AGENTIC FEDERATED LEARNING: THE FUTURE OF DISTRIBUTED TRAINING ORCHESTRATION

Rafael O. Jarczewski* **Gabriel U. Talasso*** **Leandro Villas*** **Allan M. de Souza***

ABSTRACT

Although Federated Learning (FL) promises privacy and distributed collaboration, its effectiveness in real-world scenarios is often hampered by the stochastic heterogeneity of clients and unpredictable system dynamics. Existing static optimization approaches fail to adapt to these fluctuations, resulting in resource underutilization and systemic bias. In this work, we propose a paradigm shift towards Agentic-FL, a framework where Language Model-based Agents (LMagents) assume autonomous orchestration roles. Unlike rigid protocols, we demonstrate how server-side agents can mitigate selection bias through contextual reasoning, while client-side agents act as local guardians, dynamically managing privacy budgets and adapting model complexity to hardware constraints. More than just resolving technical inefficiencies, this integration signals the evolution of FL towards decentralized ecosystems, where collaboration is negotiated autonomously, paving the way for future markets of incentive-based models and algorithmic justice. We discuss the reliability (hallucinations) and security challenges of this approach, outlining a roadmap for resilient multi-agent systems in federated environments.

1 INTRODUCTION

Federated Learning (FL) has emerged as a paradigm for training Machine Learning (ML) and Artificial Intelligence (AI) models based on distributed and collaborative learning among multiple clients in a network McMahan et al. (2017), where models are trained locally, and only the weights are shared with the central server to produce a global model. It enables improvements in learning by leveraging more computational resources and data at the edge, without compromising participants' privacy by sending raw data to a central server. Within this field, several challenges have been explored in recent years, such as data heterogeneity, since different sources contribute to the same global model, and computational and communication limitations, as edge devices often struggle with the intensive training of large models Kairouz et al. (2021a).

These challenges have driven numerous research efforts aiming to address them through various strategies. For instance, client selection methods focus on the utility and necessity of each client at every round, avoiding unnecessary computational costs de Souza et al. (2024); Fu et al. (2023); Jarczewski et al. (2024); Lai et al. (2021). Optimized model aggregation techniques mitigate divergence issues caused by heterogeneous data Li et al. (2021); Nguyen et al. (2024); Salehi et al. (2024); Kang et al. (2024b), while model personalization approaches tailor the global model to meet the specific needs of each client Tan et al. (2022); Piao et al. (2024); Talasso et al. (2024). On the resource side, solutions aim to make FL more efficient through model quantization, which reduces weight precision to lighten training and communication costs, model pruning, which removes redundant parameters, and additional strategies such as Parameter-Efficient Fine-Tuning (PEFT) Xu et al. (2023b); Ding et al. (2023) adapted to FL for recent generative models Ye et al. (2024); Zhang et al. (2024).

However, despite the progress made in mitigating these challenges, the inherent dynamics, complexity, and heterogeneity of distributed FL require holistic and adaptable methods to manage the evolving demands of the system. Approaches focusing on a single aspect, such as advanced aggregation

*Institute of Computing, University of Campinas and Hub of Artificial Intelligence and Cognitive Architectures - H.IAAC, Campinas, Brazil

mechanisms to improve performance, can inadvertently increase latency or computational load on client devices Li et al. (2025). Similarly, strategies centered solely on client selection or model personalization often overlook broader impacts on energy consumption, communication costs, or total training time. Moreover, as dynamic environments evolve, client and federation requirements, such as resource constraints or data drift, may change over time, rendering static solutions ineffective Criado et al. (2022). Fixed, single-purpose approaches are hampered by the complexity and dynamism that make solving real-world problems difficult, highlighting the necessity for autonomous and flexible systems.

In this context, we present this work as a position paper to show how integrating Language Model Agents (LM-Agents) into FL enables the development of flexible and adaptable solutions. Yao et al. (2023b). LM-Agents are systems capable of perceiving their environment, analyzing contextual information, and making dynamic decisions, while continuously improving through feedback. We argue that this adaptive and autonomous capability enables a more holistic understanding of the FL environment, considering aspects such as resource allocation, data heterogeneity, client performance, and dynamic participation. Additionally, we show, through a proof-of-concept, that this approach can leverage several state-of-the-art solutions at the appropriate time, based on the requirements of clients. Accordingly, the main contributions of this paper are summarized as follows:

- We motivate and argue for the benefits of using LM-Agents in FL, showing how agentic systems hold great potential for addressing unresolved challenges in distributed scenarios due to their complexity and dynamicity.
- We discuss the key challenges and open directions for employing agentic systems in the orchestration of federated systems, which can be explored in future developments in both industry and academia.
- Finally, we demonstrate the viability of integrating LM-Agents into FL with a proof-of-concept[1].

The remainder of this paper is organized as follows: Section 2 provides the context for FL and LM-Agents; Section 3 introduces the Agentic-FL paradigm and explains how agents can be integrated into FL; Section 4 demonstrates the main challenges of integrating these two strategies; Section 6 concludes this work.

2 BACKGROUND

2.1 FEDERATED LEARNING

FL is a distributed and collaborative paradigm for model training that enables multiple clients to contribute their computational resources and local data toward building a shared, global model McMahan et al. (2017). In this setting, participants receive an initial model whose structure depends on the specific application and environment, and train it locally using their private datasets. After local training, each client sends its updated model parameters to a central server, which aggregates the client updates, typically using a weighted average proportional to the local dataset sizes, and broadcasts the updated global model back to the clients. This process is repeated over multiple communication rounds until global model convergence.

This paradigm offers several advantages, including enhanced data privacy and compliance with data protection regulations, as raw data never leaves the clients. Depending on the application, these clients can be personal smartphones, enterprises, hospitals, or financial institutions holding sensitive information. Moreover, FL leverages distributed computational resources, parallelizing the training process and reducing dependency on a single centralized node. Its inherently distributed nature also provides access to more diverse and representative datasets, allowing better model generalization and robustness.

