-based automation and tool invocation.
- **Relevance to Real-World Scenarios:** Many chosen methods address constraints encountered in practical deployments, including label imbalance, resource limitations, domain adaptation, and personalization needs.

This comprehensive collection enables systematic benchmarking, comparative evaluation, and modular composition in our federated learning framework *FedAgentBench*. Each method contributes unique strengths and trade-offs, making them valuable candidates for real-world and research applications.

Listing 9: Federated Learning Algorithm Descriptions for Server-based algorithm selector agents

```
[
    {
        "algorithm": "FedAvg",
        ""description"": "The foundational algorithm in federated
            learning, where clients perform multiple steps of local
            stochastic gradient descent (SGD) and periodically average
            their models on a central server. It is simple and
            communication-efficient but struggles with non-IID data
            distributions."
    },
    {
        "algorithm": "FedAvgM",
        ""description"": "An extension of FedAvg that integrates server-
            side momentum during model aggregation. This is a classical
            federated learning approach that stabilizes training and
            improves convergence in the presence of data heterogeneity
            across clients."
    },
    {
        "algorithm": "FedProx",
        ""description"": "Classical federated learning algorithm that
            enhances FedAvg by adding a proximal term to the local
            objective functions, discouraging local updates from drifting
             too far from the global model. This addresses system and
            statistical heterogeneity among clients."
    },
    {
        "algorithm": "SCAFFOLD",
        ""description"": "Classical federated learning algorithm that
            incorporates control variates to correct client-drift caused
            by non-IID data. Each client maintains local control
            variables to align updates with the global objective,
            improving convergence stability."
    },
    {
        "algorithm": "MOON",
        ""description"": "Traditional Federated learning algorithm that
            implements model-level contrastive learning by aligning
            current local models with the global model while contrasting
            them with past local models. This enhances representation
            learning under non-IID settings."
    },
    {
        "algorithm": "FedDyn",
        ""description"": "Regularization-based federated learning
            approach that introduces a dynamic regularization term into
            local objectives that evolves over time to better match the
            global objective. This mechanism helps mitigate divergence
            and stabilizes training in heterogeneous environments."
    },
```

Table 4: Overview of Federated Learning Algorithms (Part 1)

Method	Source	Key Idea	Strengths	Limitations
FedAvg (McMahan et al., 2017)	McMahan et al., 2016	Clients perform local SGD and periodically average with the server.	Simple and communication-efficient.	Degrades with non-IID data due to client drift.
FedAvgM (Hsu et al., 2019)	Hsu et al., 2019	Adds server-side momentum to FedAvg.	Improves convergence on non-IID data.	Requires careful momentum tuning.
FedMD (Li & Wang, 2019)	Li et al., NeurIPS 2019	Uses public dataset for knowledge distillation across heterogeneous models.	Supports diverse architectures.	Requires public dataset.
FedPer (Arivazhagan et al., 2019)	Arivazhagan et al., arXiv 2019	Uses client-specific layers with shared global layers.	Balances global and local learning.	Designing layer split is non-trivial.
LG-FedAvg (Liang et al., 2020)	Liang et al., NeurIPS 2019 Workshop	Aggregates global layers, retains local ones.	Preserves local personalization.	Complex model synchronization.
CFL (Sattler et al., 2019)	Sattler et al., arXiv 2019	Clusters clients and trains separate models.	Addresses data heterogeneity.	Doesn't scale well with many clusters.
FedProx (Li et al., 2020b)	Li et al., 2020	Adds proximal term to local loss.	Handles statistical/system heterogeneity.	May slow down convergence.
SCAFFOLD (Karimireddy et al., 2020)	Karimireddy et al., 2020	Uses control variates to correct drift.	Better convergence on non-IID data.	Extra storage and computation.
APFL (Deng et al., 2020)	Deng et al., arXiv 2020	Adaptive mixing of global and local models.	Combines generalization and personalization.	Requires careful mixing parameter tuning.
Per-FedAvg (Fallah et al., 2020)	Fallah et al., NeurIPS 2020	Combines FL with MAML.	Enables fast personalization.	Needs second-order gradients.
pFedMe (Dinh et al., 2022)	Dinh et al., NeurIPS 2020	Uses Moreau envelopes for bi-level optimization.	Fast convergence and good personalization.	Requires tuning of regularization.
MOON (Li et al., 2021a)	Li et al., CVPR 2021	Aligns local and global models via contrastive loss.	Strong representation learning.	Needs previous model storage.
FedDyn (Acar et al., 2021)	Acar et al., ICLR 2021	Dynamic regularization to align objectives.	Mitigates client drift.	More complex optimization.
FedGen (Zhu et al., 2021)	Zhu et al., ICML 2021	Uses synthetic data for knowledge distillation.	Enables data-free generalization.	Depends on generator quality.
FedOpt (Reddi et al., 2021)	Reddi et al., ICLR 2021	Uses adaptive optimizers (Adam/Yogi) in FL.	Fast/stable convergence.	Hyperparameter tuning required.

Table 5: Overview of Federated Learning Algorithms (Part 2)

Method	Source	Key Idea	Strengths	Limitations
CCVR (Luo et al., 2021)	Wang et al., NeurIPS 2021	Virtual representations for calibration.	No real data sharing needed.	Relies on distribution approximations.
FedEM (Marfoq et al., 2022)	Marfoq et al., NeurIPS 2021	Mixture model for multi-task personalization.	Captures cross-client distributions.	Assumes shared latent structure.
Ditto (Li et al., 2021c)	Li et al., ICML 2021	Maintains global and personalized models.	Robust and fair personalization.	Needs dual model training.
FedRep (Collins et al., 2023)	Collins et al., ICML 2021	Shared encoder with local classifiers.	Combines global and local strengths.	Coordination needed for shared layer.
pFedHN (Shamsian et al., 2021)	Shamsian et al., ICML 2021	Hypernetworks generate personalized models.	Communication efficient.	Complex hypernetwork training.
FedFomo (Zhang et al., 2021)	Zhang et al., ICLR 2021	Aggregates based on client similarity.	Personalization without raw data.	Similarity computation overhead.
FedBN (Li et al., 2021d)	Li et al., ICLR 2021	Local BN layers for domain adaptation.	Improves performance on non-IID data.	No global BN normalization.
FedLC (Zhang et al., 2022)	Zhang et al., ICML 2022	Logits calibration to handle label skew.	Effective on imbalanced datasets.	Needs label distribution estimation.
MetaFed (Chen et al., 2023b)	IJCAI 2022	Cyclic knowledge distillation across federations.	Enhances collaboration.	Federation coordination required.
FedRoD (Chen & Chao, 2022)	ICLR 2022	Adaptive aggregation for balancing general/personal models.	Personalized and generalizable.	May fail under high heterogeneity.
FedProto (Tan et al., 2022)	AAAI 2022	Prototype-based feature alignment.	Preserves global semantics.	Quality depends on prototypes.
pFedLA (Ma et al., 2022)	Ma et al., CVPR 2022	Layer-wise model aggregation.	Fine-grained personalization.	Management complexity.
FedBABU (Oh et al., 2022)	Oh et al., ICLR 2022	Aggregates body and keeps local heads.	Improves representation learning.	Less consistent predictions.
FedAP (Lu et al., 2022)	Chen et al., IEEE 2022	Adaptive BN for healthcare FL.	Handles domain shift.	Sensitive to BN statistics.

Table 6: Overview of Federated Learning Algorithms (Part 3)

Method	Source	Key Idea	Strengths	Limitations
FedSR (Nguyen et al., 2022a)	NeurIPS 2022	Domain generalization via representation regularization.	Lightweight and simple.	May fail in extreme domain shift.
FedALA (Zhang et al., 2023)	AAAI 2023	Adaptive local aggregation weights.	Relevance-aware updates.	Unstable weight estimation.
FedFed (Yang et al., 2023)	Yang et al., NeurIPS 2023	Distills critical features.	Improves generalization.	Needs good feature selection.
Elastic Aggregation (Chen et al., 2023a)	Chen et al., CVPR 2023	Sensitivity-based update weighting.	Balances adaptation/stability.	Adds computation.
ADCOL (Li et al., 2023b)	ICML 2023	Adversarial alignment of features.	Handles domain shift.	Adversarial training instability.
FedIIR (Guo et al., 2023)	ICML 2023	Learns invariant relationships for OOD generalization.	Strong generalization.	Needs assumptions on invariance.
pFedSim (Tan et al., 2023)	Tan et al., arXiv 2023	Similarity-based aggregation.	Enables personalization.	Hard to measure similarity.
PeFLL (Scott et al., 2025)	ICLR 2024	Meta-learns to personalize clients.	Fast client adaptation.	High computation cost.
FLUTE (Liu et al., 2024a)	ICML 2024	Efficient rep learning under underparameterization.	Resource efficient.	May sacrifice expressivity.
FedAS (Yang et al., 2024)	CVPR 2024	Reduces global-local inconsistency.	More consistent updates.	More complex training.
Floco (Grinwald et al., 2025)	NeurIPS 2024	Uses connected modes to model clients.	Leverages inter-client structure.	Needs client connectivity info.

Table 7: Categorization of FL Algorithms

Category	Algorithms
(i) Classical FL algorithms	FedAvg, FedAvgM, FedProx, SCAFFOLD, MOON, FedLC
(ii) Personalized FL algorithms	Per-FedAvg, pFedMe, FedRep, FedPer, FedBN, pFedLA, pFedHN, FedFomo, LG-FedAvg, APFL, FedEM, pFedSim, FedBABU, CCVR
(iii) Regularization-based approaches	Ditto, FedDyn, FedRoD, FedAS, SCAFFOLD, pFedMe
(iv) Knowledge Distillation-based methods	FedGen, FedMD, FedFed, MetaFed
(v) Domain generalization techniques	FedSR, FedIIR, ADCOL, FedProto, FedAP
(vi) Optimization and scheduling variants	FedOpt, FedAvgM, FedALA, Elastic Aggregation, FLUTE, PeFLL, CFL


```
    {
        "algorithm": "FedLC",
        ""description"": "Classical federated learning algorithm that
            applies logits calibration techniques during local training
            to address label distribution skew. This helps balance
            prediction confidence and improve accuracy on imbalanced or
            non-IID datasets."
    },
    {
        "algorithm": "FedGen",
        ""description"": "Personalized Federated Learning leveraging
            knowledge distillation that uses a server-side generative
            model to synthesize data representations for knowledge
            distillation, enabling model personalization without
            requiring access to client data. This preserves privacy while
             supporting generalization."
    },
    {
        "algorithm": "CCVR",
        ""description"": "Personalized Federated Learning that uses
            virtual representations drawn from approximated data
            distributions to calibrate classifiers. This approach
            improves generalization in non-IID scenarios without needing
            to exchange actual data between clients."
    },
    {
        "algorithm": "FedOpt",
        ""description"": "Federated adaptive optimization scheme that
            extends FedAvg by integrating adaptive gradient methods like
            FedAdam, FedYogi, and FedAdagrad, which dynamically adjust
            learning rates and enhance convergence performance in diverse
             federated settings."
    },
    {
        "algorithm": "Elastic Aggregation",
        ""description"": "Classical federated optimization scheme that
            introduces elasticity in the aggregation process by assigning
             dynamic weights to client updates based on the sensitivity
            of model parameters. This balances stability and adaptability
            , improving performance on heterogeneous datasets."
    },
    {
        "algorithm": "FedFed",
        ""description"": "Federated learning algorithms that allows
            partial feature sharing between clients and server and
```

```
            mitigates data heterogeneity by distinguishing between
            performance-sensitive and performance-robust features and
            selectively distilling the former. This allows clients to
            retain useful features while benefiting from cross-client
            generalization."
    },
    {
        "algorithm": "pFedSim",
        ""description"": "Personalized Federated Learning Algorithm that
            enhances personalization by aggregating client models based
            on the similarity of their data distributions. Clients with
            more similar data contribute more significantly to each other
            's updates, enabling customized learning without explicit
            data sharing."
    },
    {
        "algorithm": "FedMD",
        ""description"": "Personalized Federated Learning Algorithm that
            supports clients with heterogeneous architectures by
            performing knowledge distillation using a shared public
            dataset. Clients align on output predictions rather than
            model parameters, enabling collaborative training without
            requiring architectural uniformity."
    },
    {
        "algorithm": "APFL",
        ""description"": "Personalized Federated Learning Algorithm that
            implements an adaptive mixing strategy where each client
            maintains both a local and a global model. The final model
            output is a weighted combination, and the mixing coefficient
            is learned during training to achieve optimal personalization
            ."
    },
    {
        "algorithm": "LG-FedAvg",
        ""description"": "Personalized Federated Learning Algorithm that
            decomposes models into local and global components, where
            only the global part is aggregated across clients. This
            preserves local knowledge while benefiting from global trends
            , supporting personalized learning in non-IID settings."
    },
    {
        "algorithm": "FedBN",
        ""description"": "Personalized Federated Learning Algorithm that
            keeps batch normalization layers local to each client while
            sharing the rest of the model globally. This enables
            adaptation to client-specific feature distributions and
            enhances performance under feature heterogeneity."
    },
    {
        "algorithm": "FedPer",
        ""description"": "Personalized Federated Learning Algorithm that
            introduces personalization by partitioning the model into a
            globally shared base and a locally updated head. This
            structure allows clients to fine-tune their models based on
            local data while retaining shared representations."
    },
    {
        "algorithm": "FedRep",
        ""description"": "Personalized Federated Learning Algorithm that
            learns a common feature extractor shared across clients and
            allows each client to train its own classifier head. This
            separation supports personalization without requiring full
            model updates across the federation."
    },
```

```
    {
        "algorithm": "Per-FedAvg",
        ""description"": "Personalized Federated Learning Algorithm that
            combines meta-learning (specifically MAML) with federated
            learning to learn a global initialization that can be rapidly
             personalized to each clients local data, enabling quick
            adaptation with limited samples."
    },
    {
        "algorithm": "pFedMe",
        ""description"": "Personalized Federated Learning Algorithm that
            formulates personalized federated learning as a bi-level
            optimization problem using Moreau envelopes, which allows
            decoupling global and local updates. This improves
            convergence and supports better personalization."
    },
    {
        "algorithm": "FedEM",
        ""description"": "Personalized Federated Learning Algorithm that
            performs multi-task learning. It treats each clients model
             as part of a mixture of distributions and trains them via
            the Expectation-Maximization algorithm. This enables multi-
            task personalization by modeling shared and unique components
             across clients."
    },
{
        "algorithm": "Ditto",
        ""description"": "Personalized Federated Learning Algorithm that
            simultaneously trains a global model for generalization and a
             personalized model for each client, ensuring fairness and
            robustness through dual-objective optimization."
    },
    {
        "algorithm": "pFedHN",
        ""description"": "Personalized Federated Learning Algorithm that
            utilizes a central hypernetwork that generates personalized
            model weights for clients, enabling parameter sharing while
            allowing client-specific adaptations."
    },
    {
        "algorithm": "pFedLA",
        ""description"": "Personalized Federated Learning Algorithm that
            performs layer-wise model aggregation, assigning personalized
             importance to each layer across clients to improve fine-
            grained adaptation in non-IID environments."
    },
    {
        "algorithm": "CFL",
        ""description"": "Federated Learning algorithm that clusters
            clients based on model or data similarity and trains distinct
             models per cluster to effectively manage heterogeneity
            across groups."
    },
    {
        "algorithm": "FedFomo",
        ""description"": "Personalized Federated Learning Algorithm that
            maintains a personalized model by aggregating updates from
            peer clients weighted by similarity scores, using a first-
            order gradient approximation to ensure communication
            efficiency."
    },
    {
        "algorithm": "FedBabu",
        ""description"": "Personalized Federated Learning Algorithm that
            improves personalized learning by aggregating only the shared
```

```
            body (feature extractor) of the model while keeping client-
            specific heads independent."
    },
    {
        "algorithm": "FedAP",
        ""description"": "Personalized Federated Learning Algorithm that
            employs adaptive batch normalization to tailor models to
            healthcare clients, effectively handling distribution shifts
            across medical institutions."
    },
    {
        "algorithm": "MetaFed",
        ""description"": "Personalized Federated Learning Algorithm that
            applies a cyclic knowledge distillation framework across
            federated groups, improving model generalizability without
            raw data exchange and without necessity of a server."
    },
    {
        "algorithm": "FedRoD",
        ""description"": "Regularization-based Federated Learning
            approach that balances the benefits of generalization and
            personalization by adaptively mixing global and local model
            components using regularized dual objectives."
    },
    {
        "algorithm": "FedProto",
        ""description"": "Personalized and generalizable Federated
            learning algorithm that aligns client features through the
            use of global class prototypes, promoting semantic
            consistency while preserving personalization."
    },
    {
        "algorithm": "FedALA",
        ""description"": "Personalized Federated learning algorithm that
            aggregates local models adaptively by learning relevance-
            based weights for each client, enabling better
            personalization through dynamic influence modeling."
    },
    {
        "algorithm": "PeFLL",
        ""description"": "Personalized Federated learning algorithm that
            incorporates meta-learning to personalize model updates for
            each client by learning an optimal initialization that
            generalizes quickly to local tasks."
    },
    {
        "algorithm": "FLUTE",
        ""description"": "Personalized Federated learning algorithm that
            addresses model underparameterization in resource-constrained
             environments by learning efficient global and local decoders
             for distributed representation learning."
    },
    {
        "algorithm": "FedAS",
        ""description"": "Personalized Federated learning algorithm using
             regularization-based approach that aligns global and local
            model updates using adaptive strategies to reduce
            inconsistency and improve convergence in personalized
            federated learning."
    },
    {
        "algorithm": "Floco",
        ""description"": "Personalized Federated learning algorithm that
            models client relationships using a graph of local modes and
```

```
            clusters them for collaborative training, leveraging shared
            structure without central data."
    },
    {
        "algorithm": "FedSR",
        ""description"": "Federated domain generalization-based technique
             that applies simple regularization across domain
            representations to improve out-of-distribution generalization
             in federated settings."
    },
    {
        "algorithm": "ADCOL",
        ""description"": "Federated domain generalization-based technique
             that uses adversarial learning to align feature spaces
            across clients, enabling domain generalization under non-IID
            conditions."
    },
    {
        "algorithm": "FedIIR",
        ""description"": "Federated domain generalization-based technique
             that identifies and leverages invariant relationships across
             domains to enhance generalization to out-of-distribution
            data in federated settings."
    }
]
```

C.5 LLMs AS THE AGENT CORE COMPONENTS

MODEL SELECTION JUSTIFICATION

To assess the reasoning, planning, and tool-use capabilities of large language model (LLM) agents in the context of real-world federated learning workflows, we evaluate a set of 24 LLMs on the FedAgentBench suite. The selected models span both proprietary and open-source categories, ensuring broad coverage across scale, training data diversity, and model access paradigms.

