Position: Agentic Federated Learning for AI-Driven Strategy Design and Optimization

Haoyuan Li [1] **Mathias Funk** [1] **Jindong Wang** [2] **Aaqib Saeed** [1]

Abstract

The rapid progress of large language models (LLMs) has gained considerable attention for their universal capability of conducting reasoning, decision-making across diverse domains. These advances have revolutionized natural language understanding tasks, particularly in code generation, contributing to the prosperity of developing autonomous agents for end-to-end programming in software engineering applications. Despite these recent successes, their application to the design of federated learning systems (FL) remains nascent. In this position paper, we advocate for an agentic FL paradigm that harnesses cooperating task-specialized LLM agents to automate the entire FL lifecycle. We outline a four-stage workflow in which planning, coding, and optimizing agents iteratively generate, refine, and validate FL strategies under a human-inspired development process. We emphasize open research directions to advance multi-agent FL systems that adaptively configure and manage real-world FL deployments.

1. Introduction

Recent advancements in large language models (LLMs) have catalyzed a breakthrough in building LLM-based agents for accomplishing complex, multi-step problems in code generation (Zhang et al., 2024b; Tao et al., 2024a; Islam et al., 2024; Nunez et al., 2024; Tao et al., 2024b). The development of LLM-based agents augments the functionality with access to external tools like APIs or search engines as knowledge bases to retrieve diverse information (Lewis et al., 2020; Chen et al., 2024b), showing remarkable capability in reasoning, planning, and solving programming challenges across a wide spectrum of domains (Li et al., 2024b; Tao et al., 2024b; Chang et al., 2024; Lange et al., 2025; Gandhi et al., 2025). However, their performance in agentic federated learning (AgenticFL) programming is not explored.

Existing Federated Learning (FL) approaches have primarily focused on developing robust and effective strategies to address a series of interrelated challenges encountered during collaborative and decentralized training between clients and servers (as summarized in Table 2). Typically, these methods target specific FL issues, such as data heterogeneity and communication efficiency, from a local or systemic perspective, thus addressing only isolated tasks within the FL context. In contrast, real-world deployments expose FL strategies to multifaceted challenges that static designs do not address. For instance, while many studies on evaluating the robustness of their proposed methods have concentrated on heterogeneous client data (i.e. non-IID distributions), practical scenarios reveal that heterogeneity manifests itself across devices (Zhang et al., 2024a; Jia et al., 2024; Xu et al., 2024), data distributions (Li et al., 2020; Reddi et al., 2020; Wang et al., 2020; Acar et al., 2021; Kim et al., 2022), model architectures (Diao et al., 2020; Kim et al., 2023; Li et al., 2022), task objectives (Marfoq et al., 2021; Chen & Zhang, 2022; Lu et al., 2024), and communication resources (Chen et al., 2020; Nguyen et al., 2022; Yu et al., 2023; Liu et al., 2024a). This bottleneck in current FL strategies underscores the need for LLM-powered programming agents capable of adaptively designing FL strategies that are ready for deployment in the wild.

A promising solution to tackle these challenges is to construct an agentic workflow to dynamically design FL strategies that can adapt to various user queries. Early approaches utilizing an LLM-powered coding agent for automatic code generation in application development (Muennighoff et al., 2023; Dong et al., 2024; Ridnik et al., 2024) by directly generating code from problem descriptions or sample I/O, employing prompting techniques such as chain-of-thought (Wei et al., 2022) and ReAct (Yao et al., 2023). Despite the success of LLMs in code generation, many of these single-agent methods struggle with complex problem solving tasks

[1]Department of Industrial Design, Eindhoven University of Technology, Eindhoven, Netherlands [2]Department of Arts & Sciences, College of William & Mary, Williamsburg, Virginia 23185, USA. Correspondence to: Haoyuan Li <h.y.li@tue.nl>.

Accepted at the ICML 2025 Workshop on Collaborative and Federated Agentic Workflows (CFAgentic@ICML'25), Vancouver, Canada. July 19, 2025.

as the input token increases (Li et al., 2023; Levy et al., 2024; Qian et al., 2024) in the context length of programming problems.

Many studies in LLMs have particularly contributed to the construction of multiagent frameworks to significantly enhance the performance of LLMs in complex problem solving tasks by facilitating collaborations between specialized autonomous agents (Yang et al., 2025; Zhang et al., 2025). In particular, in automated code generation, Huang et al. (2023) proposed AgentCoder that separates code generation from test case design and execution to enhance accuracy while reducing token overhead. Islam et al. (2024) emulate the human programming cycle by autonomously integrating retrieval, planning, coding, and debugging to improve competitive code synthesis. Ishibashi & Nishimura (2024) proposed self-organized agents that dynamically distribute code generation and optimization tasks among proliferating agents for large-scale code generation and optimization. Furthermore, recent research has revealed the capability of multi-agent collaboration for automatic programming in a long-term context for task-oriented code generation in robotics engineering (Wang et al., 2025) or scientific research (Gandhi et al., 2025; Hong et al., 2024). For example, Chen et al. (2024a); Liu et al. (2024b); Phan et al. (2024); Ma et al. (2024) adopt multi-agent processes to decouple programming roles based on contextual information to resolve real-world GitHub issues in SWE-bench (Jimenez et al., 2024), achieving superior performance compared to single-agent methods.