However, implementing FL in real-world scenarios poses several challenges. One major issue is system heterogeneity, since edge devices often have limited computational power and communication bandwidth, constraining local training and model transmission. Another critical challenge is data heterogeneity: different clients typically collect data from distinct, non-identically distributed

[1]The implementation is available at `https://github.com/rafaeloj/k-agent`

sources Hamedi et al. (2025). Furthermore, as a distributed computing environment, FL must deal with dynamic participation: clients may join or leave the federation intermittently, become temporarily unavailable, or experience variations in computational resources and data distribution shifts. These factors significantly complicate the stability and convergence of federated training Xiang et al. (2024a); Crawshaw & Liu (2024a).

Numerous approaches have been proposed to mitigate these issues, including client selection strategies, improved aggregation algorithms, personalization methods, and communication-efficient techniques Lai et al. (2021); Tan et al. (2022); Salehi et al. (2024); Wang et al. (2020); Karimireddy et al. (2020). Nonetheless, flexible solutions that generalize across diverse operational conditions and adaptive mechanisms capable of handling environmental changes such as client availability, resource variation, and data distribution shifts remain an open research challenge Hamedi et al. (2025). In this context, this work introduces a novel approach that leverages LM-Agents to enhance flexibility and adaptivity in federated learning, enabling dynamic selection and coordination of different strategies according to the current state of the federation.

2.2 LLM-BASED AGENTS

In recent years, LM-Agents have emerged as powerful autonomous systems for adaptive decision-making and coordination across complex environments Wang et al. (2024). Their ability to interpret context, plan actions, and communicate through natural language makes them highly suitable for distributed and dynamic systems. Within the context of FL, agents offer a natural abstraction for both clients and the central server in tasks of coordination and orchestration, enabling analysis, reasoning, self-coordination, and adaptive responses to changing conditions such as varying client availability, resource constraints, and data shifts Wang et al. (2024).

LM-Agents typically comprise four main components Acharya et al. (2025). The profile defines the agent's objectives and behaviors, constraints and limitations, and available skills, shaping its overall behavior and decision-making goals. The memory maintains contextual knowledge in both recent interactions (short-term memory) and accumulated experience (long-term memory), enabling consistent and improved decision-making over time. The planning component is responsible for decomposing goals into executable steps, reasoning about strategies, and selecting actions aligned with the current environment state. Finally, the action component executes these decisions by interacting with the environment, other agents, or external tools. Together, these elements are enhanced by LLMs, which analyze textual descriptions and environment data and generate the planning and action outputs required for agent execution.

Beyond these core components, several complementary techniques have been proposed to enhance the reasoning and operational capabilities of LM-Agents. Tool usage allows agents to access external functions, APIs, or computational resources, extending their abilities beyond the language model itself Schick et al. (2023). Frameworks such as ReAct Yao et al. (2023b) (Reason + Act) and Chain-of-Thought (CoT) reasoning Wei et al. (2022a) enable structured problem-solving, combining step-by-step deliberation with interactive action execution. These mechanisms allow agents not only to reason abstractly but also to act in real time, query data, monitor performance, and refine their strategies based on feedback, thereby increasing autonomy and robustness in dynamic environments. Furthermore, more sophisticated solutions were developed to enhance these capabilities, such as Tree-of-Thought (ToT), Proof-of-Thought (PoT), and Graph-of-Thought (GoT) Yao et al. (2023a); Ganguly et al. (2024); Besta et al. (2024). Furthermore, LLM agents enable the use of reinforcement learning for online learning; this capability, coupled with the ability to solve complex problems in LLMs, makes LLM agents powerful in dynamic environments Peiyuan et al. (2024).

In the context of FL, these characteristics make LM-Agents particularly advantageous. Their inherent reasoning and memory mechanisms allow them to model and adapt to system heterogeneity, dynamically select or prioritize clients, personalize training strategies, and adjust aggregation methods based on the current federation state. Furthermore, the ability to use external tools enables real-time communication monitoring and performance analysis, fostering self-adaptive orchestration. Consequently, LM-Agents introduce a promising paradigm for constructing flexible, context-aware, and adaptive FL systems capable of evolving with their environments and optimizing collaboration among distributed participants.

3 AGENT SYSTEMS FOR FEDERATED LEARNING

FL is dynamic and multivariate in nature, making its training unstable and challenging. Therefore, researchers study separately how each component of the federated system impacts training. In this section, we explain the key challenges of FL, considering both the server side and the client side, and how an agentic approach can be used to address these problems. We also describe how the literature tackles these challenges, explore the research opportunities that arise, and discuss how Large Language Model (LLM) Agents can offer a promising solution.

3.1 SERVER-SIDE AGENTS

In FL, systemic problems such as *i) synchronization*, *ii) communication overhead*, and *iii) fault tolerance* make its implementation challenging. Furthermore, other factors such as *iv) mitigation of bias* and *v) security* are also important for maintaining a fair and secure system for users. Therefore, much research is being conducted to mitigate these impacts on FL.

Synchronization. The standard FL protocol relies on synchronous communication, often being limited by slower clients (stragglers). While asynchronous FL attempts to mitigate these delays via partial aggregation with time limits, it introduces the challenge of model staleness Wu et al. (2021). Specifically, updates from resource-limited devices may arrive late, and merging these stale weights degrades the overall model performance compared to the progress already achieved by faster clients.

In this scenario, LM-Agents contribute through their adaptation to dynamic environments. Long-term memory allows contextualization of connectivity patterns: for example, by identifying that a client exhibits high latency only at specific times (e.g., at night), the agent's reasoning component can set a dynamic timeout adjusted for that round, instead of prematurely discarding it. Furthermore, global reasoning allows distinguishing isolated failures from systemic problems, such as the simultaneous unavailability of devices in the same geographic region, enabling more robust recovery strategies than simple fixed rules.

Another aspect of FL investigates Partial Training Techniques that mitigate latency by adjusting the model size to the local hardware Pfeiffer et al. (2023), but make aggregation complex: the weighted average fails when faced with heterogeneous parameters, leaving parts of the architecture underrepresented Pfeiffer et al. (2023). In addition, the construction of submodels via "neuron importance" creates a systemic bias in which clients capable of training a complete architecture dominate learning, hindering the generalization capacity of the global model Wen et al. (2023).