We include 10 proprietary LLMs from leading industrial labs such as OpenAI and Anthropic, including multiple variants of GPT-4. These models represent the current frontier of general-purpose foundation models, often topping benchmarks in instruction-following, tool use, and reasoning. Their inclusion allows us to benchmark state-of-the-art commercial performance in the agentic FL setting.

We particularly include a range of GPT-family models developed by OpenAI to cover both ends of the performance-efficiency spectrum in proprietary large language models (LLMs). The rationale is threefold:

(i) **Proven Instruction-Following and Reasoning Abilities:**
GPT-4 and its variants have consistently demonstrated state-of-the-art performance across multiple benchmarks involving instruction following, task decomposition, and multi-step reasoning capabilities essential for evaluating LLM agents in complex federated learning pipelines such as FedAgentBench.

(ii) **Variants across Performance Tiers and Costs:**
The selection spans high-end models (e.g., GPT-4.1, GPT-4o) and lightweight alternatives (e.g., GPT-4.1-mini, GPT-o3-mini). This allows us to study the trade-offs between agent reasoning quality and computational/resource efficiency, particularly relevant for real-world FL deployment where cost and inference speed matter.

(iii) **Industry Adoption and API Availability:**
These models are widely adopted in both academic and industrial applications and offer stable, reproducible APIs. This ensures consistent evaluation and compatibility with tool-augmented LLM agent frameworks.

Besides, we evaluate 14 open-source LLMs across four major families: LLaMA, DeepSeek, Qwen, and Gemma. These models are chosen for their state-of-the-art performance in open benchmarks, availability in multiple parameter scales (from 9B to 685B), and varying architectural innovations

(e.g., distillation in DeepSeek, instruction tuning in Qwen, and scalability in Gemma). This selection ensures a representative spectrum of recent advances in open-source LLM development, and provides insight into how scale, family, and fine-tuning affect FL-agent performance.

By including both proprietary and open models across diverse sizes and pretraining paradigms, our evaluation is designed to offer fair, scalable, and realistic comparisons, while informing the community of strengths and limitations across model categories in complex multi-agent settings like FedAgentBench.

Table 8: Descriptions for Proprietary LLMs in FedAgentBench

Model	Description	Capabilities	Use Rationale	Caveats / Notes
GPT-4.1	Latest high-performance model from OpenAI with advanced reasoning and planning.	Chain-of-thought reasoning, tool use, structured outputs.	Reference proprietary agent for end-to-end workflows.	High cost and latency; not ideal for real-time execution.
GPT-4o	Multimodal flagship model supporting vision-language tasks.	Multilingual, tool calling, multimodal reasoning.	Evaluated for vision + tool scenarios.	New model; some outputs may vary between calls.
GPT-4	Original GPT-4 model with top-tier generalization.	Long-context, reasoning, structured outputs.	Used as baseline for reasoning accuracy.	Slower than turbo and newer variants.
GPT-4-Turbo	Faster and cheaper version of GPT-4 for API use.	Efficient inference, similar capabilities to GPT-4.	Preferred when cost is a concern.	Slightly less coherent outputs.
GPT-4.1-mini	Distilled variant optimized for fast inference.	Good single-step logic, mid-range planning.	Used in real-time assistant agents.	Weaker on edge-case and ambiguous tasks.
GPT-4o-mini	Smaller variant of GPT-4o with multimodal support.	Vision-language support, low-latency.	Benchmarked in low-resource multimodal agents.	Reduced performance in logic-intensive tasks.
GPT-o4-mini	Lightweight GPT-4 style model.	Text generation and simple instructions.	Ablation studies for low-cost GPT agents.	Unclear origin; may alias other mini variants.
GPT-o3-mini	GPT-3.5-based efficient variant.	Very fast, single-turn chat.	Used for comparison with older architectures.	Weak reasoning; not reliable for planning.
GPT-3.5 Turbo	Predecessor to GPT-4, cheaper and widely used.	Fast, capable for basic instruction and QA.	Low-cost reference for proprietary agents.	Token alignment issues in structured tasks.
Claude-3.7 Sonnet	Mid-size model from Anthropic with alignment tuning.	Safety-aligned generation, multilingual, tool use.	Benchmarked against non-OpenAI proprietary model.	Slightly lower fluency than top Claude variants.

D Results and Discussions

We conducted extensive evaluations of both proprietary and open-source LLM agents across 6 environments, out of which the success rates for Histopathology have been mentioned in the main paper. The success rates for the remaining 5 environments *viz*., Dermatology, Ultrasound, MRI, Fundus and X-Ray environments are reported here. The results of these experiments are presented in Tables 10-15. These tables capture performance under two paradigms: fine-grained multi-step guidance and goal-oriented single-shot instruction, revealing consistent trends across modalities. Notably, the independent script generation setting in Table 12 illustrates a sharp decline in performance for most agents, underscoring the challenges of long-horizon task planning without explicit decomposition. Overall Time-requirement metrics for task resolution are summarized in Table16, providing a holistic view of capability and practicality across LLM variants. Figs 10-36 show snippets of different phases of the FL workflow with various LLMs and different imaging modalities which help to understand their success and failure modes.

D.1 Discussion on agentic performance in individual healthcare environment

The overall comparative agentic performance in all environments has been summarized in Table 17. Furthermore, we also analyze the performance of individual environments. Table 10 reports the performance of open-source and proprietary LLM agents in the **Dermatology environment**. Proprietary models obtain the strongest results under both guidance regimes. **GPT-4.1** is the highest-performing system, achieving consistent $5/5$ scores on most sub-tasks and the highest **Overall** performance (**94.29** with fine-grained guidance; **88.57** with goal-oriented guidance).

Table 9: Descriptions for Open-Source LLMs in FedAgentBench

Model	Description	Capabilities	Use Rationale	Caveats / Notes
LLaMA-4 Maverick	Latest LLaMA release (2025) with top-tier accuracy in reasoning and instruction following.	Instruction following, long-context reasoning, coding tasks.	Used for evaluating high-end open-source agents.	Resource heavy; slower than lighter LLaMA variants.
LLaMA-4 Scout	2025 LLaMA-4 variant optimized for cost-efficient inference.	Balanced reasoning and fast response for system agents.	Used as mid-range open-source agent in system and logic tasks.	Less expressive than Maverick.
LLaMA-3 70B	Flagship LLaMA model (2024) with extensive instruction tuning.	Reasoning, multilingual tasks, tool use.	Used for top-tier open-source evaluation.	Less performant than newer LLaMA-4 variants.
LLaMA-3 8B	Smaller variant of LLaMA-3 for constrained environments.	General understanding, good for fast responses.	Used in real-time benchmarking of lighter agents.	Limited capacity in multi-hop reasoning.
DeepSeek-V3	Latest release from DeepSeek with strong Chinese-English capability.	Multilingual chat, code, reasoning.	Used to test multilingual and cross-domain agents.	Less stable tool usage.
DeepSeek-R1	General purpose 2024 DeepSeek model.	Basic LLM tasks, reasoning.	Baseline open-source reference.	Lower precision under stress tests.
DeepSeek-R1-Distill	Distilled version of DeepSeek-R1 on LLaMA-70B.	Fast inference, low-resource usage.	Used in lightweight evaluations.	Lower performance ceiling.
Qwen 3 235B	Massive MoE model by Alibaba; high capacity and strong multilingual.	Multilingual, few-shot generalization, long context.	Benchmarked as high-capacity open-source agent.	Costly to run, sparse documentation.
Qwen QwQ 32B	Intermediate-sized multilingual Qwen model.	Instruction following, QA, multilingual chat.	Used as cost-performance mid-range Qwen agent.	Less stable tool usage.
Qwen 3 30B	Well-balanced Qwen variant.	Reliable output, structured reasoning.	Used in systems requiring stable decoding.	Reduced multilingual coverage vs 235B.
Qwen 3 14B	Smaller Qwen for lightweight use.	Quick single-turn tasks.	Used in sub-agents and pre-filtering roles.	Shallow reasoning, poor long-context.
Gemma 3 27B Instruct	Instruction-tuned model by Google.	Tool use, summarization, chat.	Tested for logic tasks.	Less capable in multi-modal domains.
Gemma 3 12B Instruct	Smaller Gemma variant.	Common NLP tasks.	System-level fast agent.	May misfire structured outputs.
Gemma 2 9B Instruct	Previous generation Gemma model.	Lightweight inference.	Tested in low-cost agent scenarios.	Lowest instruction accuracy among Gemmas.

Table 10: Comparison of open-source and proprietary LLM agents across different stages of federated learning: Client Selection (Client-Sel), Data Pre-processing (Data-Pre), Label Harmonization (Label-Harm), and Federated Training (Fed-Train) in **Dermatology** environment based on skin cancer detection task. a/b refers to the proportion of successful runs 'a' out of the total number of runs 'b'

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
	a_1, a_2, a_3	a_4	a_5	a_6, a_7		a_1, a_2, a_3	a_4	a_5	a_6, a_7	
Proprietary Models										
GPT-4.1	5/5, 5/5, 5/5	5/5	3/5	5/5, 5/5	94.29	5/5, 4/5, 5/5	5/5	3/5	4/5, 5/5	88.57
GPT-4o	5/5, 3/5, 5/5	5/5	1/5	1/5, 5/5	71.43	5/5, 1/5, 5/5	5/5	1/5	1/5, 5/5	65.71
GPT-4	5/5, 4/5, 5/5	0/5	1/5	3/5, 5/5	65.71	5/5, 1/5, 5/5	0/5	0/5	2/5, 5/5	51.43
GPT-4-Turbo	5/5, 3/5, 5/5	2/5	1/5	3/5, 5/5	68.57	5/5, 3/5, 5/5	5/5	1/5	2/5, 5/5	74.29
GPT-4.1-mini	5/5, 5/5, 5/5	5/5	3/5	3/5, 5/5	88.57	5/5, 5/5, 5/5	3/5	3/5	3/5, 5/5	82.86
GPT-4o-mini	5/5, 1/5, 3/5	5/5	3/5	3/5, 4/5	68.57	5/5, 0/5, 3/5	5/5	1/5	2/5, 4/5	57.14
GPT-o4-mini	5/5, 4/5, 5/5	5/5	3/5	3/5, 5/5	85.71	5/5, 3/5, 5/5	4/5	2/5	3/5, 4/5	74.29
GPT-o3-mini	5/5, 3/5, 5/5	0/5	2/5	3/5, 5/5	65.71	5/5, 1/5, 5/5	0/5	2/5	3/5, 5/5	60.00
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 3/5	31.43
Claude-3-7-Sonnet	5/5, 2/5, 3/5	2/5	1/5	2/5, 3/5	51.42	5/5, 2/5, 3/5	2/5	1/5	2/5, 5/5	57.14
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 1/5, 5/5	5/5	5/5	4/5, 5/5	85.71	5/5, 1/5, 5/5	4/5	4/5	4/5, 5/5	80.00
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.85
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.85
LLaMA-4 Maverick	5/5, 1/5, 4/5	5/5	3/5	2/5, 5/5	71.43	5/5, 1/5, 4/5	5/5	3/5	3/5, 5/5	74.29
LLaMA-4 Scout	5/5, 1/5, 5/5	5/5	3/5	2/5, 5/5	74.29	5/5, 2/5, 5/5	5/5	3/5	2/5, 5/5	77.14
Large Models										
DeepSeek-R1-70B	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-3-70B	5/5, 0/5, 5/5	1/5	1/5	2/5, 5/5	54.29	5/5, 0/5, 5/5	2/5	2/5	1/5, 5/5	57.14
Medium Models										
Qwen QwQ 32B	5/5, 4/5, 5/5	5/5	4/5	4/5, 5/5	91.43	5/5, 4/5, 5/5	5/5	3/5	3/5, 5/5	85.71
Qwen3-30B	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 0/5, 5/5	1/5	1/5	1/5, 5/5	51.43	5/5, 0/5, 5/5	1/5	1/5	1/5, 5/5	51.43
LLaMA-3-8B	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71
Qwen-3-14B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 4/5	40.00
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

Table 11: Comparison of open-source and Proprietary LLM agents in **Ultrasound** environment for **breast cancer detection task**

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
	S_1, C_1, S_2	C_2	C_3	S_3, S_4		S_1, C_1, S_2	C_2	C_3	S_3, S_4	
Proprietary Models										
GPT-4.1	5/5, 3/5, 5/5	5/5	5/5	5/5, 5/5	94.29	5/5, 3/5, 5/5	5/5	5/5	5/5, 5/5	94.29
GPT-4o	5/5, 0/5, 5/5	5/5	3/5	1/5, 5/5	68.57	5/5, 0/5, 5/5	5/5	2/5	1/5, 5/5	65.71
GPT-4	5/5, 3/5, 5/5	1/5	1/5	3/5, 5/5	65.71	5/5, 3/5, 5/5	0/5	1/5	3/5, 5/5	62.86
GPT-4-Turbo	5/5, 3/5, 5/5	1/5	1/5	3/5, 5/5	65.71	5/5, 3/5, 5/5	4/5	1/5	3/5, 5/5	74.29
GPT-4.1-mini	5/5, 3/5, 5/5	5/5	3/5	4/5, 5/5	85.71	5/5, 2/5, 5/5	3/5	4/5	3/5, 5/5	77.14
GPT-4o-mini	5/5, 1/5, 3/5	5/5	3/5	3/5, 5/5	71.43	5/5, 1/5, 3/5	5/5	1/5	5/5, 5/5	71.43
GPT-o4-mini	5/5, 3/5, 5/5	5/5	3/5	4/5, 5/5	85.71	5/5, 3/5, 5/5	4/5	3/5	4/5, 5/5	82.86
GPT-o3-mini	5/5, 2/5, 5/5	1/5	1/5	3/5, 4/5	60.00	5/5, 1/5, 5/5	1/5	2/5	3/5, 5/5	62.86
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 4/5	34.29
Claude-3-7	5/5, 2/5, 3/5	2/5	1/5	3/5, 3/5	54.29	5/5, 2/5, 3/5	2/5	1/5	3/5, 3/5	54.29
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 3/5, 5/5	5/5	5/5	4/5, 5/5	91.43	5/5, 2/5, 5/5	4/5	5/5	4/5, 5/5	85.71
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 3/5, 5/5	5/5	4/5	4/5, 5/5	88.57	5/5, 3/5, 5/5	5/5	3/5	3/5, 5/5	82.86
LLaMA-4 Scout	5/5, 3/5, 5/5	5/5	4/5	3/5, 5/5	85.71	5/5, 1/5, 5/5	5/5	3/5	2/5, 5/5	74.28
Large Models										
DeepSeek-R1-70B	5/5, 3/5, 5/5	3/5	1/5	2/5, 5/5	74.28	5/5, 1/5, 5/5	3/5	0/5	2/5, 5/5	68.57
LLaMA-3-70B	5/5, 3/5, 5/5	2/5	2/5	2/5, 5/5	68.57	5/5, 3/5, 5/5	2/5	2/5	2/5, 5/5	68.57
Medium Models										
Qwen QwQ 32B	5/5, 3/5, 5/5	4/5	4/5	4/5, 5/5	85.71	5/5, 3/5, 5/5	2/5	4/5	4/5, 5/5	80.00
Qwen3-30B	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 0/5, 4/5	2/5	1/5	1/5, 5/5	51.43	5/5, 0/5, 4/5	1/5	1/5	1/5, 5/5	48.57
LLaMA-3-8B	5/5, 0/5, 4/5	4/5	2/5	1/5, 5/5	60.00	5/5, 0/5, 4/5	4/5	2/5	1/5, 5/5	60.00
Qwen-3-14B	5/5, 0/5, 0/5	0/5	0/5	0/5, 5/5	28.57	5/5, 0/5, 0/5	0/5	0/5	0/5, 4/5	25.71
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

A second performance tier includes **GPT-4.1-mini**, **GPT-o4-mini**, **GPT-4o**, **GPT-4-Turbo**, **GPT-o3-mini**, and **GPT-4**. **GPT-3.5-Turbo** shows substantially lower accuracy, and **Claude-3-7-Sonnet** ranks in the middle range.