In this position paper, we first present four fundamental rationales that motivate the adoption of AgenticFL frameworks in FL design. In Section 3, we formalize the AgenticFL paradigm as a multi-agent workflow, introducing a traversal schema that mirrors the human programming cycle. We conclude by outlining promising avenues for future research, encouraging the community to further investigate how collaborative multi-agent systems can innovate and optimize FL architectures.

2. Motivation of Agentic FL Strategy Design

Rationale 1

Federated learning confronts multifaceted challenges in real-world applications.

Recent advances in FL have demonstrated its promise for decentralized learning across domains such as healthcare, finance, and personalized recommendations (Zhang et al., 2021; Imteaj & Amini, 2022; Zhou et al., 2024), yet real-world deployments (Fig. 1) must contend with stringent privacy requirements, device-level vulnerabilities, and model-integrity threats, especially in IoMT-based healthcare settings (Nguyen et al., 2022; Antunes et al., 2022), while most existing FL strategies address these challenges in isolation, highlighting the need for an integrated, ready-to-deploy framework for complex, multifaceted applications.

Rationale 2

Federated Learning strategies can benefit from complementary strategies.

Existing FL methods typically tackle isolated tasks while being able to enhance with complementary strategies. For example, in non-IID settings, Li et al. (2024a); Zhang et al. (2022); Jiang et al. (2022); Hu et al. (2024) integrate algorithms mitigating feature and label skew with established strategies such as FedProx (Li et al., 2020), FedNova (Wang et al., 2020), and SCAFFOLD (Karimireddy et al., 2020) to improve performance in heterogeneous environments, and Chen & Chao (2020); Dong et al. (2022); Li et al. (2022); Kim et al. (2022); Ye et al. (2023b;a) show that adding novel local-training regularizers and modular enhancements to existing FL strategies further bolsters model robustness and delivers significant benchmark gains.

Rationale 3

Federated Learning strategies are vulnerable to diverse scenarios or settings.

FL deployments face diverse client environments, data distributions, communication budgets, and computational constraints, causing existing FL strategies to behave unpredictably under limited communication rounds (Wu et al., 2022; Li et al., 2021a; Karimireddy et al., 2020; Mendieta et al., 2022), exhibit sensitivity to client resource heterogeneity (Ye et al., 2023b; Diao et al., 2020; Kim et al., 2023; Li et al., 2022), and suffer performance fluctuations from hyperparameter configurations and participation variability in client selection (Wu et al., 2023; Cho et al., 2022; Yan et al., 2023; Jhunjhunwala et al., 2022; Xu et al., 2021), thereby underscoring the need for adaptive AgenticFL strategy designs.

Rationale 4

Federated Learning deployment requires adaptive framework integration.

Federated learning frameworks range from research-oriented systems such as Flower (Beutel et al., 2020), OpenFL (Reina et al., 2021), and PySyft (Ziller et al., 2021), which offer modular environments for rapid prototyping, to industrial-oriented platforms such as FATE (Liu et al., 2021), IBM-FL (Ludwig et al., 2020), NVIDIA FLARE (Roth et al., 2022), and FedML (He et al., 2020), which emphasize scalability, robustness, and operational efficiency. This dichotomy motivates an agentic FL framework for adap-