Agent-based methods offer a dynamic solution to this aggregation dilemma. Unlike static algorithms, a server agent can use its long-term memory to track the update frequency of each parameter or layer across runs. Upon detecting that certain sections of the model are underrepresented (due to the prevalence of weak devices), the agent's reasoning mechanism can intervene, assigning higher weights to rare updates or planning future selection to prioritize clients capable of training those neglected sections. This adaptive orchestration mitigates hardware bias and ensures the integrity of the overall model.

Communication overhead. Although many studies aim to reduce the communication overhead caused by FL, this remains an open challenge Daly et al. (2024). The distributed nature of the data requires multiple rounds of communication to achieve convergence during training. Consequently, sending updates over the network incurs a large payload, as these updates are directly linked to the size of the models, including large-scale architectures such as fundational models Daly et al. (2024).

One way to mitigate the impact of communication overhead is through client selection. Traditionally, this is used to reduce communication bottlenecks by decreasing the number of devices sending models in each round. Essentially, client selection aims to identify the best devices within the federated system, given a specific criterion (e.g., loss, amount of data, channel size), to boost training and reduce the number of communication rounds McMahan et al. (2017); Jee Cho et al. (2022). However, selection algorithms that require hyperparameter optimization are vulnerable to misconfiguration problems. On the other hand, dynamic selection solutions Lai et al. (2021); de Souza et al. (2024); Zheng et al. (2024), while more efficient in federated environments, generally consider only a few selection criteria or are computationally expensive. Therefore, adaptive mechanisms

that adjust to the system's needs and utilize multivariate criteria within the federated system are foundational to its evolution.

In this way, agent-based solutions are promising because they introduce a layer of meta-reasoning over client selection. By cross-referencing historical performance (long-term memory) with the current state of the network, the agent can infer which selection strategy maximizes utility at that specific moment, overcoming the rigidity of static algorithms. For example, when detecting a scenario of high hardware homogeneity but heterogeneous data, the agent can dynamically opt for Power-of-choice Jee Cho et al. (2022); while in situations where the balance between efficiency and loss is critical, it can transition to utility-based strategies such as Oort Lai et al. (2021). More than just selecting devices, the reasoning component allows the agent to adjust the cardinality of the selection in real time, avoiding communication bottlenecks that would go unnoticed by fixed heuristics.

In parallel, the agent can act on payload mitigation through intelligent management of compression techniques. Instead of applying uniform quantization, which can degrade the performance of capable clients, an orchestrating agent can negotiate the precision of the model individually. Using inference tools, the agent decides on sparcification or aggressive quantization only on clients with severe battery or bandwidth constraints, maintaining the integrity of the entire model for robust devices. This granularity of decision, unfeasible for traditional optimizers due to the explosion of the search space, becomes tractable through the generalization capabilities of agents.

Fault tolerance. In distributed systems, it is common for devices to fail, either due to connectivity issues or physical reasons such as low battery levels Crawshaw & Liu (2024b); Xiang et al. (2024b). Thus, when selected by the server, they may be unable to perform the task or return the result, generating synchronization problems as well as delays in training. Furthermore, the server itself may fail, leading to loss of state and requiring recovery and state synchronization Xu et al. (2023a). In this context, LM-Agents can be introduced as a complementary mechanism: when devices return online, they can send information about the reasons for their failure, which the agent can use to make informed decisions to prevent those devices from becoming underrepresented during training. At the system level, agents may also support re-establishing connections, handling error logs, and triggering recovery procedures.

Bias. We can divide the sources of bias into three categories Benmalek & Seddiki (2024); Benarba & Bouchenak (2025): *i) Demographic bias:* In these cases, the imbalance of information generates an underrepresentation that is propagated to the global model if not properly addressed Criado et al. (2022); Zhu et al. (2021). Within this context, the data used to train the models is personal; therefore, creating a single generic model for all devices may not be the best choice Tan et al. (2023). *ii) Systemic bias:* This is directly linked to devices with greater computational power or better connectivity, which allows them to participate in more training rounds Benarba & Bouchenak (2025). Consequently, the global model tends to represent these devices more accurately than those with limited capabilities. *iii) Algorithmic bias:* Many studies that aim to solve other challenges either ignore the bias they themselves introduce or explicitly rely on biased criteria. For example, some selection mechanisms use metrics such as loss to choose which device will be trained in the next round Jee Cho et al. (2022); Lai et al. (2021).

To address these problems, many strategies adopt an approach that produces multiple models for different groups or clients Talasso et al. (2024); Tan et al. (2022; 2023). While they achieve excellent results through collaboration and personalization, challenges such as balancing generalization and personalization, avoiding negative knowledge transfer, and managing algorithmic complexity remain open Yang et al. (2024); M. Ghari & Shen (2024). Thus, agents bring transparency and explainability to training decisions. To mitigate systemic bias, the agent can dynamically apply quantization or pruning techniques, enabling the participation of clients with limited resources without delaying federation. Furthermore, LM-Agents add a qualitative and semantic analysis to combat algorithmic bias, being able to interpret client metadata (e.g., "sensor located in a rural area") and prioritize it to ensure demographic diversity, a nuance that simple numerical loss metrics fail to capture.

Security. Although FL emerged as a privacy-oriented solution often outweighs its intrinsic security guarantees. Consequently, the focus has shifted to preventing attacks on privacy, such as data inference, and on the integrity of the model Gao et al. (2025). Classic methods like Differential Privacy (DP), Homomorphic Encryption (He), and Robust Aggregation (RA) mitigate these vulnerabilities,

but face rigid trade-offs: DP noise degrades accuracy, he explodes computational cost, and RA can discard legitimate data by mistaking it for attacks Ma et al. (2022); Yurdem et al. (2024).

In this context, agents act as adaptive guardians. Unlike static policies, a security agent can dynamically manage the privacy budget (ϵ), adjusting the level of injected noise based on the sensitivity of the data detected in the current round, balancing protection and utility. Furthermore, in defense against model poisoning, the LM-Agents's contextual reasoning allows it to distinguish between statistical anomalies caused by data heterogeneity (Non-Independently Identically Distributed (Non-IID) clients) and actual malicious attacks, a subtle distinction where algorithms based purely on thresholds often fail.