Table 12: Comparison of open-source and Proprietary LLM agents for **breast cancer detection task** in **Ultrasound** environment on independent script generation for solving individual task.

Model	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
Proprietary Models					
GPT-4.1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-4o	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-4	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-4-Turbo	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-4.1-mini	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-4o-mini	5/5, 0/5, 3/5	0/5	0/5	0/5, 5/5	37.14
GPT-o4-mini	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-o3-mini	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	0/5, 4/5	25.71
Claude-3-7	5/5, 0/5, 3/5	0/5	0/5	0/5, 3/5	31.43
Open-source Models					
Huge Models					
DeepSeek-V3	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Scout	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Large Models					
DeepSeek-R1-70B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-3-70B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Medium Models					
Qwen QwQ 32B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3-30B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models					
Gemma-2-9B	5/5, 0/5, 4/5	0/5	0/5	0/5, 5/5	40.0
LLaMA-3-8B	5/5, 0/5, 4/5	0/5	0/5	0/5, 5/5	40.0
Qwen-3-14B	5/5, 0/5, 0/5	0/5	0/5	0/5, 4/5	25.71
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

Table 13: Comparison of open-source and Proprietary LLM agents for **brain tumor detection task** in **MRI** environment

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
GPT-4.1	5/5, 5/5, 5/5	5/5	5/5	5/5, 5/5	100.00	5/5, 5/5, 5/5	5/5	5/5	5/5, 5/5	100.00
GPT-4o	5/5, 3/5, 5/5	5/5	4/5	1/5, 5/5	71.43	5/5, 3/5, 5/5	5/5	3/5	1/5, 5/5	68.57
GPT-4	5/5, 5/5, 5/5	1/5	2/5	3/5, 5/5	71.43	5/5, 4/5, 5/5	0/5	1/5	3/5, 5/5	65.71
GPT-4-Turbo	5/5, 5/5, 5/5	1/5	2/5	3/5, 5/5	71.43	5/5, 4/5, 5/5	4/5	1/5	3/5, 5/5	77.14
GPT-4.1-mini	5/5, 4/5, 5/5	5/5	3/5	4/5, 5/5	88.57	5/5, 3/5, 5/5	3/5	3/5	3/5, 5/5	77.14
GPT-4o-mini	5/5, 3/5, 3/5	5/5	3/5	3/5, 5/5	77.14	5/5, 2/5, 3/5	5/5	2/5	5/5, 5/5	74.29
GPT-o4-mini	5/5, 5/5, 5/5	5/5	3/5	4/5, 5/5	91.43	5/5, 4/5, 5/5	4/5	2/5	4/5, 5/5	85.71
GPT-o3-mini	5/5, 5/5, 5/5	1/5	1/5	4/5, 4/5	71.42	5/5, 4/5, 5/5	1/5	1/5	4/5, 5/5	74.29
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	1/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 4/5	34.29
Claude-3-7	5/5, 4/5, 3/5	2/5	1/5	4/5, 3/5	57.14	5/5, 3/5, 3/5	2/5	1/5	3/5, 3/5	57.14
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29	5/5, 3/5, 5/5	4/5	5/5	4/5, 5/5	88.57
DeepSeek-R1	5/5, 2/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 2/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 1/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 5/5, 5/5	5/5	4/5	4/5, 5/5	94.29	5/5, 4/5, 5/5	5/5	3/5	3/5, 5/5	85.71
LLaMA-4 Scout	5/5, 4/5, 5/5	5/5	4/5	2/5, 5/5	85.71	5/5, 3/5, 5/5	5/5	3/5	2/5, 5/5	74.29
Large Models										
DeepSeek-R1-70B	5/5, 5/5, 5/5	3/5	1/5	2/5, 5/5	74.29	5/5, 4/5, 5/5	3/5	0/5	2/5, 5/5	68.57
LLaMA-3-70B	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43
Medium Models										
Qwen QwQ 32B	5/5, 4/5, 5/5	4/5	4/5	4/5, 5/5	88.57	5/5, 4/5, 5/5	2/5	4/5	4/5, 5/5	82.86
Qwen3-30B	5/5, 2/5, 5/5	0/5	0/5	1/5, 5/5	48.57	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 1/5, 2/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 2/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 1/5, 4/5	2/5	1/5	1/5, 5/5	51.43	5/5, 1/5, 4/5	1/5	1/5	1/5, 5/5	48.57
LLaMA-3-8B	5/5, 3/5, 4/5	4/5	2/5	1/5, 5/5	62.86	5/5, 2/5, 4/5	4/5	2/5	1/5, 5/5	60.00
Qwen-3-14B	5/5, 1/5, 2/5	0/5	0/5	0/5, 5/5	28.57	5/5, 0/5, 2/5	0/5	0/5	0/5, 4/5	25.71
Gemma3-12B-instruct	5/5, 1/5, 2/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 2/5	0/5	0/5	0/5, 0/5	14.29

Across stages, higher-capacity models are most reliable on **Client Selection** and **Federated Training**, frequently achieving perfect scores $(5/5,\ 5/5)$. Performance degrades most notably on **Data Preprocessing** and especially **Label Harmonization**, where mid-tier and smaller models often obtain $0/5$ or $1/5$, reducing their **Overall** scores even when later stages are solved correctly.

Among open-source systems, **DeepSeek-V3** performs best (85.71 / 80.00) with comparatively balanced behavior across stages. **LLaMA-4 Maverick** and **LLaMA-4 Scout** form the next group (71.43–85.71 depending on guidance). Lower-capacity or less-aligned open-source models (e.g., **DeepSeek-R1**, **Qwen3-235B**, **Gemma3-12B-instruct**) frequently fail in early pipeline stages and therefore yield the lowest scores.

Table 14: Comparison of open-source and Proprietary LLM agents for **Glaucoma detection task** in **Fundus** environment

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
	a_1, a_2, a_3	a_4	a_5	a_6, a_7		a_1, a_2, a_3	a_4	a_5	a_6, a_7	
Proprietary Models										
GPT-4.1	5/5, 5/5, 5/5	5/5	5/5	4/5, 5/5	97.14	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29
GPT-4o	5/5, 2/5, 5/5	5/5	3/5	1/5, 5/5	74.29	5/5, 2/5, 5/5	5/5	3/5	1/5, 5/5	74.29
GPT-4	5/5, 4/5, 5/5	1/5	1/5	3/5, 5/5	68.57	5/5, 4/5, 5/5	0/5	1/5	3/5, 5/5	65.71
GPT-4-Turbo	5/5, 4/5, 5/5	1/5	1/5	3/5, 5/5	68.57	5/5, 4/5, 5/5	4/5	1/5	3/5, 5/5	77.14
GPT-4.1-mini	5/5, 4/5, 5/5	5/5	3/5	4/5, 5/5	88.57	5/5, 2/5, 5/5	3/5	4/5	3/5, 5/5	77.14
GPT-4o-mini	5/5, 3/5, 3/5	5/5	3/5	3/5, 5/5	77.14	5/5, 2/5, 3/5	5/5	1/5	4/5, 5/5	71.43
GPT-o4-mini	5/5, 4/5, 5/5	5/5	3/5	4/5, 5/5	88.57	5/5, 4/5, 5/5	4/5	3/5	4/5, 5/5	85.71
GPT-o3-mini	5/5, 4/5, 5/5	1/5	1/5	4/5, 4/5	68.57	5/5, 4/5, 5/5	1/5	2/5	4/5, 5/5	74.29
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 4/5	34.29
Claude-3-7	5/5, 3/5, 3/5	2/5	1/5	3/5, 3/5	57.14	5/5, 3/5, 3/5	2/5	1/5	3/5, 3/5	57.14
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29	5/5, 3/5, 5/5	4/5	5/5	4/5, 5/5	88.57
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 4/5, 5/5	5/5	4/5	4/5, 5/5	91.43	5/5, 4/5, 5/5	5/5	3/5	3/5, 5/5	85.71
LLaMA-4 Scout	5/5, 4/5, 5/5	5/5	4/5	2/5, 5/5	85.71	5/5, 1/5, 5/5	5/5	3/5	2/5, 5/5	74.28
Large Models										
DeepSeek-R1-70B	5/5, 4/5, 5/5	3/5	1/5	2/5, 5/5	71.43	5/5, 4/5, 5/5	3/5	0/5	2/5, 5/5	68.57
LLaMA-3-70B	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43
Medium Models										
Qwen QwQ 32B	5/5, 4/5, 5/5	4/5	4/5	4/5, 5/5	88.57	5/5, 4/5, 5/5	2/5	4/5	4/5, 5/5	82.86
Qwen3-30B	5/5, 1/5, 5/5	0/5	0/5	1/5, 5/5	48.57	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 0/5, 4/5	2/5	1/5	1/5, 5/5	51.43	5/5, 0/5, 4/5	1/5	1/5	1/5, 5/5	48.57
LLaMA-3-8B	5/5, 1/5, 4/5	4/5	2/5	1/5, 5/5	62.86	5/5, 0/5, 4/5	4/5	2/5	1/5, 5/5	60.00
Qwen-3-14B	5/5, 0/5, 0/5	0/5	0/5	0/5, 5/5	28.57	5/5, 0/5, 0/5	0/5	0/5	0/5, 4/5	25.71
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

Table 15: Comparison of open-source and Proprietary LLM agents for **pneumonia detection task** in chest **X-Ray** environment

Model	Fine-grained guidance					Goal-oriented guidance				
	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall	Client-Sel	Data-Pre	Label-Harm	Fed-Train	Overall
	a_1, a_2, a_3	a_4	a_5	a_6, a_7		a_1, a_2, a_3	a_4	a_5	a_6, a_7	
Proprietary Models										
GPT-4.1	5/5, 5/5, 5/5	5/5	5/5	5/5, 5/5	100.00	5/5, 5/5, 5/5	5/5	5/5	5/5, 5/5	100.00
GPT-4o	5/5, 1/5, 5/5	5/5	3/5	1/5, 5/5	71.43	5/5, 1/5, 5/5	5/5	2/5	1/5, 5/5	68.57
GPT-4	5/5, 5/5, 5/5	1/5	1/5	3/5, 5/5	71.43	5/5, 4/5, 5/5	0/5	1/5	3/5, 5/5	65.71
GPT-4-Turbo	5/5, 5/5, 5/5	1/5	1/5	3/5, 5/5	71.43	5/5, 4/5, 5/5	4/5	1/5	3/5, 5/5	77.14
GPT-4.1-mini	5/5, 4/5, 5/5	5/5	3/5	4/5, 5/5	88.57	5/5, 2/5, 5/5	3/5	4/5	3/5, 5/5	77.14
GPT-4o-mini	5/5, 3/5, 3/5	5/5	3/5	3/5, 5/5	77.14	5/5, 2/5, 3/5	5/5	1/5	5/5, 5/5	74.29
GPT-o4-mini	5/5, 5/5, 5/5	5/5	3/5	4/5, 5/5	91.43	5/5, 4/5, 5/5	4/5	3/5	4/5, 5/5	85.71
GPT-o3-mini	5/5, 5/5, 5/5	1/5	1/5	4/5, 4/5	71.42	5/5, 4/5, 5/5	1/5	2/5	4/5, 5/5	74.29
GPT-3.5-Turbo	5/5, 0/5, 0/5	0/5	0/5	1/5, 3/5	25.71	5/5, 0/5, 0/5	2/5	0/5	1/5, 4/5	34.29
Claude-3-7	5/5, 3/5, 3/5	2/5	1/5	3/5, 3/5	57.14	5/5, 3/5, 3/5	2/5	1/5	3/5, 3/5	57.14
Open-source Models										
Huge Models										
DeepSeek-V3	5/5, 4/5, 5/5	5/5	5/5	4/5, 5/5	94.29	5/5, 3/5, 5/5	4/5	5/5	4/5, 5/5	88.57
DeepSeek-R1	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
Qwen3 235B	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86	5/5, 0/5, 5/5	0/5	0/5	0/5, 5/5	42.86
LLaMA-4 Maverick	5/5, 5/5, 5/5	5/5	4/5	4/5, 5/5	94.29	5/5, 4/5, 5/5	5/5	3/5	3/5, 5/5	85.71
LLaMA-4 Scout	5/5, 4/5, 5/5	5/5	4/5	2/5, 5/5	85.71	5/5, 1/5, 5/5	5/5	3/5	2/5, 5/5	74.28
Large Models										
DeepSeek-R1-70B	5/5, 5/5, 5/5	3/5	1/5	2/5, 5/5	74.28	5/5, 4/5, 5/5	3/5	0/5	2/5, 5/5	68.57
LLaMA-3-70B	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43	5/5, 4/5, 5/5	2/5	2/5	2/5, 5/5	71.43
Medium Models										
Qwen QwQ 32B	5/5, 4/5, 5/5	4/5	4/5	4/5, 5/5	88.57	5/5, 4/5, 5/5	2/5	4/5	4/5, 5/5	82.86
Qwen3-30B	5/5, 1/5, 5/5	0/5	0/5	1/5, 5/5	48.57	5/5, 0/5, 5/5	0/5	0/5	1/5, 5/5	45.71
Gemma3-27B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29
Small Models										
Gemma-2-9B	5/5, 0/5, 4/5	2/5	1/5	1/5, 5/5	51.43	5/5, 0/5, 4/5	1/5	1/5	1/5, 5/5	48.57
LLaMA-3-8B	5/5, 1/5, 4/5	4/5	2/5	1/5, 5/5	62.86	5/5, 0/5, 4/5	4/5	2/5	1/5, 5/5	60.00
Qwen-3-14B	5/5, 0/5, 0/5	0/5	0/5	0/5, 5/5	28.57	5/5, 0/5, 0/5	0/5	0/5	0/5, 4/5	25.71
Gemma3-12B-instruct	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29	5/5, 0/5, 0/5	0/5	0/5	0/5, 0/5	14.29

Finally, **fine-grained guidance** consistently improves **overall** performance compared to **goal-oriented guidance**, indicating that explicit stepwise instructions help agents navigate the multi-stage federated learning workflow in Dermatology more effectively.

Table 11 compares open-source and proprietary LLM agents in the **Ultrasound environment** for breast cancer detection task under two guidance paradigms: fine-grained guidance, where each subtask is explicitly defined and goal-oriented guidance, where the model is only given the overall objective. Each model's performance is evaluated on four core subtasks, and the final column

Table 16: Comparison of average time taken by each agent to solve respective tasks (in seconds) using different LLMs.

Model	S_1	C_1	S_2	C_2	C_3	S_3	S_4
Proprietary Models							
GPT-4.1	1.8	64.8	55.5	302.4	130.7	54.1	18.8
GPT-4o	1.0	58.7	30.9	311.3	201.0	53.5	9.6
GPT-4	2.9	235.2	87.4	172.3	615.5	243.7	31.5
GPT-4-Turbo	1.8	81.2	54.8	259.9	266.7	76.7	16.6
GPT-4.1-mini	1.0	78.1	29.9	183.6	161.5	69.8	9.7
GPT-4o-mini	1.0	73.4	29.3	370.7	292.1	77.0	10.7
GPT-o4-mini	4.2	164.8	127.4	404.6	503.9	168.2	42.5
GPT-o3-mini	4.9	156.2	145.9	177.7	412.3	172.1	44.5
GPT-3.5-Turbo	1.1	51.1	32.8	163.9	199.9	52.7	9.9
Claude-3-7	3.9	231.6	115.5	414.0	457.7	203.0	37.2
Open-source							
Huge Models							
DeepSeek-V3	4.4	169.3	131.2	554.1	461.5	197.2	44.1
DeepSeek-R1	8.1	162.9	242.1	567.1	328.0	134.2	77.4
Qwen3 235B	11.0	180.3	328.8	642.8	440.7	168.9	108.3
Large Models							
LLaMA-4 Maverick	1.2	98.9	37.2	124.2	282.7	118.3	13.6
LLaMA-4 Scout	2.3	105.3	69.1	172.0	300.4	103.6	24.6
DeepSeek-R1-70B	1.5	96.0	44.4	168.0	312.5	99.0	15.2
LLaMA-3-70B	1.5	93.2	45.4	193.7	257.4	76.3	15.0
Medium Models							
Qwen QwQ 32B	0.8	77.2	24.0	186.0	253.1	74.2	8.4
Qwen3-30B	2.3	73.9	68.2	164.4	297.4	83.7	24.6
Gemma3-27B-instruct	2.8	140.9	82.8	297.4	535.0	133.9	26.3
Small Models							
Gemma-2-9B	0.5	116.9	15.4	105.3	283.1	111.2	5.1
LLaMA-3-8B	1.4	155.3	42.6	212.1	573.4	144.9	13.5
Qwen-3-14B	4.1	165.1	123.6	520.0	357.3	176.5	45.0
Gemma3-12B-instruct	3.1	184.7	94.4	400.2	487.9	195.1	33.5

Table 17: Summary Table showing overall performance (%) across six FL environments under Fine-grained (FG) and Goal-oriented (GO) guidance.