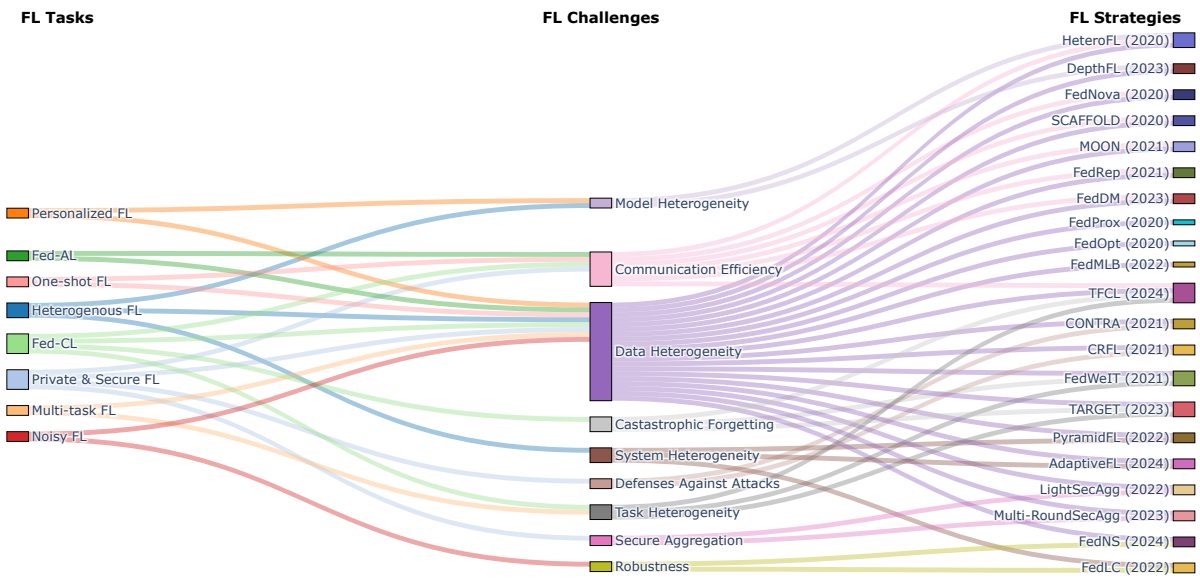

Figure 1: Overview of various Federated Learning (FL) approaches and their focus areas.

tive, on-demand, task-specific strategy design to combine experimental innovation with production-grade reliability.

3. Towards Multi-Agents Collaboration in Agentic FL Ecosystem

Figure 2 depicts our proposed AgenticFL workflow for designing and implementing agent systems. Inspired by the typical software engineering lifecycle, we organize the process into four successive stages—planning, programming, optimization, and deployment—each supported by specialized agents that collaborate according to the contextual information they observe.

3.1. Progressive Plan with User-in-the-Loop

The AgenticFL framework initiates with a multi-turn elaboration process to construct robust coding initiatives. This phase integrates user feedback through structured user-agent interactions, establishing meta-review criteria that guide subsequent coding processes. The formalized user feedback F_u informs the planning process P, such that $P(F_u) \rightarrow \{T_1, T_2, \ldots, T_n\}$, where each T_i represents a discrete coding task with defined objectives and constraints for a specific FL problem.

3.2. Iterative Codebase Generation and Validation

Once the specification is settled, a *coding agent* and a *debugging agent* operate as a tightly coupled pair within a single "worker group." The coder translates each sub-task into executable code modules, while the debugger executes the generated code against unit tests, integration tests, or simulation scripts. Within the worker group architecture, two specialized agents operate in tandem: Coder (A_c) and Debugger (A_d). These agents engage in an iterative coding–testing loop expressed as $\{A_c \leftrightarrow A_d\}^k$, where k represents the number of iteration cycles until convergence to a functionally correct codebase C that satisfies the predefined task specifications and validation criteria.

3.3. Self-Debating for Optimizing coding candidates

The optimization phase employs a reflection agent A_r that operates in a dual-stage manner. Initially, A_r determines optimization directions aligned with federated learning objectives (e.g., global convergence rate, communication efficiency). Subsequently, A_r performs structured self-critique to generate detailed optimization proposals $P = \{p_1, p_2, \ldots, p_n\}$. A committee of reviewer agents powered by different LLMs $\{R_1, R_2, \ldots, R_m\}$ evaluates these candidates using a LLM-based metric $G(\cdot)$ (e.g., G-EVAL (Liu et al., 2023)) to rank the coding proposals from multiple dimensions including novelty, correctness, and testability.

$$P_k = \underset{P' \subset P, |P'|=k}{\arg\max} \sum_{p_i \in P'} \frac{1}{m} \sum_{j=1}^{m} G(p_i, R_j) \qquad (1)$$

The top-k candidates P_k are forwarded to the worker group $\{A_c, A_d\}$ for implementation as executable artifacts