3.2 Client-side Agents

Devices within a system are responsible for model training; however, each device operates in its own context, performing additional tasks beyond model training and potentially participating in multiple federations Skovajsova et al. (2025). Significant advancements in model optimization, such as quantization and Small Language Model (SML) enable devices to run language models Belcak et al. (2025); Egashira et al. (2024). This possibility allows for intelligent and autonomous task management on the client side. We divide client challenges into three categories: i) task manager, ii) security, and iii) training.

Task Manager. Edge devices execute multiple simultaneous tasks in a resource-constrained environment. While traditional schedulers work for static routines, they fail to handle the stochastic CPU and memory volatility typical of resource-constrained Internet of Things (iot) scenarios Akubathini et al. (2021). As the complexity of tasks increases, mechanisms capable of real-time adaptation become necessary Friha et al. (2024). Unlike fixed priority queues, a management agent can reason about the device's usage history to predict availability windows and orchestrate training execution, minimizing conflicts with the user experience and optimizing energy consumption.

Security. In classic FL architectures, the server is assumed to be trustworthy, a premise frequently violated in real-world scenarios. While DP mitigates risks, the application of static noise degrades the model's utility. Here, a guardian agent overcomes fixed mechanisms by dynamically managing the client's privacy budget. The agent can assess the sensitivity of the data in the current sample and inject noise proportionally, or detect gradient inversion attempts by monitoring the behavior of server requests, blocking participation if the pattern deviates from normality. This active and contextual defense is unfeasible for passive cryptographic protocols.

Training. Local training is the core of federation, but suffers from the double pressure of data and resource heterogeneity. Static strategies fail when attempting to optimize both simultaneously Panchal et al. (2023); Kang et al. (2024a); Liu et al. (2020). A training agent can act as a local orchestrator, capable of aligning hyperparameters not only to the architecture but also to the instantaneous state of the device. For example, upon detecting a highly skewed data distribution, the agent can apply specific regularization techniques or autonomous oversampling. Simultaneously, if resources become scarce, it unilaterally decides to apply aggressive quantization or sparcification, ensuring that the contribution is sent, even if degraded, instead of aborting the process, maximizing federation resilience.

However, while SMLs enable this local orchestration, the computational overhead and token generation costs remain critical bottlenecks for battery life and inference latency on edge devices. Therefore, developing efficient prompt strategies and managing token costs are essential research proposals for the future of LM-Agents.

4 Challenge and Future Directions

LM-Agents have proven to be a promising methodology for mitigating FL problems, but their introduction also raises issues that require further study. Therefore, in this subsection, we discuss the challenges that emerge in this area as well as the research opportunities that follow.

Figure 1 illustrates how an agentic architecture can be built within a distributed environment. Inside the server, a multi-agent system is responsible for managing the distributed environment, while on the client side, a LM-Agents manages local tasks. The server-side agent acts based on client states, activity histories, and previous decisions, determining which tasks should be executed next

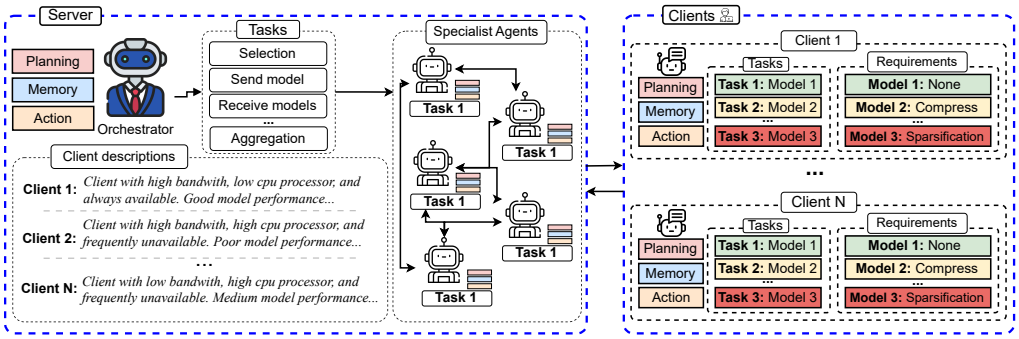

Figure 1: AgenticFL Paradigm: shows how an agent or a multi-agent system can be introduced into the FL pipeline.

and identifying their dependencies. These tasks are then forwarded to specialist agents responsible for solving each task created by the orchestrator, and these specialists may also exchange information with one another. This approach helps to modularize and isolate each stage of the distributed pipeline. Thus, we are not limited to selection and aggregation but can also customize models to meet device-specific needs. For example, if the system detects clients with limited computational resources, the orchestrator agent can update their descriptions upon observing slow responses and create a quantization task before sending the model. Similarly, if a client is underperforming, the system can request additional training time. These agents can even maintain their own communication channels with the server, informing it of the requirements associated with each task or model. In this way, the client can define its level of contribution and the conditions under which it will train the model in coordination with the orchestrating agent. However, there are a number of problems in linking these two areas, such as:

Reliability and Hallucination. Despite their sophistication, LM-Agents are still susceptible to hallucinations Yang et al. (2025). In the context of FL, these flaws can lead the agent to misinterpret the utility of a client, resulting in the repeated selection of ineffective devices or the exclusion of significant contributors. From the client's perspective, agents orchestrating local training may hallucinate inappropriate hyperparameters; when aggregated, these updates can cause unintentional model poisoning, degrading the global model and hindering federation convergence. Therefore, the use of safety mechanisms, such as guardrails or structured outputs, to validate values and impose strict operational constraints becomes imperative.

Agent Components. Agent systems can be constructed in several ways, and their architectures are directly shaped by the problems they aim to solve. This applies equally to FL; thus, it is necessary to investigate how each agent component can contribute to the federated learning process, from both the server and client perspectives.