Model	Dermatology		Ultrasound		MRI		Fundus		X-ray		Histopathology	
	FG	GO	FG	GO	FG	GO	FG	GO	FG	GO	FG	GO
Proprietary Models												
GPT-4.1	94.29	88.57	94.29	94.29	100.00	100.00	97.14	94.29	100.00	100.00	94.29	94.29
GPT-4o	71.43	65.71	68.57	65.71	71.43	68.57	74.29	74.29	71.43	68.57	65.71	62.86
GPT-4	65.71	51.43	65.71	62.86	71.43	65.71	68.57	65.71	71.43	65.71	54.29	51.43
GPT-4-Turbo	68.57	74.29	65.71	74.29	71.43	77.14	68.57	77.14	71.43	77.14	57.14	65.71
GPT-4.1-mini	88.57	82.86	85.71	77.14	88.57	77.14	88.57	77.14	88.57	77.14	85.71	80.00
GPT-4o-mini	68.57	57.14	71.43	71.43	77.14	74.29	77.14	71.43	77.14	74.29	65.71	60.00
GPT-o4-mini	85.71	74.29	85.71	82.86	91.43	85.71	88.57	85.71	91.43	85.71	77.14	68.57
GPT-o3-mini	65.71	60.00	60.00	62.86	71.42	74.29	68.57	74.29	71.42	74.29	71.43	68.57
GPT-3.5-Turbo	25.71	31.43	25.71	34.29	25.71	34.29	25.71	34.29	25.71	34.29	25.71	31.43
Claude-3-7-Sonnet	51.42	57.14	54.29	54.29	57.14	57.14	57.14	57.14	57.14	57.14	51.43	57.14
Open-source Models												
Huge Models												
DeepSeek-V3	85.71	80.00	91.43	85.71	94.29	88.57	94.29	88.57	94.29	88.57	91.43	88.57
DeepSeek-R1	42.86	42.85	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86
Qwen3-235B	42.86	42.85	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86	42.86
LLaMA-4 Maverick	71.43	74.29	88.57	82.86	94.29	85.71	91.43	85.71	94.29	85.71	77.14	71.43
LLaMA-4 Scout	74.29	77.14	85.71	74.28	85.71	74.29	85.71	74.28	85.71	74.28	80.00	77.14
Large Models												
DeepSeek-R1-70B	45.71	42.86	74.28	68.57	74.29	68.57	71.43	68.57	74.28	68.57	42.86	42.86
LLaMA-3-70B	54.29	57.14	68.57	68.57	71.43	71.43	71.43	71.43	71.43	71.43	54.29	60.00
Medium-sized Models												
Qwen QwQ 32B	91.43	85.71	85.71	80.00	88.57	82.86	88.57	82.86	88.57	82.86	85.71	82.86
Qwen3-30B	45.71	45.71	45.71	45.71	48.57	45.71	48.57	45.71	48.57	45.71	45.71	45.71
Gemma3-27B-instruct	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29
Small Models												
Gemma-2-9B	51.43	51.43	51.43	48.57	51.43	48.57	51.43	48.57	51.43	48.57	57.14	54.29
LLaMA-3-8B	65.71	65.71	60.00	60.00	62.86	60.00	62.86	60.00	62.86	60.00	65.71	65.71
Qwen-3-14B	42.86	40.00	28.57	25.71	28.57	25.71	28.57	25.71	28.57	25.71	42.86	40.00
Gemma3-12B-instruct	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29	14.29

reports the average normalized score. GPT-4.1 achieves the highest performance (94.29) under both guidance types, demonstrating strong generalization across all subtasks. Smaller models like Gemma3-12B-instruct significantly underperform (14.29), especially when tasks require coherent execution across multiple stages. Open-source models such as DeepSeek-V3 and LLaMA-4 Maverick exhibit competitive performance with proprietary models under fine-grained prompts but show mild performance decline in goal-oriented execution.

Table 12 evaluates LLM agents' capability to simultaneously plan and generate independent scripts for each subtask in the Ultrasound environment without any explicit or implicit guidance on the workflow or availability of tools. It is expected to plan the entire process for completion of each subtask as well as write scripts for completing the tasks. This setup is more challenging than the previous table. A uniform drop in performance is observed across all models, regardless of type or size. Most top proprietary models, such as the GPT-4 series, drop to a common score of 42.86, indicating reliance on guided execution for complex task planning. Mid- and small-scale models like Claude-3-7, Gemma3-12B, and Qwen-3-14B perform poorly, with scores as low as 14.29 to 31.43, demonstrating the importance of tools for domain-specific and robust task understanding.

Table 13 presents evaluation in the **MRI environment**, following the same structure. GPT-4.1 again leads with a perfect score (100.00) under both guidance types. A general trend of better performance under fine-grained guidance than goal-oriented guidance is maintained across most models. Open-source large-scale models such as DeepSeek-V3 and LLaMA-4 Maverick narrow the performance gap significantly, achieving scores above 85 under fine-grained guidance. Models with weaker subtask handling like Gemma3-12B-Instruct remain consistently poor performers, struggling to follow multi-step instructions even in highly structured MRI tasks.

Table 14 presents the evaluation of proprietary and open-source LLM agents deployed in the **Fundus environment**. The table demonstrates that proprietary models such as GPT-4.1 and GPT-4.1-mini achieve near-perfect scores across both guidance styles, indicating robust task execution capabilities. GPT-4.1 achieves the highest overall fine-grained score (97.14) and maintains a strong goal-oriented score (94.29), suggesting high generalization capacity even with minimal instruction. In contrast,

smaller models like GPT-3.5-Turbo and Gemma3-12B-instruct exhibit major limitations, particularly under goal-oriented prompting, often failing multiple subtasks and scoring below 35.

Among open-source models, DeepSeek-V3 and LLaMA-4 Maverick lead performance under both guidance types, with fine-grained scores above 90 and goal-oriented scores above 85. These models close the gap with top proprietary agents, showcasing the progress of the open-source ecosystem. However, performance drops significantly in lightweight models such as Qwen-3-14B and Gemma3-12B-instruct, which perform well only on the most basic subtasks and fail to coordinate complex operations under goal-driven conditions.

Table 15 presents results for open-source and proprietary LLM agents in the **XRay environment**. Proprietary systems remain strongest: GPT-4.1 achieves ceiling performance ($5/5$ on all sub-tasks. A second tier follows with GPT-o4-mini, GPT-4.1-mini, GPT-4o-mini, GPT-4-Turbo, GPT-4, extbfGPT-4o, and GPT-o3-mini. Claude-3-7 shows moderate performance, while GPT-3.5-Turbo demonstrates substantially weaker performance.

Open-source models narrow the gap in this modality. DeepSeek-V3 and LLaMA-4 Maverick approach the top proprietary tier, with LLaMA-4 Scout and Qwen QwQ 32B delivering competitive results. Lower-performing models include DeepSeek-R1, Qwen3-235B, and Gemma3-27B-instruct. Among smaller models, LLaMA-3-8B exceeds the performance of Gemma-2-9B and Qwen-3-14B.

Stage-wise behaviour matches other environments: strong models consistently solve Client Selection and Federated Training ($5/5$, $5/5$), whereas weaker models falter on Data Pre-processing and Label Harmonization, yielding frequent $0/5$ or $1/5$. Fine-grained guidance generally improves Overall scores relative to goal-oriented guidance, confirming the benefit of explicit stepwise supervision for X-Ray workflows. Overall, all the tables reveal two key insights: (1) proprietary models consistently outperform open-source ones across both settings, (2) fine-grained prompting benefits all models but especially weaker ones. More insightful discussion on the results can be found in Appendix D.3.

D.2 Discussion on time-efficiency

Table 16 compares the average time taken (in seconds) by each agent across the seven subtasks (S1–S4, C1–C3) in the pipeline. GPT-4.1 is among the fastest overall, particularly in inference-heavy subtasks like S1 and S4. Open-source models such as Qwen3-235B and DeepSeek-R1 exhibit significantly higher latency, especially in complex subtasks like C2, where times range from approximately 550 to 640 seconds. Lightweight models such as Qwen QwQ 32B and Gemma-2-9B complete tasks much faster but at the cost of performance, as seen in the other tables. This table complements the prior performance evaluations by highlighting the efficiency–performance tradeoff, which is critical for real-world federated deployments.

We have conducted a comparison of **time–efficiency vs. performance** for each agent role (S1, C1, S2, C2, C3, S3, S4) across model families. Overall, we observe the following:

C2 (data prep) and C3 (label harmonization) dominate wall-clock time for almost every model. S1/S2/S3/S4 are comparatively light; differences here are smaller and rarely drive total runtime. The best choices balance high stage success and short C2/C3 times. Agent-wise takeaways (cross-model):

(i) S1 (server task extraction/broadcast). Times are uniformly small. Fastest include Gemma-2-9B (0.5s), QwQ-32B (0.8s), GPT-4o/4o-mini/4.1-mini (1.0s). This stage won't bottleneck overall runtime, so one should prefer models with higher downstream success rather than saving fractions of a second here.

(ii) C1 (client selection). A moderate cost stage. GPT-3.5-Turbo (51.1s) and QwQ-32B (77.2s) are among the fastest; GPT-4.1 (64.8s) and GPT-4.1-mini (78.1s) are also efficient. Very large open-source models (e.g., Qwen3-235B 180s) are slower without clear gains.

(iii) S2 (approval/coordination). Also light in terms of time complexity. Gemma-2-9B (15.4s), QwQ-32B (24.0s), GPT-4o/4o-mini/4.1-mini (29–30s) are quickest.

(iv) C2 (data prep / cleaning). One of the two big time sinks. Fastest include Gemma-2-9B (105s) and LLaMA-4 Maverick (124s); GPT-3.5 (164s), Qwen3-30B (164s), LLaMA-3-70B (194s), QwQ-32B (186s) are solid. GPT-4.1 (302s) and huge open-source (DeepSeek-V3 554s; Qwen3-235B 643s) are

Table 18: User instruction samples mapped to their ground-truth federated learning algorithms. Each instruction encodes a distinct FL requirement such as class-imbalance mitigation, adaptive optimization, heterogeneous-architecture personalization, prototype-based collaboration, or domain generalization and the corresponding correct algorithm is shown in the rightmost column.

Instr. #	User instruction or requirement	Correct Algorithm
1	Train a federated learning model using an algorithm designed to mitigate both inter-client and intra-client class imbalance while still producing a strong global model.	**FedLC**
2	Train a federated learning model that supports a dynamic gradient adjustment scheme, allowing the learning rate to adapt based on client updates and training dynamics.	**FedOpt**
3	Train personalized federated learning models where each client maintains a distinct architecture. Use server-side knowledge distillation to enable joint learning while preventing client drift.	**FedMD**
4	Train personalized federated learning models where raw parameters cannot be exchanged. Instead, allow clients to exchange only class-centroid embeddings for collaboration.	**FedProto**
5	Train a federated domain-generalization model that learns domain-invariant representations across clients, enabling strong performance on unseen out-of-distribution clients.	**FedSR**

slower. LLaMA-4 Maverick and QwQ-32B are strong Pareto options (good success, reasonable C2 time).

(v) C3 (label harmonization). The other major time sink and the hardest stage. Standout: GPT-4.1 (131s)—both fast and high success. Next tier includes QwQ-32B (253s) and LLaMA-3-70B (257s), which are respectable; GPT-4o (201s) is faster than many but weaker on Label Harmonization accuracy. GPT-4 (616s) and huge open-source (e.g., DeepSeek-V3 462s) are slow here.

(vi) S3 (algorithm selection). Lightweight. GPT-3.5 (52.7s), GPT-4o (53.5s), GPT-4.1 (54.1s) are quickest; QwQ-32B (74s) is not far behind. This stage rarely determines end-to-end time.

(vii) S4 (training trigger/monitor). Very small across models. Gemma-2-9B (5.1s) is fastest; QwQ-32B (8.4s), GPT-4o/4.1-mini (9–10s) are close. Not a driver of total latency.

We summarize the overall recommendations based on our experiments below:

Best overall (reliability & time): GPT-4.1 with exceptional C3 time (130.7s) and top success. Best open-source Pareto: Qwen QwQ 32B with 186s for C2 and 253s for C3 with strong success; or LLaMA-4 Maverick if faster C2 is needed (124s). Budget/latency-focused orchestration: GPT-4.1-mini or GPT-4o-mini (But need to keep in mind the success drop on C3). It is advisable to avoid very large open-source for time-critical runs unless one specifically needs open-source + the higher success of DeepSeek-V3 (and can pay the time cost).

D.3 Discussion on client selection, reasoning vs non-reasoning models and failure modes:

Qualitative analysis of client selection across modalities. Figures 10-34 present the qualitative agentic performance in the *Client Selection* stage under three clinical modalities, *viz.,* **skin cancer** (dermatology), **histopathology** (breast cancer detection), and **X-Ray** (pneumonia detection) and contrast *non-thinking/reasoning* and *thinking/reasoning* LLM agents. Across all settings, the figures illustrate *when/how* the server approves or declines prospective clients for federated training. For non-thinking agents (e.g., Figs. 10-12; 21-25; 28-30), the selection is typically concise: the model applies eligibility checks and emits a binary decision (approve/decline) with minimal justification. This often highlights crisp gating on dataset relevance to the target task, basic quality constraints, and coarse client readiness.

Table 19: FL algorithm choices per user instruction (see Table 18) for each model. [ ] denotes no valid algorithm returned.

Model	Instr. 1	Instr. 2	Instr. 3	Instr. 4	Instr. 5
Ground Truth	**FedLC**	**FedOpt**	**FedMD**	**FedProto**	**FedSR**
GPT-4.1	FedLC	FedOpt	FedMD	FedProto	FedSR
GPT-4o	FedLC	FedDyn	[ ]	CCVR	FedIIR
GPT-4	FedLC	FedDyn	FedMD	CCVR	FedSR
GPT-4-Turbo	FedLC	FedOpt	[ ]	FedProto	FedIIR
GPT-4.1-mini	FedLC	FedDyn	[ ]	FedProto	FedSR
GPT-4o-mini	FedLC	FedDyn	[ ]	CCVR	FedIIR
GPT-o4-mini	FedLC	FedOpt	[ ]	FedProto	FedIIR
GPT-03-mini	FedLC	FedOpt	[ ]	FedProto	FedIIR
GPT-3.5-Turbo	FedProx	FedOpt	[ ]	CCVR	[ ]
Claude-3-7-Sonnet	FedLC	FedOpt	[ ]	[ ]	[ ]
DeepSeek-V3	FedLC	FedOpt	[ ]	FedProto	FedSR
DeepSeek-R1	[ ]	FedDyn	[ ]	CCVR	[ ]
Qwen3 235B	FedProx	FedDyn	[ ]	[ ]	FedProx
LLaMA-4 Maverick	FedProx	FedOpt	FedGen	FedProx	FedProx
LLaMA-4 Scout	FedProx	FedOpt	FedGen	FedProx	FedProx
DeepSeek-R1-70B	[ ]	FedOpt	[ ]	CCVR	[ ]
LLaMA-3-70B	FedProx	FedOpt	[ ]	FedProx	FedProx
Qwen QwQ 32B	FedLC	FedOpt	FedMD	FedProto	FedSR
Qwen3-30B	[ ]	[ ]	[ ]	FedProto	[ ]
Gemma3-27B-instruct	FedProx	FedDyn	[ ]	[ ]	[ ]
Gemma-2-9B	FedLC	FedDyn	[ ]	[ ]	[ ]
LLaMA-3-8B	FedProx	FedDyn	[ ]	FedProto	[ ]
Qwen-3-14B	[ ]	[ ]	[ ]	[ ]	[ ]
Gemma3-12B-instruct	FedProx	FedDyn	[ ]	[ ]	[ ]

Impact of using thinking/reasoning agents For **thinking/reasoning** agents (e.g., Figs.13-17, 26-27, 31-34), the server-facing rationale becomes more elaborate. These figures show richer criteria such as finer judgements about class balance, labeling consistency, or potential contribution to global convergence before issuing approve/decline decisions. While this often results in clearer, auditable justifications, it can also introduce overhead: Fig. 16 exemplifies *overthinking*, where extended deliberation adds verbosity without changing the final decision. Taken together, the sequences suggest a trade-off: explicit reasoning improves transparency and sometimes catches subtle issues, but may reduce efficiency and occasionally distract from the primary selection objective.