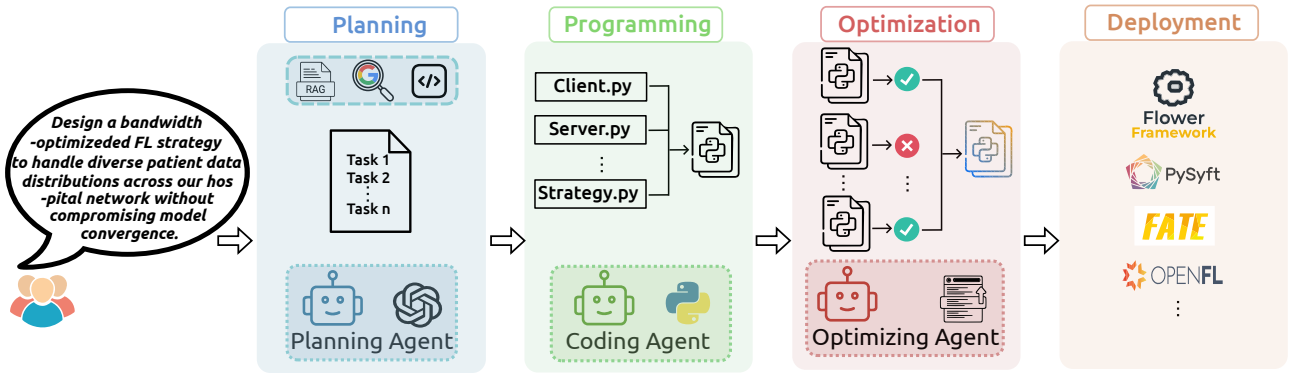

Figure 2: A general overview of agentic federated learning auto-programming workflow.

$\{a_1, a_2, ..., a_k\}$. These artifacts undergo execution-based evaluation against a standardized FL simulation (i.e., client datasets and network conditions) and measure empirically relevant performance indicators (e.g., test accuracy, total communication rounds, end-to-end runtime). The artifact that demonstrates optimal performance in these metrics is selected as the final optimized output.

3.4. General Workflow of Agentic FL Programming

Planning. The AgenticFL workflow is initiated by a *planning agent*, which transforms the user's high-level specification into a precise, structured set of FL design specification. First, the agent decomposes the query into core requirements, such as choice of model architecture, privacy budget, client heterogeneity, and communication resource, and encodes these as structured planning objectives. To ground its decisions, the agent dynamically issues API calls to domain-specific tools, consults web search results for algorithmic best practices, and retrieves analogous deployments and protocol templates from a Retrieval-Augmented Generation (RAG) database. By integrating these exemplars and blueprints, the planner produces a coherent development roadmap that specifies modular subtasks, recommended hyperparameter ranges, and fallback strategies. This enriched, context-aware prompt is then passed forward to downstream agents to guide programming and optimization.

Programming. Next, the resulting blueprint is forwarded to *coding agents*. Conditioned on the planner's specification, the coder emits an executable but minimal codebase that comprises client-side training loops, server-side aggregation logic, and auxiliary infrastructure (data loaders, logging hooks, and differential privacy). The agent maintains modularity by encapsulating each functional component in a separate module or class.

Optimization. Once a baseline implementation is available, a suite of candidate refinements is produced at the optimization stage with adaptive aggregation schedules, compression schemes. These candidates are evaluated via simulated FL rounds on representative data partitions to assess each variant's empirical performance (e.g., convergence speed, communication overhead, privacy leakage) and selects the most promising configuration as the optimized output.

Deployment. Finally, the optimized artifact is passed to a deployment sub-routine. Depending on the target environment (e.g., research prototype or production), the final optimization output is transformed into a task-specific package for one of the major FL frameworks (e.g., Flower (Beutel et al., 2020), PySyft (Ziller et al., 2021), or NVIDIA FLARE (Roth et al., 2022)). This adaptation entails translating generic training loops and aggregation routines into the framework's native API calls, provisioning execution environments via containerization, and generating orchestration scripts to coordinate distributed clients.

4. Conclusion

Through this paper, we propose a principled AgenticFL framework for FL design, inspired by the human software-development cycle. We first motivate the necessity of adopting multi-agent collaboration in FL through several key rationales, followed by introducing a structured workflow comprising planning, programming, optimization, and deployment stages, each executed by specialized autonomous agents. We highlighted key functional roles for each agent, discussed their interplay across the FL lifecycle.

Future work includes developing the AgenticFL framework with role-based agents based on FL task requirements, conducting empirical studies on the efficiency of different task specifications, and investigating adaptive agent behaviors that respond to dynamic federated environments. We hope this work inspires the community to further investigate and refine multi-agent collaborative methodologies, ultimately advancing the efficiency, flexibility, and scalability of automated agentic FL system design.

References

Acar, D. A. E., Zhao, Y., Navarro, R. M., Mattina, M., Whatmough, P. N., and Saligrama, V. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

Antunes, R. S., André da Costa, C., Küderle, A., Yari, I. A., and Eskofier, B. Federated learning for healthcare: Systematic review and architecture proposal. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13 (4):1–23, 2022.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Li, K. H., Parcollet, T., de Gusmão, P. P. B., et al. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

Chang, C.-C., Ho, C.-T., Li, Y., Chen, Y., and Ren, H. Drc-coder: Automated drc checker code generation using llm autonomous agent. *arXiv preprint arXiv:2412.05311*, 2024.