Within FL, actions performed in the past can be very useful for understanding the current state of the federation and for determining what needs to be done to achieve the expected result. Thus, the memory component is fundamental for building an agent in this context. However, there are different ways to represent information to the agent, such as using text, embeddings stored in databases, or structured lists. Finding the best representation for each step and each piece of information within the distributed system is challenging, as is defining how to display client metadata to the agent, namely, how to efficiently leverage the context windows of language models without sending unnecessary information. Furthermore, the large amount of information generated must be stored in a way that enables efficient and practical retrieval, making the study of storage methods and retrieval mechanisms essential.

Within a distributed system, the order and manner in which tasks are performed are important for conserving resources and efficiently utilizing system components. Therefore, planning task sequences based on both past actions and the current state, while aiming for short- and long-term gains, is also challenging. Understanding how to enable the agent to plan its actions using CoT, ToT, and similar techniques to ensure reliability and robustness in its decisions will make its behavior more effective. Furthermore, the planning and reasoning process generally generates a large number of

tokens, linked to the information produced by the federated system; therefore, it is important to identify mechanisms to mitigate this impact.

Security. Beyond the problems inherent to FL, LM-Agents introduce additional vulnerabilities that require investigation. Miao Yu et al. summarize these issues as follows: the Language Model (LM) itself is vulnerable to **jailbreaking**, that is, methods used to bypass security mechanisms; **prompt injection**, which attempts to alter the agent's behavior by injecting malicious prompts into the LLM; and **backdoors**, which insert triggers during training to produce attacker-specified outputs. In addition, it is necessary to prevent clients from poisoning the model's memory with false information through **memory poisoning**, or from obtaining improper information that leads to **privacy leakage** Yu et al. (2025). Techniques capable of handling attempts at manipulation or abuse of tools used by the agent are fundamental to preventing overload and serious impacts on the federated system. In scenarios where many clients deploy agents, risks such as cooperative agent attacks or infection attacks that aim to corrupt agent behavior naturally arise, and existing studies on collaborative or topological defense strategies may be leveraged to address these threats.

Federated pipeline. Today, there are different FL paradigms, such as asynchronous, hierarchical, horizontal, and vertical federation Kairouz et al. (2021b). With the advancement of agents, we can envision a framework in which agents distributed across clients can request federation when necessary. In this way, a client-side agent could, through internal mechanisms, determine when its model is no longer adequate or is becoming outdated and subsequently request a new model from the server, which would initiate a federation round after receiving a sufficient number of such requests. This new pipeline introduces adaptability to the models and flexibility to the training process, enabling federation to occur only among clients that require it.

Exploring new metrics with Code Agents. Traditional metrics such as accuracy and loss are often insufficient for comprehensive decision-making, and assessing client utility more holistically remains a challenge. Code Agents can propose and implement solutions that evaluate new metrics and combine information to construct previously undefined ones. For example, they can identify a group of disadvantaged clients and generate a minority fairness metric, enabling the system to optimize its actions to benefit this group in subsequent rounds.

Scalability and Contextual Limits Scaling the federation to thousands of clients inevitably hits the physical limits of LM context windows, demanding architectural innovations to reduce information overload. To address this, we envision three complementary directions. First, Retrieval-Augmented Generation (RAG) can dynamically fetch only relevant client metadata, creating similarity clusters rather than processing the entire network at once. Second, hierarchical multi-agent architectures can divide the workload, where local sub-agents analyze specific client subgroups and forward only condensed, high-value candidates to a central orchestrator. Finally, integrating deterministic filtering tools can pre-categorize clients based on heuristics, allowing the agent to focus its reasoning solely on a pre-selected, manageable pool of candidates.

Strategic Modeling and Mechanism Design: In a decentralized ecosystem, clients are naturally driven by local constraints and objectives, which may lead them to underreport their available resources to save battery, or overstate their data quality to bias the global model toward their specific needs. To design robust defenses against selfish or malicious clients, Agentic-FL can be formalized through Mechanism Design. Each client i has a truly private type, denoted by $\theta_i = (b_i, c_i, q_i, bw_i)$, where b_i is the battery level, c_i the computational cost, q_i the data quality, and bw_i the bandwidth. The client reports a type $\hat{\theta}_i$ to the Orchestrator Agent, and can report incorrect values ($\hat{\theta}_i \neq \theta_i$) to gain advantages. The server acts as the mechanism designer, defining an allocation rule $w_i(\hat{\theta}_i)$ (the weight or probability of selection in the round) and a penalty or reputation function Π. The utility of the strategic client is given by:

$$U_i(\hat{\theta}_i|\theta_i) = v_i \cdot w_i(\hat{\theta}_i) - C(\theta_i) - \Pi(\text{audit}(\hat{\theta}_i)) \tag{1}$$

where v_i is the value the client assigns to influence in the global model and $C(\theta_i)$ is its actual training cost. While classical Mechanism Design relies on rigid numerical inputs to solve this equation, Agentic-FL introduces LM-Agents to operationalize these dynamics in unstructured environments. Here, the Orchestrating Agent uses contextual reasoning to dynamically infer the reported types $\hat{\theta}_i$ from complex metadata and behavioral logs. The open challenge for the community is the transition from static mathematical formulations to LM-Agents driven mechanisms, where agents au-

tonomously negotiate incentives, audit contributions, and adjust the value of Π in real time based on semantic analysis, rather than just numerical limits.

The integration of LM-Agents into FL transcends mitigating isolated challenges; it signals a paradigm shift from centralized and static orchestration to autonomous federated ecosystems. While traditional FL treats clients as passive "workers" our long-term vision projects a scenario where agents act as economic brokers in a decentralized knowledge marketplace. In this future, the training pipeline is no longer triggered by a central server but negotiated peer-to-peer: clients initiate federations based on local needs (e.g., "I need to update my night vision model") and negotiate participation based on new metrics of utility and fairness, not just accuracy. This evolution transforms FL from a distributed optimization process into a complex mechanism for incentive negotiation and on-demand collaboration, opening frontiers for the intersection between Game Theory and Machine Learning.