Failure modes: hallucination and task drift. Figures 18-19 document characteristic **hallucinations** during client selection with skin cancer datasets. In one case, the model drifts to an *irrelevant task*, attempting to solve something other than client eligibility; in another, it answers in *Russian*, a response channel misaligned with the specified instruction and downstream system expectations. Such behaviors indicate vulnerability to prompt misinterpretation and context leakage even at the pre-training data curation stage. The remaining thinking-model traces (e.g., Fig 20) demonstrate successful recoveries where the agent returns to the approval/decline protocol after structured reasoning.

Consistency across datasets and tasks. Across **histopathology** (breast cancer) and **X-Ray** (pneumonia) examples, we observe the same qualitative patterns: non-thinking models provide fast, rule-like triage; thinking models surface nuanced justifications but are susceptible to verbosity and occasional digressions. The figures collectively map the decision boundary between acceptance and rejection anchored in dataset/task alignment and basic quality signals while exposing two practical risks for agentic selection: (i) *over-elaboration*, which inflates latency without added value, and (ii) *hallucination/task drift*, which can misroute the pipeline if not caught by server-side validation. These qualitative insights complement the quantitative tables, clarifying *how* different prompting regimes lead to the observed approval/decline outcomes in federated client onboarding.

Table 20: Impact of FL algorithm selection and data preprocessing correctness on downstream model performance.

FL algorithm selection	Comments	Data pre	Accuracy	Precision	Recall	F1 Score	Round no.
[]	Defaulting to FedAvg	×	57.6488	58.0859	57.8158	57.9505	69
×	Chosen algorithm: FedProx	×	72.0668	71.9638	72.0871	72.0254	83
✓	Chosen algorithm: FedLC	×	76.7989	76.8456	76.2180	76.5305	98
[]	Defaulting to FedAvg	✓	63.7697	64.1144	63.4596	63.7853	64
×	Chosen algorithm: FedProx	✓	75.0048	75.5315	73.9155	74.7148	91
✓	Chosen algorithm: FedLC	✓	83.4788	83.1265	83.5065	83.3161	91

D.4 Federated Training Performance

To assess whether the chosen algorithm actually improves federated learning performance rather than merely satisfying the Training-start checklist, we evaluate models far beyond the Training-Start Verification metric. To validate this hypothesis, we run full end-to-end federated learning experiments, not just the setup phase. We present five different user instructions (covering traditional global FL, personalized FL, and Federated Domain Generalization) and their corresponding ground-truth algorithms in Table 18, and we report the performance of all LLMs on these five instructions in Table 19. These results show that for Instruction 1, some LLMs incorrectly select FedProx instead of FedLC, while others return no algorithm at all. The performance on Instruction 2 is also interesting, as several models latch onto the word *dynamic* and wrongly select FedDyn instead of FedOpt. We next perform a systematic analysis for Instruction 1, i.e., when the user issues the instruction:

> *"Train a federated learning model using an algorithm designed to mitigate both inter-client and intra-client class imbalance while still producing a strong global model."*

We evaluate all agentic systems for this condition across the entire Federated workflow. Across the 40-algorithm repository integrated in FedAgentBench, we observe that some agents correctly select FedLC, the only algorithm explicitly designed for class-imbalance mitigation. Some agents incorrectly choose FedProx, which regularizes client drift but does not address class imbalance. Others return no algorithm, which results in a fallback to FedAvg, the baseline Federated Learning algorithm.

Full experimental results (Appendix Tables 19 and 20 as well as Fig. 8) confirm that the algorithm choice indeed affects the final FL performance and convergence, not only the Training-start metric. To isolate contributing factors, we compare performance trajectories under two conditions:with and without a successful data-preprocessing step, and with correct, incorrect, or absent algorithm selection. We assume that the client selection and label harmonization step is performed successfully for this, else the system will throw intermediate error and the agents would not be able to reach the final step. The accuracy curves in Fig. 8 and the ablation in Table 20 show that:

1. Agents that correctly select FedLC (highlighted in red) *i.e.*, GPT-4.1, GPT-4o, GPT-4, GPT-4-Turbo, GPT-4.1-mini, GPT-4o-mini, GPT-o4-mini, GPT-O3-mini, Claude-3-7-Sonnet, DeepSeek-V3, Qwen QwQ 32B, Gemma-2-9B consistently achieve the highest accuracy, precision, recall, and F1
2. Agents that choose FedProx *i.e.* GPT-3.5-Turbo, Qwen3 235B, LLaMA-4 Maverick, LLaMA-4 Scout, LLaMA-3-70B, Gemma3-27B-instruct, LLaMA-3-8B, Gemma3-12B-instruct perform moderately better than naive FedAvg, but substantially weaker than FedLC
3. Agents that return no algorithm, *i.e.* DeepSeek-R1, DeepSeek-R1-70B, Qwen3-30B, Qwen-3-14B defaulting to FedAvg, perform the worst and fail to handle class imbalance.

The three-panel subplot in Fig. 8 further illustrates that overall performance reduces when the preprocessing step fails, affecting all agentic systems. The performance improves for the agents in red that correctly preprocess; and subplot 8 (c) shows full performance gains when all agents successfully complete preprocessing. In all these cases, we find that the correct algortihmic choice of FedLC performs better than FedProx which is incorrectly chosen by some LLMs, which is again better than defaulting to FedAvg.

Together, these results demonstrate that FEDAGENTBENCH does not rely solely on superficial "training start" checks. Instead, we validate the **actual downstream effectiveness** of agent decisions including algorithm selection via full-pipeline FL training runs, revealing meaningful differences in final performance.

E FUTURE WORK

Our failure-mode analysis highlights several limitations of current LLM agents that offer opportunities for improving future agent design and prompting strategies in the following ways:

1. **Domain-specific reasoning limitations:** Errors arising from insufficient domain-specific reasoning, particularly in tasks such as dermatology label harmonization or ultrasound dataset selection, suggest the need for domain-aware agents. Future extensions may integrate medical ontologies, specific vocabularies, or lightweight domain adapters to ensure that LLM agents reason over clinically valid label and task structures.
2. **Challenges with multi-step operations:** Many agents struggled with multi-step operations, frequently skipping essential preprocessing actions or performing them in the wrong order. This motivates the development of structured prompting templates that enforce explicit stepwise execution, checklist-style progress tracking, and intermediate self-verification before tool invocation (Chen et al., 2025a). Such structure may reduce the tendency of agents to shortcut or collapse multi-stage tasks.
3. **Overconfidence and shortcutting:** We observed systematic overconfidence and shortcutting where models produced plausible but incorrect outputs rather than expressing uncertainty. Incorporating uncertainty-aware behaviors such as confidence reporting, contrastive evaluation of alternative outputs, consistency checks, and self-reflection frameworks across multiple reasoning paths may mitigate hallucinations in structured FL operations.
4. **Lack of workspace grounding:** Hallucinations and task-type mismatches indicate that agents often reasoned without grounding their decisions in the actual client workspace. Future research could explore: (i) prompting with explicit instructions to avoid relying on prior knowledge and instead use only the information provided via prompts, descriptions, or task files, and (ii) workspace-grounded decision pipelines that require agents to inspect dataset descriptions, directory structures, and tool metadata before committing to actions.
5. **Need for adaptive prompting:** Our results show that fine-grained prompting substantially improves performance on complex tasks, whereas high-level prompting is sufficient for simpler tasks. This points toward adaptive prompting mechanisms, where the system dynamically adjusts prompt granularity through prompt optimization strategies, verification strictness, and agent role specialization based on the predicted complexity of each FL sub-task (Trivedi et al., 2025; Qu et al., 2025; Ramnath et al., 2025).

Beyond prompt- and agent-level improvements, two broader system-level directions emerge from our analysis:

Phase-specific LLM routing: One promising direction is the development of phase-specific LLM routing systems that dynamically select the most suitable agent or model for each FL sub-task. Given the heterogeneous performance of LLMs across phases such as label harmonization and client selection, an intelligent routing layer could substantially improve reliability and efficiency by leveraging the strengths of different agents.

Reinforcement learning–based reasoning: Another promising direction is the integration of reinforcement learning–based reasoning models Zhang et al. (2025); Singh et al. (2025). RL-guided refinement loops could enable agents to learn task-specific decision policies, such as resolving labeling conflicts, planning multi-step preprocessing pipelines, or selecting appropriate FL algorithms using verifiable, workspace-grounded signals. Such adaptive, feedback-driven reasoning may mitigate several observed failure patterns, especially those involving multi-step planning and semantic grounding.

Together, these directions open pathways for designing more reliable, grounded, and domain-adapted LLM agents capable of robustly orchestrating real-world federated learning workflows.

F DETAILED INSIGHTS FROM THE BENCHMARK

We summarize our observations below, providing clear reasoning and interpretation of the agents' behaviors:

1. **Task–Dataset Alignment Requires Abstract Semantic Reasoning**
   A consistent source of failure, especially in client selection and label harmonization, is the inability of many agents to reliably match task semantics with the correct dataset types. Even when tool outputs clearly specify modality or anatomy, weaker agents struggle to infer, for example, that brain tumor classification should ignore MRI segmentation datasets.
   These mistakes reflect a deeper issue. The reasoning step requires both:
   (a) interpreting the task description, and
   (b) mapping it to a dataset or label schema with differing granularity. We observe that large reasoning chains frequently drift semantically, leading to inclusion of irrelevant datasets or omission of required ones.
   For example: (i) In our benchmark, agents must infer that a task such as *"brain tumor classification"* requires **MRI classification** datasets and not similarly named **MRI segmentation** datasets even though both correspond to brain tumors.
   (ii) They must correctly interpret the semantics of disease labels, e.g., mapping terms like *"melanocytic lesion," "malignant melanoma,"* or *"melanoma in situ"* into the appropriate canonical classes.
   (iii) They must extract task intent from descriptions such as *"multi-class breast lesion detection from ultrasound images,"* identifying the modality, anatomy, and task type without explicit cues.
   (iv) They must resolve ambiguous or partially informative metadata, such as recognizing that a dataset on breast ultrasound dataset maybe unsuitable for an ultrasound classification workflow despite keyword matches as the modality of the datasets is histopathology instead of ultrasound.
   These abilities require conceptual understanding and multi-hop semantic inference, which many current models struggle to perform reliably.
2. **Fine-Grained Prompts Reduce Reasoning Drift**
   Across all environments, **structured prompting consistently improves success rates**. Fine-grained prompts constrain the reasoning space by enforcing a deterministic step order, *i.e.*, identify the task, list candidates, filter, verify, and justify, thereby reducing opportunities for hallucination. Goal-oriented prompts, by contrast, allow unconstrained reasoning drift, causing: hallucinated directories, incorrect class lists, misinterpreted dataset schemas, premature tool invocation. This effect is pronounced in Label Harmonization, where even small deviations in reasoning lead to incomplete or inconsistent mappings and so we have to provide the LLMs with examples to map fine-grained classes to broader categories in the fine-grained prompting.
   This challenge also becomes **pronounced in multi-step planning**, where several models struggle to follow the required instruction sequence and frequently deviate from the provided overall workflow. Instead of using the available tools to retrieve information from dataset folders or algorithm description files, weaker agents often rely on prior knowledge, skip essential steps, fabricate missing details, or even attempt to recreate tools that have already been supplied - behaviours that lead to unstable and incorrect reasoning.
   In our work, we observe several concrete cases where **agents ignore the tools explicitly provided for the task**. **For example**, even though the $self fclean$ tool is available to perform dataset cleaning, and dedicated file-reading and file-moving tools are provided to inspect and reorganize dataset directories, some agents often skip these tools entirely. Instead, they attempt to manually script file operations from scratch thereby hallucinating paths, misusing Python syntax, or relying on incomplete domain-specific prior knowledge, which leads to errors or incomplete outputs.
   In multiple instances, the agent fabricates commands such as mv *.jpg cleanedimages/ or invents non-existent directories like /data/clean/ rather than invoking the correct tool designed for this purpose. These behaviours underscore the difficulty models face in multi-step planning: even when a reliable tool exists, the agent may fail to recognize its relevance,

misuse it, or attempt to re-create its functionality, resulting in unstable or incorrect pipeline execution.

3. **Large Models Often Overthink and Are Not Always More Reliable**

   Interestingly, **reasoning depth does not scale monotonically with model size**. Open-weight mid-scale models such as Qwen QwQ-32B and LLaMA-4 Scout often outperform models 2–7× larger across multiple environments. A recurring pattern we observe is that larger models engage in excessive "over-thinking" and speculative reasoning that ultimately breaks the workflow.

   For example, in the client selection stage (as illustrated in Figs 10-31), some larger models repeatedly re-interpret simple rules, spending 20–30 lines debating a binary decision. In some other cases, they still fail to follow the required output template, even if they identify the correct dataset. In several cases, the agent returns long explanations or nested justifications instead of the precise string format expected by the benchmark (e.g., Approved. Prepare for training or the exact canonical algorithm name), causing downstream stages to fail due to template mismatches.

   Similarly, for FL algorithm selection, certain large models correctly infer the intended algorithm but embed it inside a paragraph or speculative rationale instead of returning the clean pre-specified output, making it unusable in subsequent phases.

   This pattern reflects a deeper reliability issue: larger models often generate unnecessarily long reasoning chains, hallucinate intermediate interpretations, or override their own correct conclusions, whereas mid-sized models tend to follow instructions more faithfully. Ultimately, reliability in this benchmark depends less on model size and more on instruction-following discipline, consistent template adherence, and robust grounding in tool-based workflows.

4. **Workspace-specific Grounding Failures Are a Major Source of Error**

   Many preprocessing steps require precise grounding in file-system realities: verifying folder structures, checking formats, validating the existence of files, and generating correct paths. Agents often fail because:

   (a) they hallucinate paths that resemble pretrained-distribution patterns,

   (b) they ignore tool outputs that contradict their prior reasoning,

   (c) they overwrite correct tool results with incorrect guesses,

   (d) they shortcut multi-step verification procedures.

   These behaviours illustrate how current LLMs often prioritize their internal generative expectations of how datasets should look over the ground-truth symbolic information provided by tools.

   A related failure pattern appears prominently in the dataset and algorithm selection stages, where agents disregard the datasets explicitly provided to them and instead rely on prior knowledge from pretraining.

   For instance, when given a fixed list of client datasets for skin cancer detection, several models ignore the actual available options and instead return well-known public datasets such as *ISIC 2018, ISIC 2019, or ISIC 2020*, even if these datasets are not part of that particular setting and are never shown to the agent through tools.

   A similar issue arises in the MRI environment, where some agents confidently select external datasets purely because they recognize these names from pretraining, despite the fact that they are not included anywhere in our simulated clients in that particular scenario.

   The same pattern appears during Federated Learning algorithm selection: agents occasionally propose algorithms such as *FedConsist, FedOptimizer*, or other variants that do not exist in our provided algorithm list. These behaviors highlight a strong tendency to fall back on pretrained "world knowledge" rather than grounding decisions in the actual symbolic inputs provided by the environment, thereby leading to systematic errors, hallucinations, and mismatches in the selection stages.

5. **Label Harmonization Requires Multi-Hop Semantic Reasoning and domain-specific knowledge**

   Label harmonization in medical datasets requires multi-hop semantic reasoning and a degree of domain-specific clinical knowledge, especially in healthcare contexts where label granularity carries diagnostic meaning.

For the binary skin-lesion task in Figure 6, the agent must understand, for example, that "Basal Cell Carcinoma," "Squamous Cell Carcinoma," and "Melanoma" are all malignant entities, while "Nevus," "Seborrheic Keratosis," and "Dermatofibroma" are benign. This distinction is rarely explicit in raw dataset labels and must be inferred through medical knowledge.

To harmonize these correctly, an agent must:

(1) infer which fine-grained labels represent malignant cancers,

(2) identify which labels represent benign lesions, and

(3) consolidate partially overlapping taxonomies across datasets.

This requires multi-hop reasoning steps such as: Nevus → benign lesion → map to Benign, or Basal Cell Carcinoma → skin cancer subtype → Malignant, as well as understanding that multiple malignant subtypes must collapse into the same canonical class. Current LLMs often lack adequate grounding in medical modalities and terminologies (or they rely on incomplete or noisy priors), which explains why some models sometimes misclassify "Atypical Nevus" as malignant or treat "Seborrheic Keratosis" as a cancer subtype. To perform reliable harmonization across institutions, agents must be conditioned with domain-specific information either through lightweight medical knowledge retrieval during the workflow, integrating structured medical taxonomies, attaching domain-specific adapters, or augmenting prompts with concise clinical definitions of relevant disease categories. Without such conditioning, the agent's harmonization decisions rely solely on general-purpose pretrained semantic priors, which are insufficient for accurate clinical label alignment and multi-hop medical label consolidation, leading to cascading errors in downstream FL stages.

All these patterns provide the first systematic view of why current LLM agents struggle even before facing real-world FL complexity, and offer concrete directions for developing more reliable agent reasoning systems.