Chen, D., Lin, S., Zeng, M., Zan, D., Wang, J.-G., Cheshkov, A., Sun, J., Yu, H., Dong, G., Aliev, A., et al. Coder: Issue resolving with multi-agent and task graphs. *arXiv preprint arXiv:2406.01304*, 2024a.

Chen, H.-Y. and Chao, W.-L. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*, 2020.

Chen, J. and Zhang, A. Fedmsplit: Correlation-adaptive federated multi-task learning across multimodal split networks. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 87–96, 2022.

Chen, Y., Ning, Y., Slawski, M., and Rangwala, H. Asynchronous online federated learning for edge devices with non-iid data. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 15–24. IEEE, 2020.

Chen, Z., Xiang, Z., Xiao, C., Song, D., and Li, B. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2024b.

Cho, Y. J., Wang, J., and Joshi, G. Towards understanding biased client selection in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 10351–10375. PMLR, 2022.

Collins, L., Hassani, H., Mokhtari, A., and Shakkottai, S. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pp. 2089–2099. PMLR, 2021.

Diao, E., Ding, J., and Tarokh, V. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. *arXiv preprint arXiv:2010.01264*, 2020.

Dong, X., Zhang, S. Q., Li, A., and Kung, H. Spherefed: Hyperspherical federated learning. In *European Conference on Computer Vision*, pp. 165–184. Springer, 2022.

Dong, Y., Jiang, X., Jin, Z., and Li, G. Self-collaboration code generation via chatgpt. *ACM Transactions on Software Engineering and Methodology*, 33(7):1–38, 2024.

Gandhi, S., Shah, D. B., Patwardhan, M., Vig, L., and Shroff, G. Researchcodeagent: An llm multi-agent system for automated codification of research methodologies. In *2nd AI4Research Workshop: Towards a Knowledge-grounded Scientific Research Lifecycle*, 2025.

He, C., Li, S., So, J., Zeng, X., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

Hong, S., Lin, Y., Liu, B., Liu, B., Wu, B., Zhang, C., Wei, C., Li, D., Chen, J., Zhang, J., et al. Data interpreter: An llm agent for data science. *arXiv preprint arXiv:2402.18679*, 2024.

Hsu, T.-M. H., Qi, H., and Brown, M. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Hu, K., Xiang, L., Tang, P., and Qiu, W. Feature norm regularized federated learning: utilizing data disparities for model performance gains. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 4136–4146, 2024.

Huang, D., Zhang, J. M., Luck, M., Bu, Q., Qing, Y., and Cui, H. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation. *arXiv preprint arXiv:2312.13010*, 2023.

Hyeon-Woo, N., Ye-Bin, M., and Oh, T.-H. Fedpara: Low-rank hadamard product for communication-efficient federated learning. *arXiv preprint arXiv:2108.06098*, 2021.

Imteaj, A. and Amini, M. H. Leveraging asynchronous federated learning to predict customers financial distress. *Intelligent Systems with Applications*, 14:200064, 2022.

Ishibashi, Y. and Nishimura, Y. Self-organized agents: A llm multi-agent framework toward ultra large-scale code generation and optimization. *arXiv preprint arXiv:2404.02183*, 2024.

Islam, M. A., Ali, M. E., and Parvez, M. R. Mapcoder: Multi-agent code generation for competitive problem solving. *arXiv preprint arXiv:2405.11403*, 2024.

Jhunjhunwala, D., Sharma, P., Nagarkatti, A., and Joshi, G. Fedvarp: Tackling the variance due to partial client participation in federated learning. In *Uncertainty in Artificial Intelligence*, pp. 906–916. PMLR, 2022.

Jia, C., Hu, M., Chen, Z., Yang, Y., Xie, X., Liu, Y., and Chen, M. Adaptivefl: Adaptive heterogeneous federated learning for resource-constrained aiot systems. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pp. 1–6, 2024.

Jiang, X., Sun, S., Wang, Y., and Liu, M. Towards federated learning against noisy labels via local self-regularization. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 862–873, 2022.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. R. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Kim, J., Kim, G., and Han, B. Multi-level branched regularization for federated learning. In *International Conference on Machine Learning*, pp. 11058–11073. PMLR, 2022.

Kim, M., Yu, S., Kim, S., and Moon, S.-M. Depthfl: Depthwise federated learning for heterogeneous clients. In *The Eleventh International Conference on Learning Representations*, 2023.

Lange, R. T., Prasad, A., Sun, Q., Faldor, M., Tang, Y., and Ha, D. The ai cuda engineer: Agentic cuda kernel discovery, optimization and composition. 2025.

Levy, M., Jacoby, A., and Goldberg, Y. Same task, more tokens: the impact of input length on the reasoning performance of large language models. *arXiv preprint arXiv:2402.14848*, 2024.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Li, C., Zeng, X., Zhang, M., and Cao, Z. Pyramidfl: A fine-grained client selection framework for efficient federated learning. In *Proceedings of the 28th annual international conference on mobile computing and networking*, pp. 158–171, 2022.