5 PROOF-OF-CONCEPT

This section illustrates the potential of use llm agents to define the number of K clients to select. First, we describe the setup and methods to construct the K-Agent. Second ,we evaluate the results of experiments.

Setup. The simulation of FL was implemented using the Flower framework, utilizing the CIFAR-10 and MNIST datasets under a Dirichlet distribution ($\alpha = 0.1$) to create a severely Non-IID scenario with 25 clients and 50 rounds of communication. The proposed method, K-Agent, was developed via LangGraph and Ollama, testing three models: Qwen3 8b, Llama3.2 3b, and Llama3.1 8b. This involved training a CNN optimized by SGD and comparing it to established baselines such as Random Selection, Round Robin, Pow-of-Choice, and Oort. Effectiveness was validated by averaging three runs based on two fundamental metrics: Distributed Accuracy, to measure overall performance; Selection Time (ST), which adds the LLM inference to the selection algorithm, which accounts for the network overhead accumulated by client selection using three techniques: CoT, Few-Shot, and Description Only (DO).

Results. Table 1 table presents the accuracy, selection time (ST), and average K resulting from different combinations of models and prompts for MNIST and CIFAR10. It can be observed that the *Qwen3 8b* model tended to choose lower K values, but with a higher standard deviation. This variability allowed for higher accuracies, but with higher ST. While for MNIST we can see that *Llama3.2 3b* achieved results similar to the other models with a lower ST due to shorter and more objective thought chains. However, despite this, in some cases the agent gets lost in the reasoning, requiring other calls and increasing its time, as demonstrated in the Oort using CoT.[2]

Regarding prompt techniques, CoT showed a predominantly lower TS. Some exceptions occurred with *Llama3.1 8b*, where a simple description was faster; however, this configuration resulted in static K values for PoC and Random, reducing the agent's adaptive effectiveness.

Combined with sophisticated techniques such as ToT, long-term and short-term memory, and even advanced retrieval techniques and tools, agents represent promising approaches for the dynamic management of federated systems.

6 CONCLUSION

In this work, we introduce Agentic-FL, a paradigm shift that evolves FL from a static optimization process to an autonomous trading ecosystem. We argue that the inherent complexity of distributed scenarios, characterized by non-IID data heterogeneity, resource volatility, and security risks, demands more than fixed algorithms; it demands cognitive adaptation.

Our analysis demonstrated that the integration of LLM-based Agents offers a robust solution on two fronts: on the Server, agents act as orchestrators capable of combating systemic bias through contextual aggregation and dynamic selection; on the Client, agents operate as "local gatekeepers" managing privacy budgets and adapting training in real time to ensure the inclusion of resource-limited devices.

[2]CoT benefits models large enough to generalize reasoning lines *step-by-step* as evidenced by Wei et al. Wei et al. (2022b).

Table 1: Impact of LLM and Prompt on Selection and Performance

Cifar 10										
Models	**Prompt**	**PoC**			**Random**			**Oort**		
		K avg	**Acc.** (%)	**ST** (s)	**K avg**	**Acc.** (%)	**ST** (s)	**K avg**	**Acc.** (%)	**ST** (s)
Qwen3	**DO**	7±5	38±13	3,79	6±5	**39±11**	4,71	12±6	**39±14**	4,45
	Few-Shot	6±4	38 ±13	1,43	5±4	38±15	2,28	9±5	38±15	2,96
	CoT	7±5	**39±13**	1,79	5±4	37±14	1,65	9 ±4	39 ±13	2,53
Llama3.1	**DO**	10±0	37±13	**0,18**	10±0	38±10	0,38	9±2	38±13	**0,19**
	Few-Shot	8±6	37±13	3,21	10±7	38±12	1,88	8±4	38+15	3,92
	CoT	8±2	38 ±12	0,53	8±2	37±14	0,51	8 ±1	39 ±12	0,57
Llama3.2	**DO**	10±0	36±13	0,29	9±1	36±14	0,21	9±1	39±11	0,31
	Few-Shot	5±0	35±13	0,31	10±7	36±11	1,88	9±1	38±14	0,27
	CoT	9±1	35+15	0,20	8±2	36±13	**0,15**	8±1	38±13	0,22
MNIST										
Models	**Prompt**	**PoC**			**Random**			**Oort**		
		K avg	**Acc.** (%)	**ST** (s)	**K avg**	**Acc.** (%)	**ST** (s)	**K avg**	**Acc.** (%)	**ST** (s)
Qwen3	**DO**	7±6	95,97±3	4,00	6±5	**96,90±1**	8,65	11±6	**97,06±1**	4,45
	Few-Shot	7±5	96,68±2	1,67	5±4	**96,32±2**	1,74	8±6	**97,22±1**	1,97
	CoT	7±5	**96,50±2**	1,63	8±2	95,25±2	0,18	9±6	**97,05±1**	2,21
Llama3.1	**DO**	10±0	**96,04±3**	**0,17**	10±2	**96,66±2**	0,18	9±2	**97,05±1**	0,19
	Few-Shot	10±1	95,76±2	0,16	6±1	95,86±3	0,34	9±1	**97,21±1**	0,33
	CoT	8±2	**96,64±2**	0,26	8±2	95,89±2	0,21	5±1	**97,09±1**	0,28
Llama3.2	**DO**	10±0	95±3	0,31	9±1	**96,64±1**	0,20	9±1	**97,22±1**	0,29
	Few-Shot	9±2	**96,26±2**	**0,16**	7±2	**96,13±2**	**0,14**	8±1	**97,00±1**	**0,16**
	CoT	9±1	**96,13±2**	0,19	8±2	**96,25±2**	0,18	5±1	**96,77±1**	2,21

By mitigating barriers to entry for heterogeneous clients and increasing resilience against attacks, Agentic-FL not only improves training efficiency but also promotes a fairer and more representative system. Looking to the future, we envision these agents as the forerunners of decentralized market models, where collaboration ceases to be merely a computational protocol and becomes an intelligent, self-managed economic exchange.