G Privacy Analysis of Harmonized Labels and Metadata

Our benchmark's contribution lies in system-level automation and task performance evaluation, not in proving privacy guarantees. However, since FedAgentBench utilizes harmonized labels and some form of metadata exchange across clients, below, we rigorously analyze the privacy implications of these harmonized labels and transmitted metadata.

G.1 Mutual Information Analysis

Let X be the original dataset at a client, and $M = f(X)$ represent the harmonized labels and metadata extracted from the local dataset X, where f includes only non-identifying structural information and label taxonomies. In practice, f is a projection or generalization map (e.g., mapping "melanoma" and "BCC" both to "malignant"). To quantify potential data tracing risk, we use Mutual Information (MI):

$$\text{MI}(X; M) = H(X) - H(X|M)$$

where H is the Shannon entropy.

To guarantee minimal traceability:

$$\text{MI}(X; M) \leq \delta, \quad \delta \to 0$$

Proof:

- By designing the function f (harmonization process), we ensure maximal entropy in $H(X|M)$.
- Assume f maps multiple distinct datasets $X_i \in \mathcal{X}$ to a similar M. Let $|\mathcal{X}| \gg |\mathcal{M}|$. This introduces significant ambiguity, thus:

$$H(X|M) \approx H(X)$$

which implies:

$$\text{MI}(X; M) \approx 0$$

Hence, tracing original data through metadata is theoretically negligible.

G.2 Differential Privacy (DP) Proof

We formalize DP guarantees.

Let $\mathcal{A}$ be a randomized mechanism (e.g., gradient updates with Gaussian noise), and D, D' two neighboring datasets differing by one record. $\mathcal{A}$ satisfies (ϵ, δ)-DP if:

$$\Pr(\mathcal{A}(D) \in S) \leq e^{\epsilon}\Pr(\mathcal{A}(D') \in S) + \delta, \quad \forall S \subseteq \text{Range}(\mathcal{A})$$

Proof Outline:

- If Gaussian noise $\mathcal{N}(0, \sigma^2)$ is added to updates during training:
$$\mathcal{A}(D) = \nabla f(D) + \mathcal{N}(0, \sigma^2)$$
- For mechanism sensitivity Δ, noise variance σ^2 satisfies:
$$\sigma \geq \frac{\Delta\sqrt{2\ln(1.25/\delta)}}{\epsilon}$$

thus rigorously satisfying DP conditions.

G.3 k-Anonymity Analysis

Let $\mathcal{C}$ be the set of clients. Metadata M ensures k-anonymity if each metadata description transmitted from a client $m \in M$ is generalized such that it matches at least k indistinguishable clients:

$$\forall m \in M, \quad |\{c \in \mathcal{C} : f(X_c) = m\}| \geq k$$

Proof:

- By metadata generalization, f is designed such that distinct datasets yield identical or highly similar metadata.
- Given $|\mathcal{C}| \gg k$, the number of clients per metadata class is enforced:
$$|\{c \in \mathcal{C} : f(X_c) = m\}| \geq k$$

thus rigorously satisfying k-anonymity.

G.4 Privacy-Utility Trade-off

Define utility U as the expected accuracy of the trained model, and privacy loss ϵ as above. We have:

$$U(\epsilon) = \mathbb{E}[\text{Acc}(M_\epsilon)] \quad \text{with} \quad \frac{dU}{d\epsilon} > 0$$

implying greater privacy (lower ϵ) results in lower accuracy.

Theoretical Bound:

- Utility degradation due to noise addition (DP) or generalization (k-anonymity) is bounded by:
$$|U(\epsilon) - U(0)| \leq O\left(\frac{1}{\epsilon}\right)$$

This rigorous mathematical analysis demonstrates that harmonized labels and metadata transmission in **FedAgentBench** can achieve stringent privacy guarantees with negligible traceability risks, aligning with formal **differential privacy** and **k-anonymity** standards.

```
Running AgentPipelines/run.py...

Human Requirements:  I want to train skin cancer classification model. Only use datase
ts that have skin cancer classes
--------------------------------------------------------------------

communication_1_content Task: Skin cancer classification
Modality: Image

Clients, please respond with the name of the selected dataset that contains skin cance
r classes suitable for this classification task.
Selector Content: **Dataset Name** : DDI_skin_dataset

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : skinL2_dataset

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : ISIC_2020

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : PH2Dataset

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : Dermnet


////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : ISIC2018_HAM10000

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: no dataset: The provided dataset "skin_disease_classification_kaggle
" contains classes for skin diseases such as acne, eye bags, and redness, but it does
not include any skin cancer classes. Therefore, it is not suitable for the skin cancer
 classification task.
////////////////////////////////////////////////////////////////////

communication_2_content Client not needed for the task
```

Figure 10: Client Selection with **skin cancer datasets** for **non-thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

```
Selector Content: no dataset. The provided dataset "Monkeypox_Skin_Image_Dataset" cont
ains skin disease classes such as Chickenpox, Measles, Monkeypox, and Normal, but it d
oes not have skin cancer classes suitable for the skin cancer classification task.
////////////////////////////////////////////////////////////////////

communication_2_content Client not needed for the task
Selector Content: **Dataset Name** : Dermquest

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : augmented_skin_condition_dataset_kaggle
This dataset contains a skin cancer class named "Carcinoma" suitable for the skin canc
er classification task.

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : fitzpatrick17k

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : Dermis

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : ISIC_2017

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : Mednode

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : ISIC_2024

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : Derma7P

////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : skin_disease_kaggle_dataset
This dataset contains skin cancer classes such as Basal Cell Carcinoma (BCC) and Melan
oma, suitable for skin cancer classification.


////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
```

Figure 11: Client Selection with **skin cancer datasets** for **non-thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

```
Selector Content: **Dataset Name** : ISIC_2019
////////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: no dataset

None of the available datasets include skin cancer classes (e.g., melanoma, basal cel
l carcinoma, squamous cell carcinoma). Therefore, there is no suitable dataset for th
e given task.
////////////////////////////////////////////////////////////////////////

communication_2_content Client not needed for the task
Selector Content: **Dataset Name** : PAD_UFES_20

////////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : ISIC_2016

////////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: **Dataset Name** : skin_disease_3_class
This dataset includes the class “basal cell carcinoma,” which is a type of skin cance
r.

////////////////////////////////////////////////////////////////////////

communication_2_content Approved. Prepare for training
Selector Content: no dataset

The only available dataset (“Dermatology_tabular dataset”) is non-image (tabular) and
 does not include dermatology images or specific skin cancer classes, so it is not su
itable for a skin cancer classification model.

////////////////////////////////////////////////////////////////////////

communication_2_content Client not needed for the task
Selector Content: no dataset

The provided dataset (“skin-infection-disease-datasaet”) covers only infectious skin
conditions (cellulitis, impetigo, athlete’s foot, nail fungus, ringworm, cutaneous la
rva migrans, chickenpox, shingles) and does not include any skin cancer classes.
////////////////////////////////////////////////////////////////////////

communication_2_content Client not needed for the task
Selector Content: no dataset

None of the available datasets include skin cancer classes; the only dataset provided
 (Monkeypox_Skin_Image_Dataset) contains Chickenpox, Measles, Monkeypox, and Normal c
ategories, which do not cover any form of skin cancer.
////////////////////////////////////////////////////////////////////////
```

Figure 12: Client Selection with **skin cancer datasets** for **non-thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

```
Selector Content: **Dataset Name** : skin_disease_3_class

////////////////////////////////////////////////////////////////////

communication_2_content Since the client has returned a dataset named "skin_disease
_3_class", it indicates that they have provided a dataset.

Therefore, the response should be: "Approved. Prepare for training".
Selector Content: The dataset description indicates that the dataset is a tabular d
ataset containing clinical features for diagnosing skin diseases, which doesn't mat
ch the modality of Dermatology Images. Therefore, this dataset is not suitable for
the task of Skin Cancer Detection using Dermatology Images.

Since I have to respond with the name of the selected dataset, I will check if ther
e's any other dataset that matches the task and modality.

**No suitable dataset found** : The dataset description doesn't match the modality
of Dermatology Images.


////////////////////////////////////////////////////////////////////

communication_2_content Since the client has returned a dataset, although it's not
suitable for the task, the client has still responded with a dataset. However, the
response indicates that the dataset is not suitable.

The correct interpretation of the task is to respond based on whether the client ha
s returned any dataset. Since the client has returned a dataset (even though it's n
ot suitable), the response should be based on the presence of a dataset.

Therefore, the response should be: "Approved. Prepare for training".
Selector Content: **Dataset Name** : skin-infection-disease-datasaet

////////////////////////////////////////////////////////////////////

communication_2_content Since the client has returned a dataset named "skin-infecti
on-disease-datasaet", I will return the approval message.

"Approved. Prepare for training"
Selector Content: **Dataset Name** : Monkeypox_Skin_Image_Dataset

////////////////////////////////////////////////////////////////////

communication_2_content Since the client has returned a dataset, **Dataset Name** :
 Monkeypox_Skin_Image_Dataset, we can infer that the client has indeed provided a d
ataset for the task.

Therefore, the response should be: "Approved. Prepare for training".
Selector Content: **Dataset Name** : Dermquest

////////////////////////////////////////////////////////////////////

communication_2_content Since the client has returned a dataset named "Dermquest",
the response should be: "Approved. Prepare for training".

Therefore, the output is: Approved. Prepare for training
```

Figure 13: Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

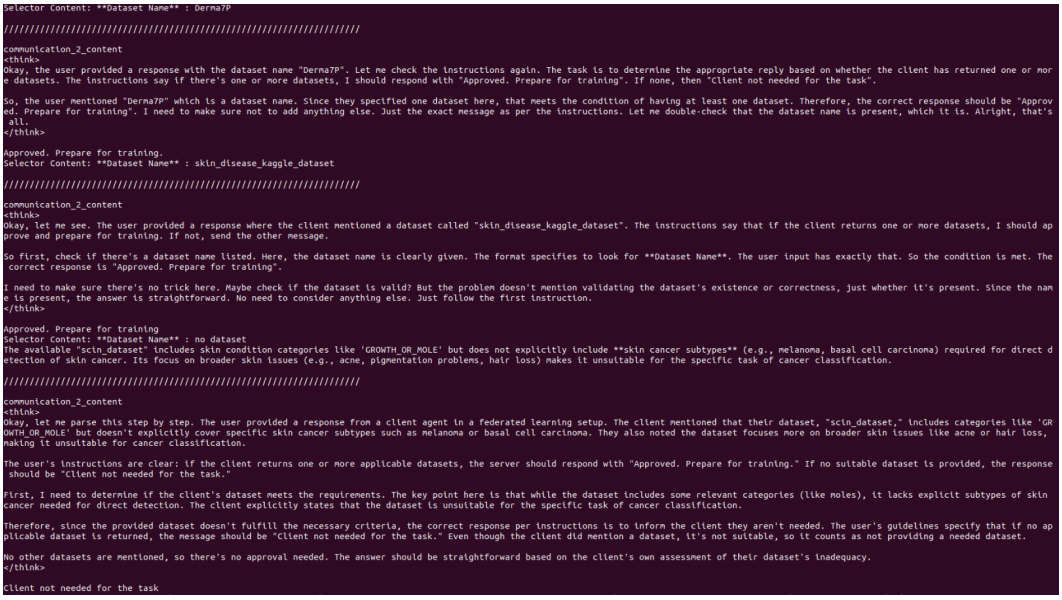


Figure 14: Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

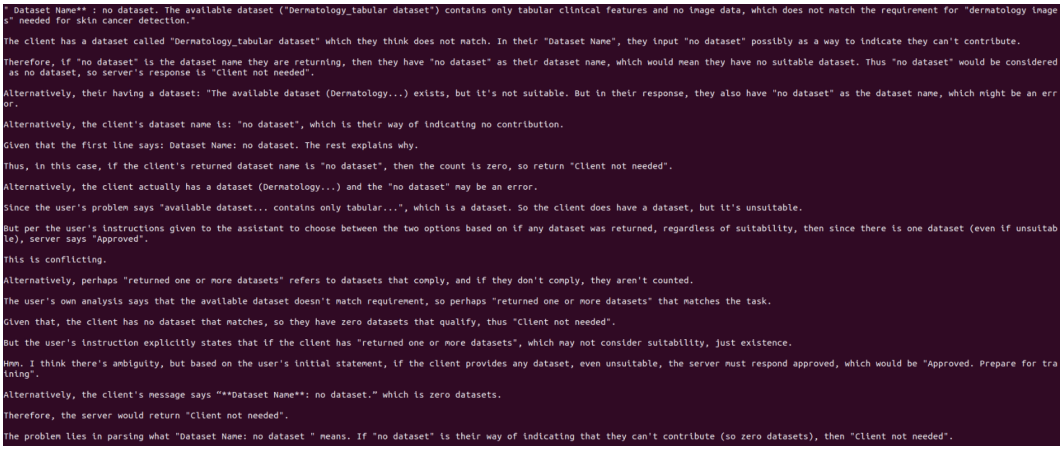


Figure 15: Client Selection with **skin cancer datasets** for **thinking/reasoning models**. It shows when/how the server approves clients for training whereas when/how it declines.

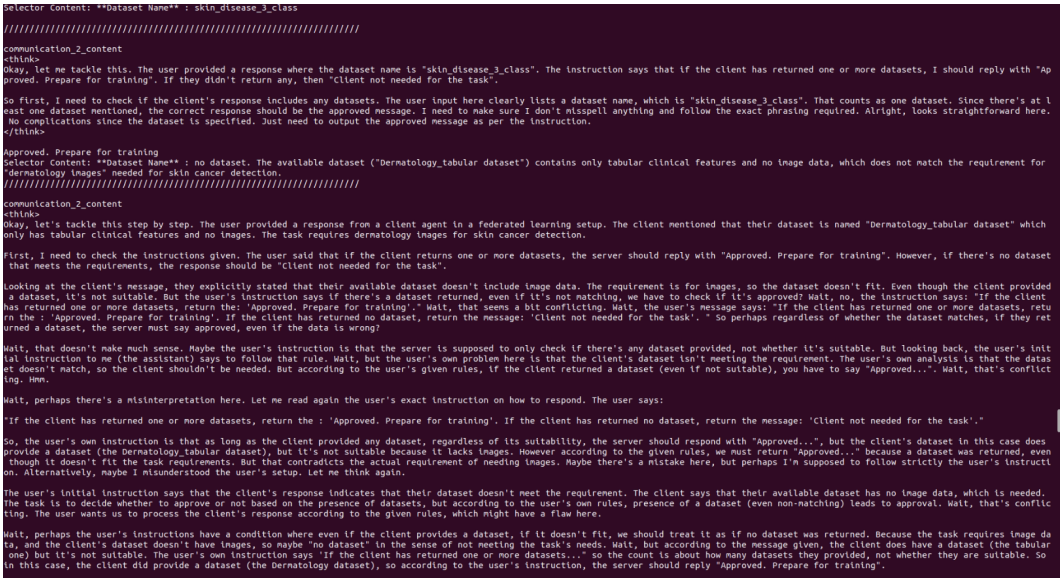


Figure 16: Client Selection with **skin cancer datasets** for **thinking/reasoning models**. It shows when/how the server approves clients for training whereas when/how it declines. **This is an instance of typical overthinking**

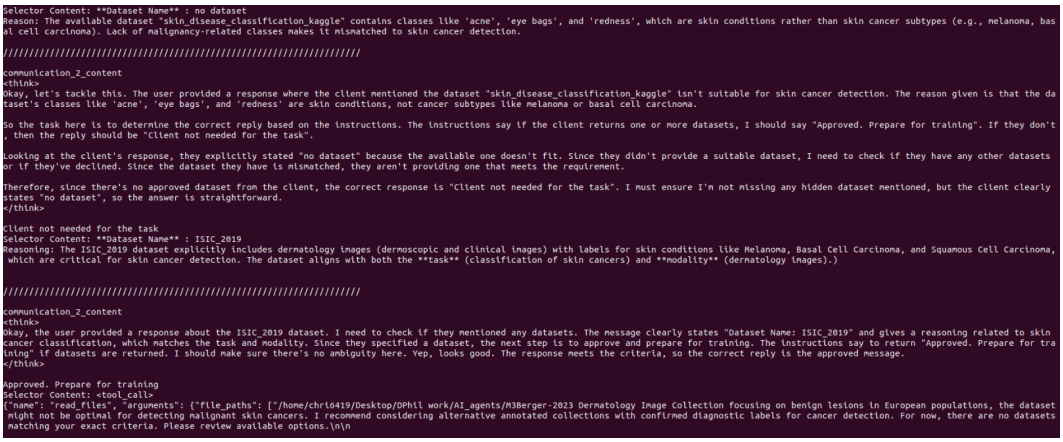


Figure 17: Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

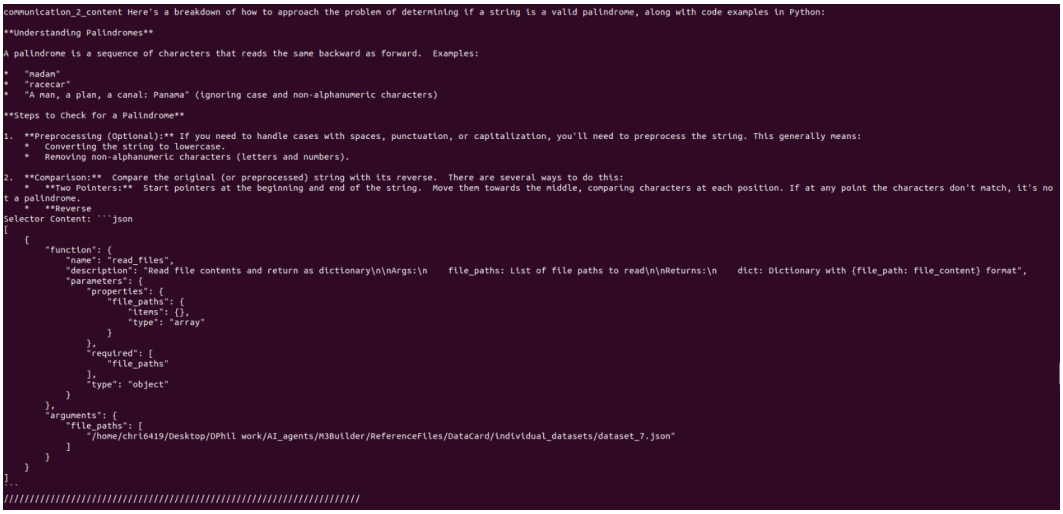


Figure 18: Typical example of **hallucination** in Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows the model is trying to solve irrelevant task.