Li, H., Funk, M., Gürel, N. M., and Saeed, A. Collaboratively learning federated models from noisy decentralized data. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 7879–7888. IEEE, 2024a.

Li, J., Wang, M., Zheng, Z., and Zhang, M. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*, 2023.

Li, Q., He, B., and Song, D. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10713–10722, 2021a.

Li, R., Wang, X., and Yu, H. Exploring llm multi-agents for icd coding. *arXiv preprint arXiv:2406.15363*, 2024b.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Li, X., Jiang, M., Zhang, X., Kamp, M., and Dou, Q. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021b.

Liu, J., Jia, J., Che, T., Huo, C., Ren, J., Zhou, Y., Dai, H., and Dou, D. Fedasmu: Efficient asynchronous federated learning with dynamic staleness-aware model update. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13900–13908, 2024a.

Liu, Y., Fan, T., Chen, T., Xu, Q., and Yang, Q. Fate: An industrial grade platform for collaborative learning with data protection. *Journal of Machine Learning Research*, 22(226):1–6, 2021.

Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., and Zhu, C. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*, 2023.

Liu, Y., Gao, P., Wang, X., Liu, J., Shi, Y., Zhang, Z., and Peng, C. Marscode agent: Ai-native automated bug fixing. *arXiv preprint arXiv:2409.00899*, 2024b.

Lu, Y., Huang, S., Yang, Y., Sirejiding, S., Ding, Y., and Lu, H. Fedhca2: Towards hetero-client federated multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5599–5609, 2024.

Ludwig, H., Baracaldo, N., Thomas, G., Zhou, Y., Anwar, A., Rajamoni, S., Ong, Y., Radhakrishnan, J., Verma, A., Sinn, M., et al. Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987*, 2020.

Ma, Y., Yang, Q., Cao, R., Li, B., Huang, F., and Li, Y. Alibaba lingmaagent: Improving automated issue resolution via comprehensive repository exploration. *arXiv preprint arXiv:2406.01422*, 2024.

Marfoq, O., Neglia, G., Bellet, A., Kameni, L., and Vidal, R. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Mendieta, M., Yang, T., Wang, P., Lee, M., Ding, Z., and Chen, C. Local learning matters: Rethinking data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8397–8406, 2022.

Muennighoff, N., Liu, Q., Zebaze, A., Zheng, Q., Hui, B., Zhuo, T. Y., Singh, S., Tang, X., Von Werra, L., and Longpre, S. Octopack: Instruction tuning code large language models. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023.

Nguyen, D. C., Pham, Q.-V., Pathirana, P. N., Ding, M., Seneviratne, A., Lin, Z., Dobre, O., and Hwang, W.-J. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (Csur)*, 55(3):1–37, 2022.

Nunez, A., Islam, N. T., Jha, S. K., and Najafirad, P. Autosafecoder: A multi-agent framework for securing llm code generation through static analysis and fuzz testing. *arXiv preprint arXiv:2409.10737*, 2024.

Phan, H. N., Nguyen, T. N., Nguyen, P. X., and Bui, N. D. Hyperagent: Generalist software engineering agents to solve coding tasks at scale. *arXiv preprint arXiv:2409.16299*, 2024.

Qian, H., Liu, Z., Zhang, P., Mao, K., Zhou, Y., Chen, X., and Dou, Z. Are long-llms a necessity for long-context tasks? *arXiv preprint arXiv:2405.15318*, 2024.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Reina, G. A., Gruzdev, A., Foley, P., Perepelkina, O., Sharma, M., Davidyuk, I., Trushkin, I., Radionov, M., Mokrov, A., Agapov, D., et al. Openfl: An open-source framework for federated learning. *arXiv preprint arXiv:2105.06413*, 2021.

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., and Pedarsani, R. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International conference on artificial intelligence and statistics*, pp. 2021–2031. PMLR, 2020.

Ridnik, T., Kredo, D., and Friedman, I. Code generation with alphacodium: From prompt engineering to flow engineering. *arXiv preprint arXiv:2401.08500*, 2024.

Roth, H. R., Cheng, Y., Wen, Y., Yang, I., Xu, Z., Hsieh, Y.-T., Kersten, K., Harouni, A., Zhao, C., Lu, K., et al. Nvidia flare: Federated learning from simulation to real-world. *arXiv preprint arXiv:2210.13291*, 2022.

Tao, L., Chen, X., Yu, T., Mai, T., Rossi, R., Li, Y., and Mitra, S. Codelutra: Boosting llm code generation via preference-guided refinement. *arXiv preprint arXiv:2411.05199*, 2024a.