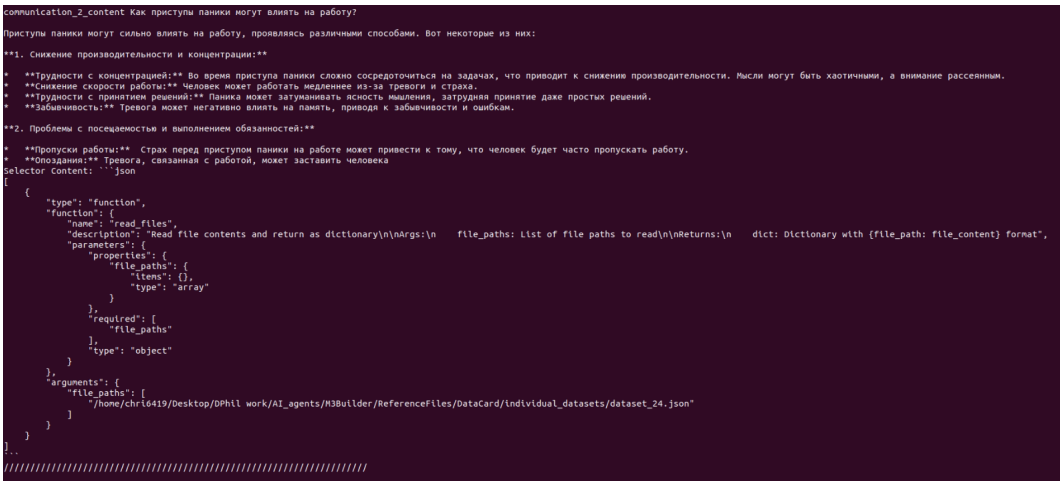


Figure 19: Typical example of **hallucination** in Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows the model is answering in Russian language which is totally unrelated to the given task.

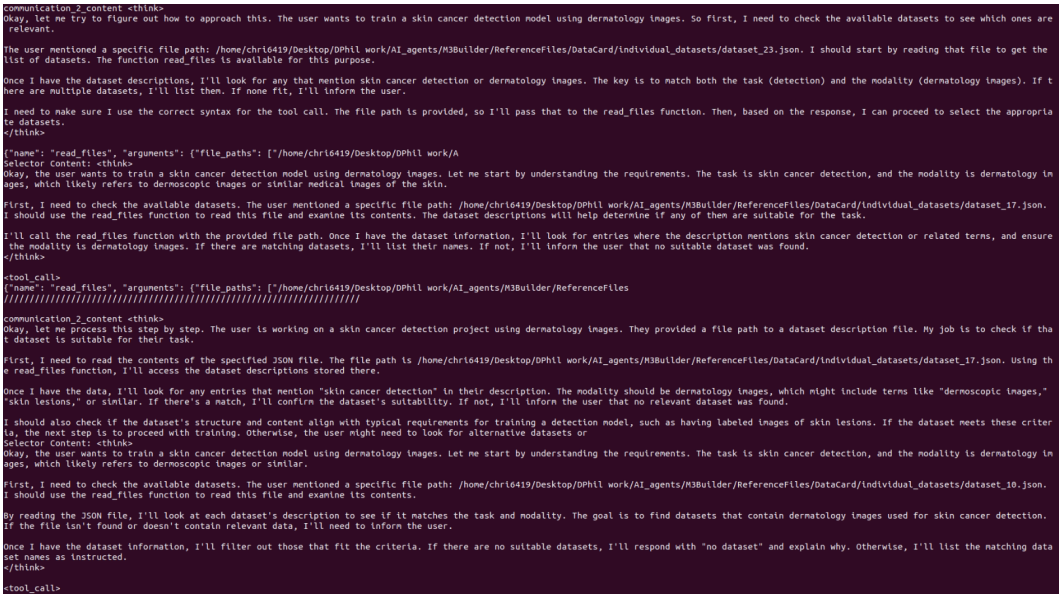


Figure 20: Client Selection with **skin cancer datasets** for **thinking/reasoning** models. It shows when/how the server approves clients for training whereas when/how it declines.

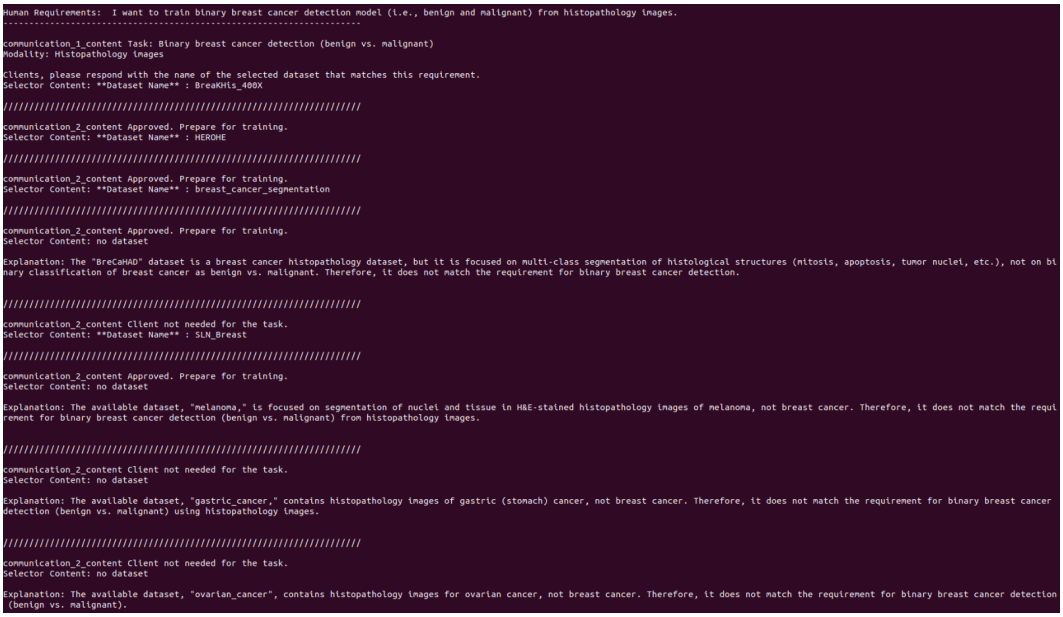


Figure 21: Client Selection with **histopathology datasets** for **non-thinking/reasoning** models on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

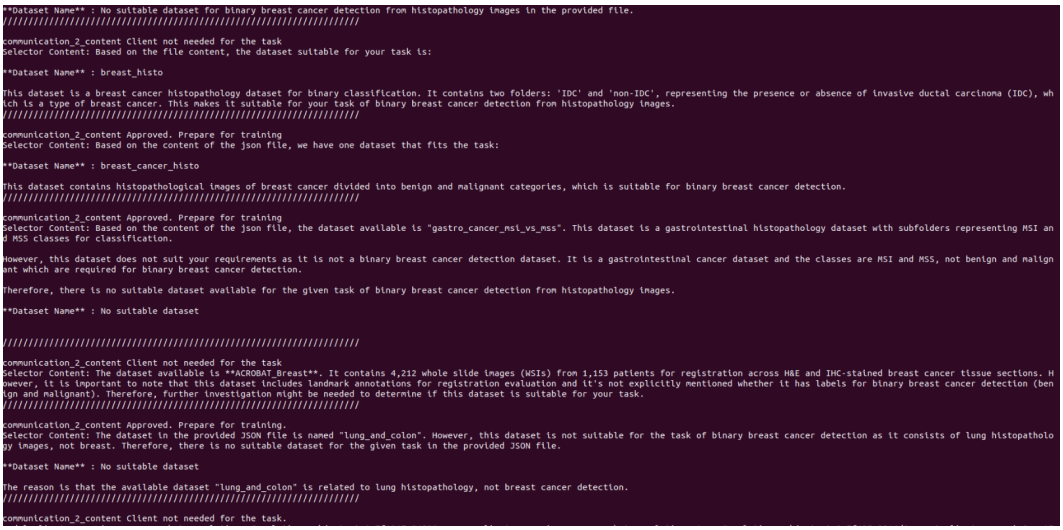


Figure 22: Client Selection with **histopathology datasets** for **non-thinking/reasoning** models on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

Figure 23: Client Selection with **histopathology datasets** for **non-thinking/reasoning** models **on breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

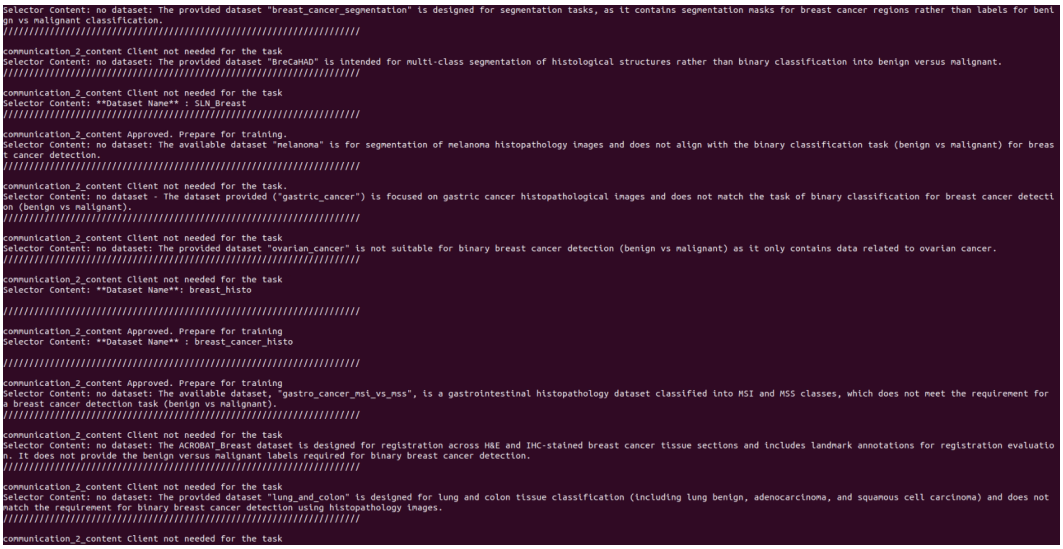


Figure 24: Client Selection with **histopathology datasets** for **non-thinking/reasoning models** on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

H Broader Social Impact

Positive Societal Impacts: FedAgentBench offers a significant advancement toward democratizing access to federated learning (FL) in healthcare by enabling agentic automation of otherwise labor-intensive tasks such as client selection, data cleaning, label harmonization, and FL algorithm configuration. This shift can particularly benefit healthcare institutions in low- and middle-income countries (LMICs) and rural areas, where hiring skilled data engineers is not feasible. By reducing the human expertise required to deploy and manage FL workflows, our benchmark promotes more equitable participation in collaborative medical AI initiatives. Furthermore, the open-source nature of our framework ensures broad accessibility, fostering reproducibility, community contributions, and educational use.

Negative Societal Impacts: The automation of critical decision-making steps in FL workflows, especially in high-stakes healthcare applications, raises concerns about over-reliance on large language models (LLMs) that may hallucinate or misinterpret complex clinical instructions. Incorrect data preprocessing or label harmonization could propagate errors into downstream training, leading to biased or unreliable models. Additionally, while the framework supports privacy-preserving learning, if misused without proper oversight, it may inadvertently reinforce data quality issues or amplify existing healthcare disparities. Careful human oversight and robust safety mechanisms must accompany such autonomous systems to ensure responsible deployment in real-world medical settings.

I LLM Usage:

LLMs (GPT-4o and GPT-5) were sometimes used for improving grammar and wording.

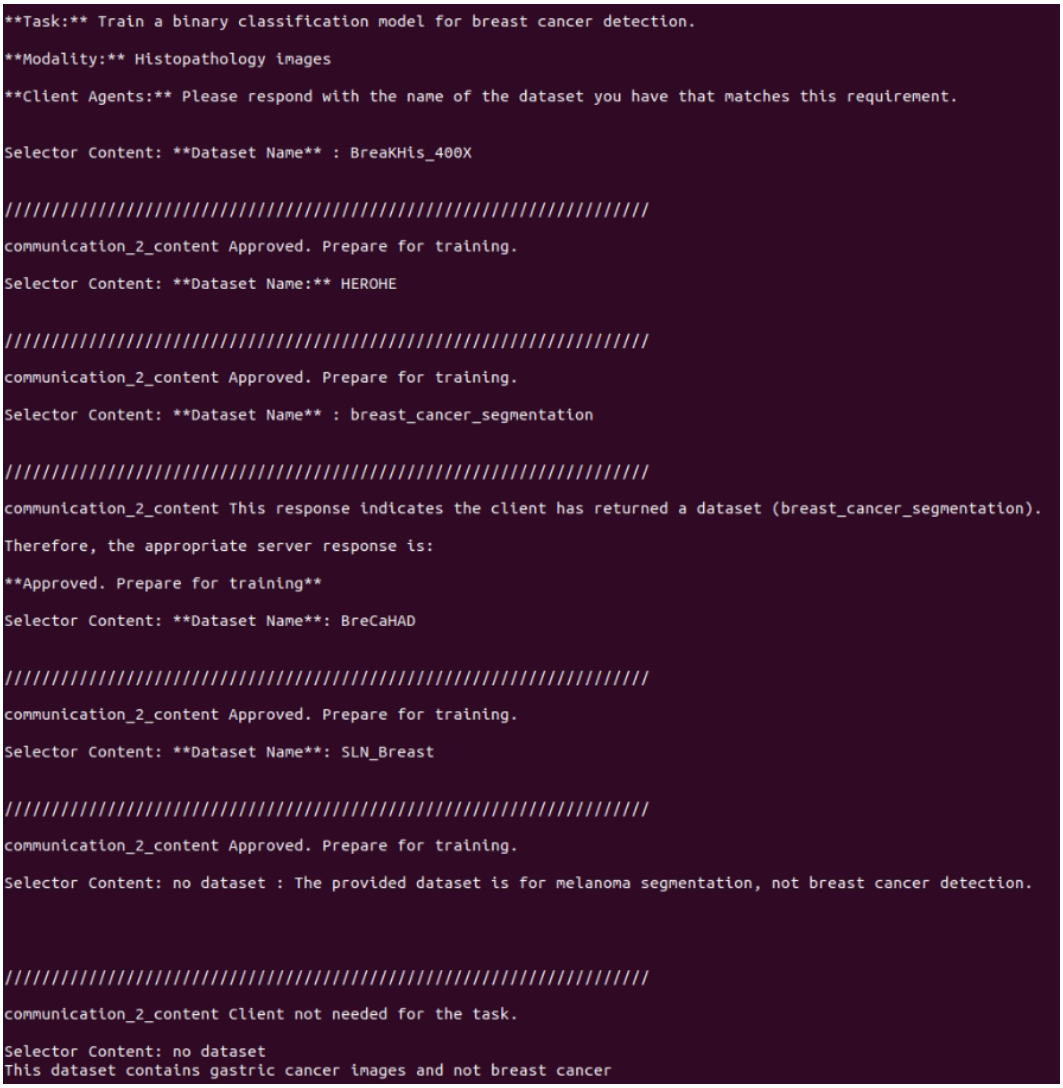


Figure 25: Client Selection with **histopathology datasets** for **non-thinking/reasoning** models on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