Tao, W., Zhou, Y., Wang, Y., Zhang, W., Zhang, H., and Cheng, Y. Magis: Llm-based multi-agent framework for github issue resolution. *Advances in Neural Information Processing Systems*, 37:51963–51993, 2024b.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Wang, X., Dong, L., Rangasrinivasan, S., Nwogu, I., Setlur, S., and Govindaraju, V. Automisty: A multi-agent llm framework for automated code generation in the misty social robot. *arXiv preprint arXiv:2503.06791*, 2025.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wu, C., Wu, F., Lyu, L., Huang, Y., and Xie, X. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1):2032, 2022.

Wu, F., Guo, S., Qu, Z., He, S., Liu, Z., and Gao, J. Anchor sampling for federated learning with partial client participation. In *International Conference on Machine Learning*, pp. 37379–37416. PMLR, 2023.

Xiong, Y., Wang, R., Cheng, M., Yu, F., and Hsieh, C.-J. Feddm: Iterative distribution matching for communication-efficient federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16323–16332, 2023.

Xu, J., Wang, S., Wang, L., and Yao, A. C.-C. Fedcm: Federated learning with client-level momentum. *arXiv preprint arXiv:2106.10874*, 2021.

Xu, Z., Xu, M., Liao, T., Zheng, Z., and Chen, C. Fedbrb: An effective solution to the small-to-large scenario in device-heterogeneity federated learning. *arXiv preprint arXiv:2402.17202*, 2024.

Yan, G., Wang, H., Yuan, X., and Li, J. Criticalfl: A critical learning periods augmented client selection framework for efficient federated learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2898–2907, 2023.

Yang, S., Li, Y., Lam, W., and Cheng, Y. Multi-llm collaborative search for complex problem solving. *arXiv preprint arXiv:2502.18873*, 2025.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Ye, R., Xu, M., Wang, J., Xu, C., Chen, S., and Wang, Y. Feddisco: Federated learning with discrepancy-aware collaboration. In *International Conference on Machine Learning*, pp. 39879–39902. PMLR, 2023a.

Ye, R., Zhu, X., Chai, J., Chen, S., and Wang, Y. Federated learning empowered by generative content. *arXiv preprint arXiv:2312.05807*, 2023b.

Yu, X., Cherkasova, L., Vardhan, H., Zhao, Q., Ekaireb, E., Zhang, X., Mazumdar, A., and Rosing, T. Asynchfl: Efficient and robust asynchronous federated learning in hierarchical iot networks. In *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*, pp. 236–248, 2023.

Zhang, J., Li, Z., Li, B., Xu, J., Wu, S., Ding, S., and Wu, C. Federated learning with label distribution skew via logits calibration. In *International Conference on Machine Learning*, pp. 26311–26329. PMLR, 2022.

Zhang, J., Zeng, S., Zhang, M., Wang, R., Wang, F., Zhou, Y., Liang, P. P., and Qu, L. Flhetbench: Benchmarking device and state heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12098–12108, 2024a.

Zhang, Q.-W., Li, F., Wang, J., Qiao, L., Yu, Y., Yin, D., and Sun, X. Factguard: Leveraging multi-agent systems to generate answerable and unanswerable questions for enhanced long-context llm extraction. *arXiv preprint arXiv:2504.05607*, 2025.

Zhang, W., Li, X., Ma, H., Luo, Z., and Li, X. Federated learning for machinery fault diagnosis with dynamic validation and self-supervision. *Knowledge-Based Systems*, 213:106679, 2021.

Zhang, Y., Pan, Y., Wang, Y., and Cai, J. Pybench: Evaluating llm agent on various real-world coding tasks. *arXiv preprint arXiv:2407.16732*, 2024b.

Zhou, L., Wang, M., and Zhou, N. Distributed federated learning-based deep learning model for privacy mri brain tumor detection. *arXiv preprint arXiv:2404.10026*, 2024.

Ziller, A., Trask, A., Lopardo, A., Szymkow, B., Wagner, B., Bluemke, E., Nounahon, J.-M., Passerat-Palmbach, J., Prakash, K., Rose, N., et al. Pysyft: A library for easy federated learning. *Federated learning systems: Towards next-generation AI*, pp. 111–139, 2021.

A. Appendix

A.1. Overview of existing FL strategies across different tasks

Table 1 summarizes a representative set of FL algorithms, highlighting the primary challenges each method addresses (e.g., statistical heterogeneity, communication efficiency, and personalization) and distilling their core technical innovations. For clarity, we annotate each method with its evaluation objective: top-1 test accuracy (◆), communication rounds/time (★), and parameter transmission volume (▼). This overview situates our agentic workflow within the broader landscape of FL research and underscores the diverse strategies that agents may leverage or augment in practice.