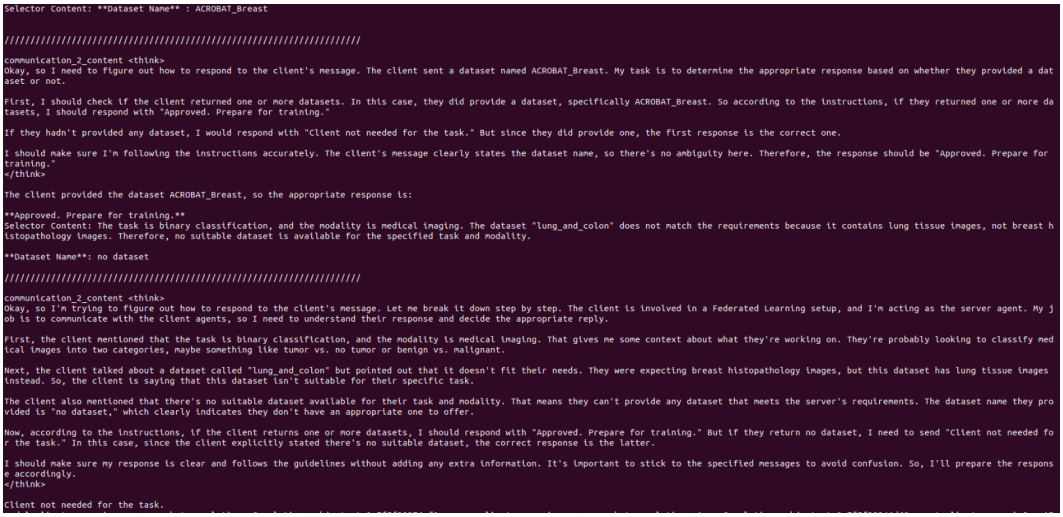


Figure 26: Client Selection with **histopathology datasets** for **thinking/reasoning models** on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

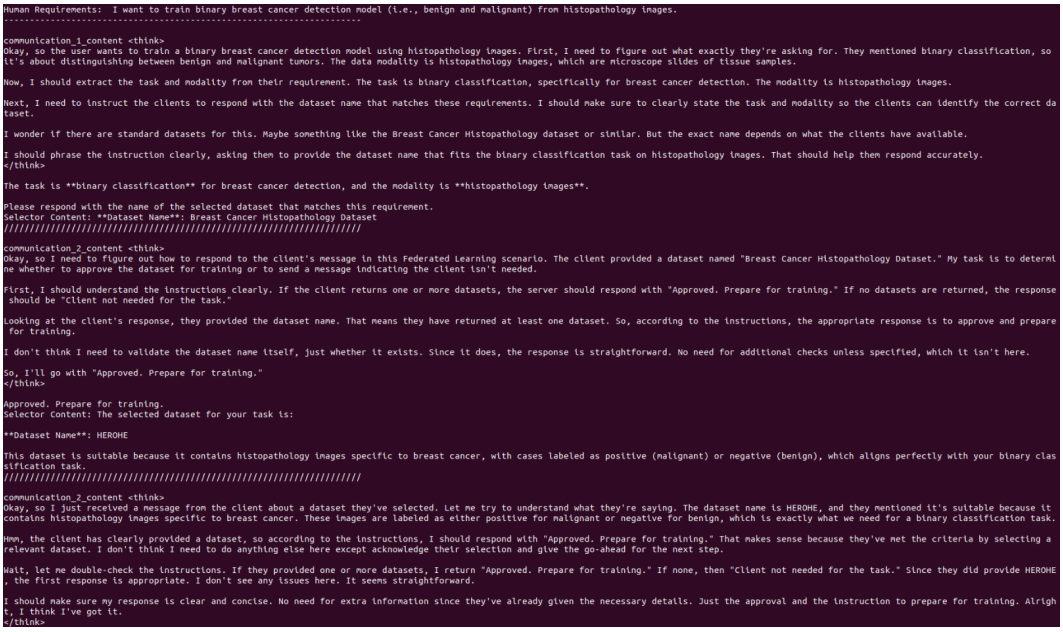


Figure 27: Client Selection with **histopathology datasets** for **thinking/reasoning models** on **breast cancer detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

Figure 28: Client Selection with **X-Ray datasets** for **non-thinking/reasoning** models **on pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

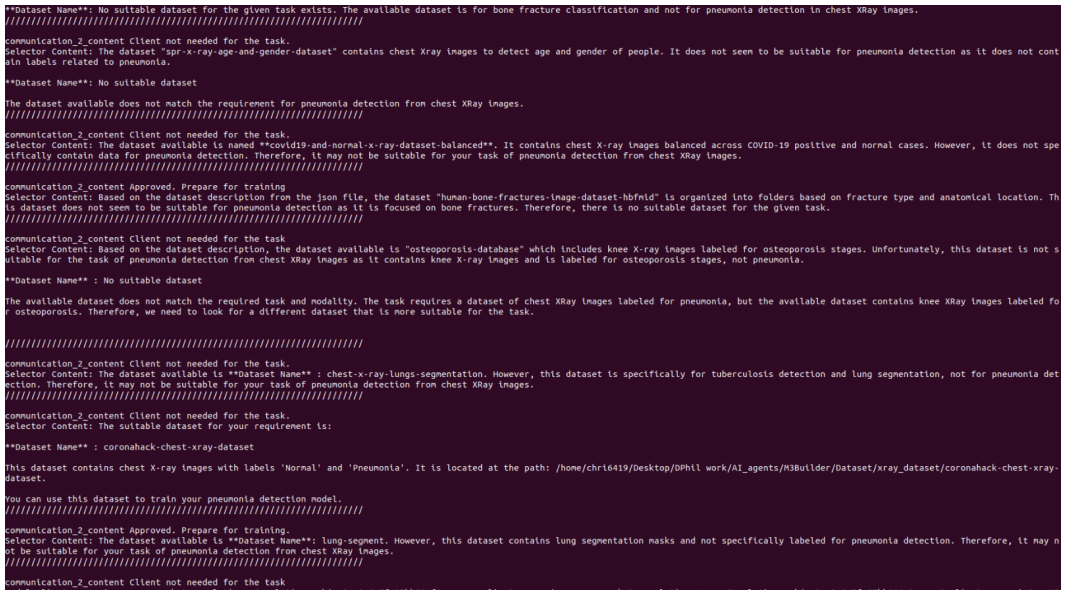


Figure 29: Client Selection with **X-Ray datasets** for **non-thinking/reasoning models** on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

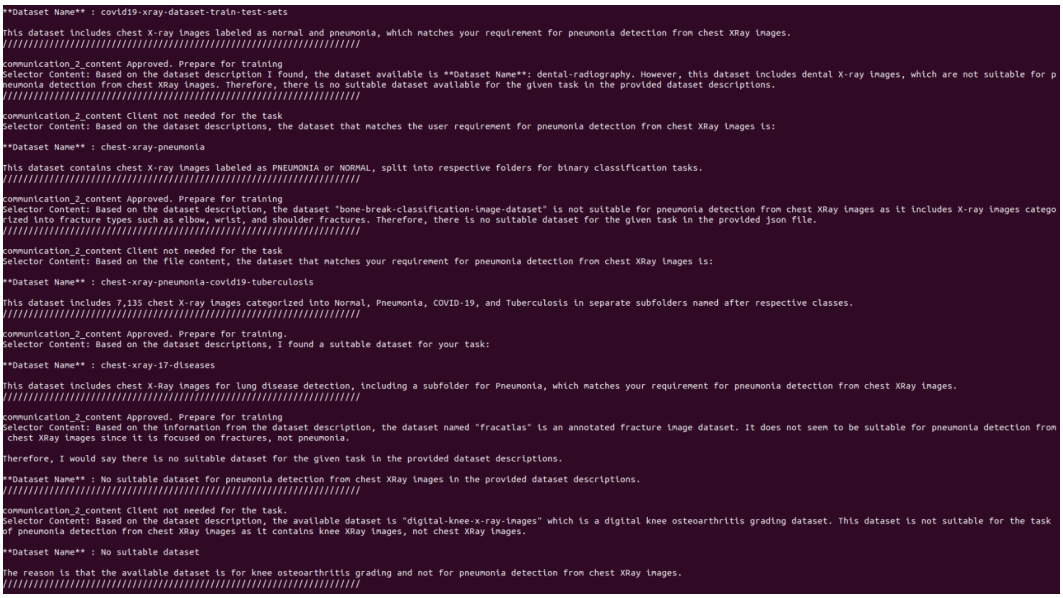


Figure 30: Client Selection with **X-Ray datasets** for **non-thinking/reasoning** models on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

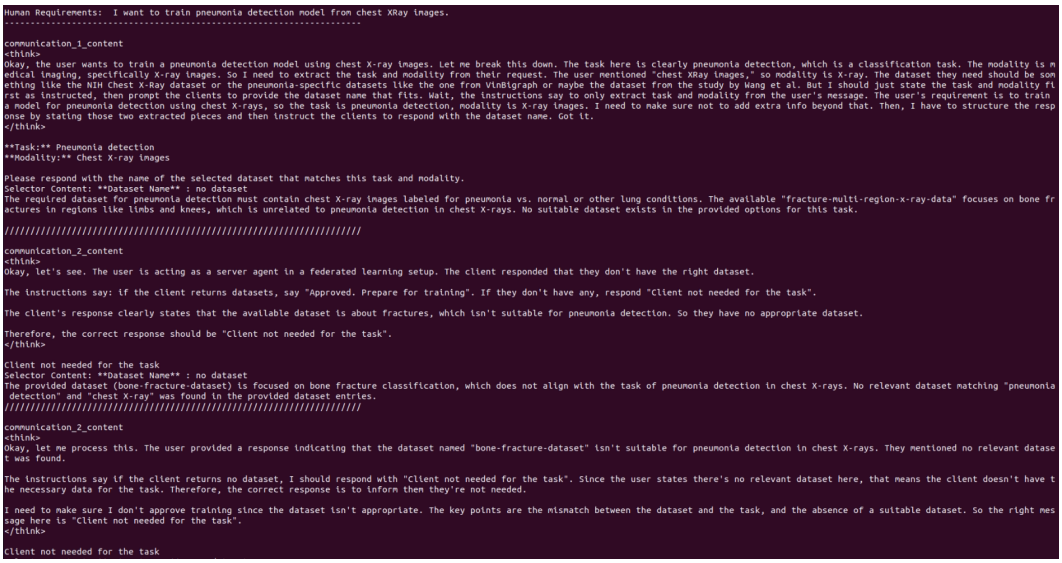


Figure 31: Client Selection with **X-Ray datasets** for **thinking/reasoning models** on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

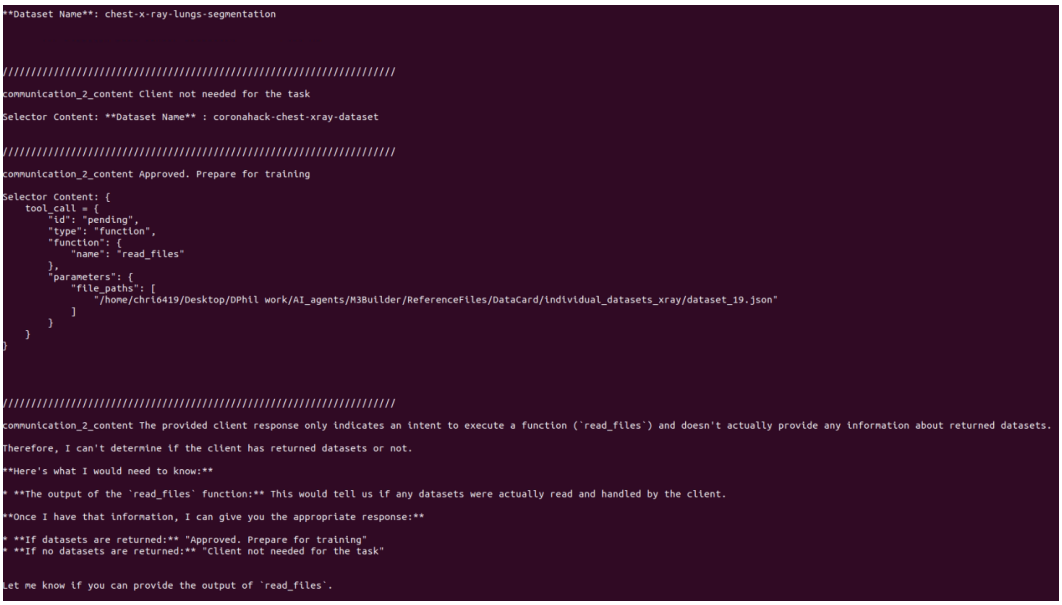


Figure 32: Client Selection with **X-Ray datasets** for **thinking/reasoning** models on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

Figure 33: Client Selection with **X-Ray datasets** for **thinking/reasoning models** on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

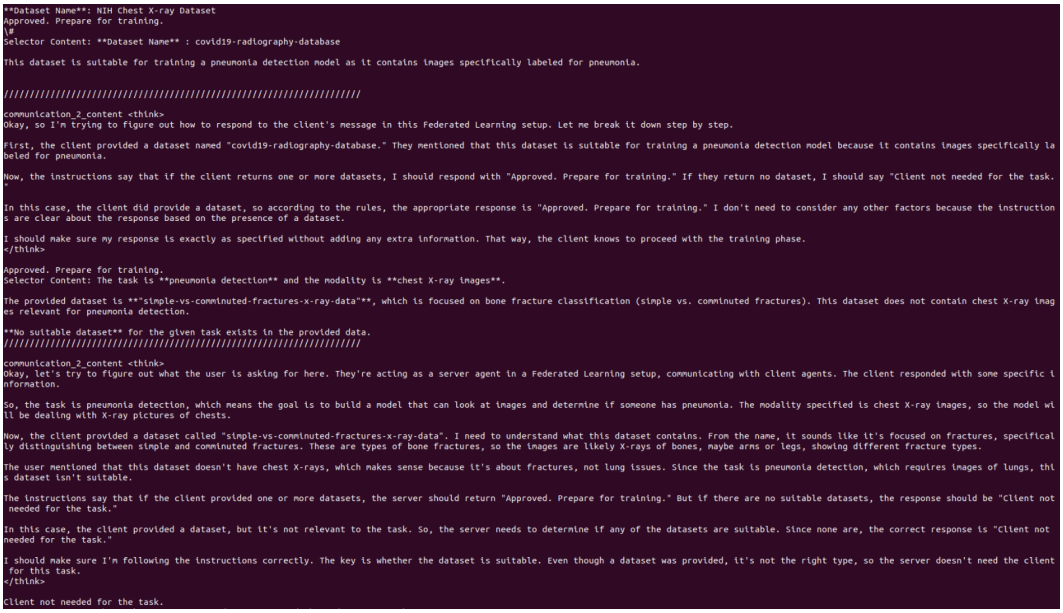


Figure 34: Client Selection with **X-Ray datasets** for **thinking/reasoning** models on **pneumonia detection task**. It shows when/how the server approves clients for training whereas when/how it declines.

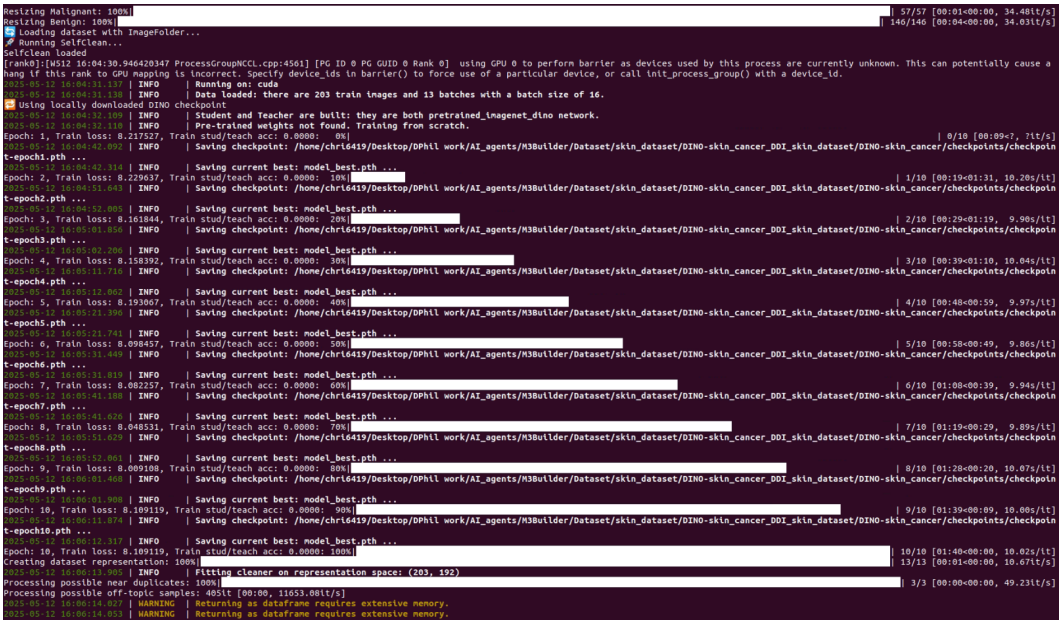


Figure 35: Data-cleaning by learning the representation space of DDI skin cancer dataset using DINO

Figure 36: Data-cleaning by learning the representation space of augmented-skin-condition-dataset-kaggle using DINO