Table 1: Overview of federated learning (FL) baselines, the challenges they address, and their key approaches. Evaluation objectives are indicated by symbols: ◆ denotes top-1 test accuracy, ★ denotes communication rounds/time, and ▼ denotes parameter transmission.

Method	Features	Description
FedAvg ◆ (McMahan et al., 2017)	basic aggregation	Averages local model updates on the server to form a global model.
FedAvgM ◆ (Hsu et al., 2019)	Non-IID data	Momentum-based variant of FedAvg for handling heterogeneous data distributions.
FedProx ◆ (Li et al., 2020)	Non-IID data	Incorporates a proximal term to stabilize local training under data heterogeneity.
FedDyn ◆ (Acar et al., 2021)	Non-IID data	Dynamically adjusts local objectives to counteract distribution drift.
FedMLB ◆ (Kim et al., 2022)	Non-IID data	Employs multi-level branched regularization with online knowledge distillation to align representations.
SCAFFOLD ◆ (Karimireddy et al., 2020)	Non-IID data, convergence rate	Utilizes control variates to mitigate client drift for improved convergence.
FedNova ◆ (Wang et al., 2020)	Non-IID data, convergence rate	Normalizes local updates to address objective inconsistency and enhance convergence speed.
FedOpt ◆ (Reddi et al., 2020)	Non-IID data, convergence rate	Leverages advanced server optimizers (e.g., Adam) to accelerate global model convergence.
FedBN ◆ (Li et al., 2021b)	Non-IID data, convergence rate	Excludes local Batch Normalization parameters from aggregation to mitigate feature shift.
MOON ◆ (Li et al., 2021a)	Non-IID data, convergence rate	Applies model-level contrastive learning to align local updates with the global model, reducing drift.
HeteroFL ◆ (Diao et al., 2020)	model heterogeneity	Distributes subnetworks adaptively to handle varied model complexities for efficient aggregation.
DepthFL ◆ (Kim et al., 2023)	model heterogeneity	Uses depth scaling with mutual self-distillation among sub-classifiers to aggregate heterogeneous models.
FedPara ◆★ (Hyeon-Woo et al., 2021)	communication cost	Re-parameterizes network layers via a low-rank Hadamard product to reduce communication overhead.
FedPAQ ◆★▼ (Reisizadeh et al., 2020)	Non-IID data, communication	Adopts periodic averaging, partial participation, and quantized updates to lower communication costs.
FedKD ◆★▼ (Wu et al., 2022)	communication cost	Employs adaptive mutual distillation with dynamic gradient compression to mitigate communication overhead.
FedDM ◆★ (Xiong et al., 2023)	communication cost	Utilizes iterative distribution matching with synthesized local surrogate functions to approximate loss landscapes and reduce rounds.

Method	Features	Description
FedRep ◆ (Collins et al., 2021)	personalization	Learns a shared low-dimensional representation with personalized heads for efficient model adaptation.
FedNS ◆ (Li et al., 2024a)	Non-IID data, noisy data	Integrates gradient norm-based detection to handle corrupt or unreliable client data.

A.2. Overview of FL Research Areas and Key Challenges

Table 2 illustrates the principal research topics in FL, identifies the key technical challenges associated with each topic, and provides a concise description of their focus. This table serves as a reference for situating our proposed agentic workflow within the broader landscape of FL research.

Table 2: Overview of federated learning (FL) topics and their associated challenges

FL Topics	Main Challenges	Description
Private & Secure FL	Communication Efficiency, Security & Privacy	Focuses on reducing communication overhead and ensuring data confidentiality through encryption and secure aggregation protocols.
Heterogeneous FL	Data and Model Heterogeneity	Addresses the issues arising from distributed clients with heterogeneous resources (e.g., data, device) to improve global model generalizability.
Personalized FL	Data Heterogeneity, Model Personalization	Tailors the global model to individual client needs, mitigating the adverse effects of statistical heterogeneity.
Multi-task FL	Data Heterogeneity	Leverages multi-task learning to simultaneously learn shared and task-specific representations, effectively managing heterogeneous data.
Fed-AL	Data Heterogeneity	Integrates active learning to select informative samples, thereby reducing the negative impact of limited labeled data resources on distributed clients.
Fed-CL	Data Heterogeneity, Catastrophic Forgetting	Employs continual learning strategies to handle sequential data tasks while preventing the loss of previously learned knowledge.
Robust FL	Data Heterogeneity, Noisy Data	Implements robust aggregation methods to counteract the influence of noisy inputs from input or label space, enhancing model stability.
One-shot FL	Communication Efficiency, Data Heterogeneity	Aims to minimize communication rounds by compressing and efficiently aggregating updates while addressing diverse data characteristics.