ACKNOWLEDGMENTS

This project was supported by the brazilian Ministry of Science, Technology and Innovations, with resources from Law nº 8,248, of October 23, 1991, within the scope of PPI-SOFTEX, coordinated by Softex and published Arquitetura Cognitiva (Phase 3), DOU 01245.003479/2024-10 and was partially sponsored by CNPq grant 407192/2025-5.

REFERENCES

Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B. Divya. Agentic ai: Autonomous intelligence for complex goals—a comprehensive survey. *IEEE Access*, 13:18912–18936, 2025. doi: 10.1109/ACCESS.2025.3532853.

Premkumar Akubathini, Sameer Chouksey, and Hariram Selvamurugan Satheesh. Evaluation of machine learning approaches for resource constrained iiot devices. In *2021 13th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 74–79, 2021. doi: 10.1109/ICITEE53064.2021.9611880.

Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. Small language models are the future of agentic ai, 2025. URL https://arxiv.org/abs/2506.02153.

Nawel Benarba and Sara Bouchenak. Bias in federated learning: A comprehensive survey. *ACM Comput. Surv.*, 57(11), June 2025. ISSN 0360-0300. doi: 10.1145/3735125. URL https://doi.org/10.1145/3735125.

Mourad Benmalek and Abdessamed Seddiki. Bias in Federated Learning: Factors, Effects, Mitigations, and Open Issues. *Revue des Sciences et Technologies de l'Information - Série ISI : Ingénierie des Systèmes d'Information*, 29(6):2137–2160, Dec 2024. doi: 10.18280/isi.290605. URL https://hal.science/hal-04855447.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michał Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: solving elaborate problems with large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i16.29720. URL https://doi.org/10.1609/aaai.v38i16.29720.

Michael Crawshaw and Mingrui Liu. Federated learning under periodic client participation and heterogeneous data: a new communication-efficient algorithm and analysis. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024a. Curran Associates Inc. ISBN 9798331314385.

Michael Crawshaw and Mingrui Liu. Federated learning under periodic client participation and heterogeneous data: A new communication-efficient algorithm and analysis. *Advances in Neural Information Processing Systems*, 37:8240–8299, 2024b.

Marcos F. Criado, Fernando E. Casado, Roberto Iglesias, Carlos V. Regueiro, and Senén Barro. Non-iid data and continual learning processes in federated learning: A long road ahead. *Information Fusion*, 88:263–280, 2022. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2022.07.024. URL https://www.sciencedirect.com/science/article/pii/S1566253522000884.

Katharine Daly, Hubert Eichner, Peter Kairouz, H. Brendan McMahan, Daniel Ramage, and Zheng Xu. Federated learning in practice: Reflections and projections. In *2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*, pp. 148–156, 2024. doi: 10.1109/TPS-ISA62245.2024.00026.

Allan M. de Souza, Filipe Maciel, Joahannes B.D. da Costa, Luiz F. Bittencourt, Eduardo Cerqueira, Antonio A.F. Loureiro, and Leandro A. Villas. Adaptive client selection with personalization for communication efficient federated learning. *Ad Hoc Networks*, 157:103462, 2024. ISSN 1570-8705. doi: https://doi.org/10.1016/j.adhoc.2024.103462. URL https://www.sciencedirect.com/science/article/pii/S1570870524000738.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature machine intelligence*, 5(3):220–235, 2023.

Kazuki Egashira, Mark Vero, Robin Staab, Jingxuan He, and Martin Vechev. Exploiting llm quantization. *Advances in Neural Information Processing Systems*, 37:41709–41732, 2024.

Othmane Friha, Mohamed Amine Ferrag, Burak Kantarci, Burak Cakmak, Arda Ozgun, and Nassira Ghoualmi-Zine. Llm-based edge intelligence: A comprehensive survey on architectures, applications, security and trustworthiness. *IEEE Open Journal of the Communications Society*, 2024.

Lei Fu, Huanle Zhang, Ge Gao, Mi Zhang, and Xin Liu. Client selection in federated learning: Principles, challenges, and opportunities. *IEEE Internet of Things Journal*, 10(24):21811–21819, 2023.

Debargha Ganguly, Srinivasan Iyengar, Vipin Chaudhary, and Shivkumar Kalyanaraman. PROOF OF THOUGHT : Neurosymbolic program synthesis allows robust and interpretable reasoning. In *The First Workshop on System-2 Reasoning at Scale, NeurIPS'24*, 2024. URL https://openreview.net/forum?id=Pxx3r14j3U.

Ying Gao, Yuxin Xie, Huanghao Deng, and Zukun Zhu. Gradient inversion attack in federated learning: Exposing text data through discrete optimization. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 2582–2591, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.176/.

Parisa Hamedi, Roozbeh Razavi-Far, and Ehsan Hallaji. Federated continual learning: Concepts, challenges, and solutions. *Neurocomputing*, 651:130844, 2025. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2025.130844. URL https://www.sciencedirect.com/science/article/pii/S0925231225015164.

Rafael O. Jarczewski, Eduardo Cerqueira, Luiz F. Bittencourt, Antonio A. F. Loureiro, Leandro A. Villas, and Allan M. de Souza. Let's federate - effective communication strategy for dynamic client participation. In *2024 International Conference on Machine Learning and Applications (ICMLA)*, pp. 361–368, 2024. doi: 10.1109/ICMLA61862.2024.00055.

Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Towards understanding biased client selection in federated learning. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera (eds.), *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 10351–10375. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/jee-cho22a.html.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021a.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14(1–2):1–210, June 2021b. ISSN 1935-8237. doi: 10.1561/2200000083. URL https://doi.org/10.1561/2200000083.

Myeongkyun Kang, Soopil Kim, Kyong Hwan Jin, Ehsan Adeli, Kilian M. Pohl, and Sang Hyun Park. Fednn: Federated learning on concept drift data using weight and adaptive group normalizations. *Pattern Recognition*, 149:110230, 2024a. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2023.110230. URL https://www.sciencedirect.com/science/article/pii/S0031320323009275.

Myeongkyun Kang, Soopil Kim, Kyong Hwan Jin, Ehsan Adeli, Kilian M Pohl, and Sang Hyun Park. Fednn: Federated learning on concept drift data using weight and adaptive group normalizations. *Pattern Recognition*, 149:110230, 2024b.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/karimireddy20a.html.

Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. Oort: Efficient federated learning via guided participant selection. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pp. 19–35. USENIX Association, July 2021.

ISBN 978-1-939133-22-9. URL https://www.usenix.org/conference/osdi21/presentation/lai.

Haoyuan Li, Jindong Wang, Mathias Funk, and Aaqib Saeed. Position: Agentic federated learning for ai-driven strategy design and optimization. In *ICML 2025 Workshop on Collaborative and Federated Agentic Workflows*, 2025.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International conference on machine learning*, pp. 6357–6368. PMLR, 2021.

Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems*, 31(8):1754–1766, 2020. doi: 10.1109/TPDS.2020.2975189.

Pouya M. Ghari and Yanning Shen. Personalized federated learning with mixture of models for adaptive prediction and model fine-tuning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 92155–92183. Curran Associates, Inc., 2024. doi: 10.52202/079017-2926. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/a7a6465b9344c5ecc691c02af40661a7-Paper-Conference.pdf.

Jing Ma, Si-Ahmed Naas, Stephan Sigg, and Xixiang Lyu. Privacy-preserving federated learning based on multi-key homomorphic encryption. *Int. J. Intell. Syst.*, 37(9):5880–5901, July 2022. ISSN 0884-8173. doi: 10.1002/int.22818. URL https://doi.org/10.1002/int.22818.

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*, pp. 1273–1282. PMLR, 2017.

Dung Thuy Nguyen, Taylor T. Johnson, and Kevin Leach. Fisc: Federated domain generalization via interpolative style transfer and contrastive learning. *CoRR*, abs/2410.22622, 2024. URL https://doi.org/10.48550/arXiv.2410.22622.

Kunjal Panchal, Sunav Choudhary, Subrata Mitra, Koyel Mukherjee, Somdeb Sarkhel, Saayan Mitra, and Hui Guan. Flash: Concept drift adaptation in federated learning. 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. 26931–26962. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/panchal23a.html.

Feng Peiyuan, Yichen He, Guanhua Huang, Yuan Lin, Hanchong Zhang, Yuchen Zhang, and Hang Li. Agile: A novel reinforcement learning framework of llm agents. *Advances in Neural Information Processing Systems*, 37:5244–5284, 2024.

Kilian Pfeiffer, Martin Rapp, Ramin Khalili, and Jörg Henkel. Federated learning for computationally constrained heterogeneous devices: A survey. *ACM Comput. Surv.*, 55(14s), July 2023. ISSN 0360-0300. doi: 10.1145/3596907. URL https://doi.org/10.1145/3596907.

Hongming Piao, Yichen Wu, Dapeng Wu, and Ying Wei. Federated continual learning via prompt-based dual knowledge transfer. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Batool Salehi, Debashri Roy, Jerry Gu, Chris Dick, and Kaushik Chowdhury. Flash-and-prune: Federated learning for automated selection of high-band mmwave sectors using model pruning. *IEEE Transactions on Mobile Computing*, 23(12):11655–11669, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessí, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Lenka Skovajsova, Ladislav Hluchý, and Michal Staňo. A review of multi-objective and multi-task federated learning approaches. In *2025 IEEE 23rd World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pp. 000035–000040, 2025. doi: 10.1109/SAMI63904.2025.10883172.

Gabriel U Talasso, Allan M de Souza, Luiz F Bittencourt, Eduardo Cerqueira, Antonio AF Loureiro, and Leandro A Villas. Fedsccs: hierarchical clustering with multiple models for federated learning. In *ICC 2024-IEEE International Conference on Communications*, pp. 3280–3285. IEEE, 2024.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE transactions on neural networks and learning systems*, 34(12):9587–9603, 2022.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9587–9603, 2023. doi: 10.1109/TNNLS.2022.3160699.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, Mar 2024. ISSN 2095-2236. doi: 10.1007/s11704-024-40231-1. URL https://doi.org/10.1007/s11704-024-40231-1.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022a. Curran Associates Inc. ISBN 9781713871088.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022b.

Jie Wen, Zhixia Zhang, Yang Lan, Zhihua Cui, Jianghui Cai, and Wensheng Zhang. A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 14(2):513–535, Feb 2023. ISSN 1868-808X. doi: 10.1007/s13042-022-01647-y. URL https://doi.org/10.1007/s13042-022-01647-y.

Wentai Wu, Ligang He, Weiwei Lin, Rui Mao, Carsten Maple, and Stephen Jarvis. Safa: A semi-asynchronous protocol for fast federated learning with low overhead. *IEEE Trans. Comput.*, 70 (5):655–668, May 2021. ISSN 0018-9340. doi: 10.1109/TC.2020.2994391. URL https://doi.org/10.1109/TC.2020.2994391.

Ming Xiang, Stratis Ioannidis, Edmund Yeh, Carlee Joe-Wong, and Lili Su. Efficient federated learning against heterogeneous and non-stationary client unavailability. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024a. Curran Associates Inc. ISBN 9798331314385.

Ming Xiang, Stratis Ioannidis, Edmund Yeh, Carlee Joe-Wong, and Lili Su. Efficient federated learning against heterogeneous and non-stationary client unavailability. *Advances in Neural Information Processing Systems*, 37:104281–104328, 2024b.

Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. Asynchronous federated learning on heterogeneous devices: A survey. *Computer Science Review*, 50:100595, 2023a. ISSN 1574-0137. doi: https://doi.org/10.1016/j.cosrev.2023.100595. URL https://www.sciencedirect.com/science/article/pii/S157401372300062X.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148*, 2023b.

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.

Yi Yang, Yitong Ma, Hao Feng, Yiming Cheng, and Zhu Han. Minimizing hallucinations and communication costs: Adversarial debate and voting mechanisms in llm-based multi-agents. *Applied Sciences*, 15(7):3676, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023b.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 6137–6147, 2024.

Miao Yu, Fanci Meng, Xinyun Zhou, Shilong Wang, Junyuan Mao, Linsey Pan, Tianlong Chen, Kun Wang, Xinfeng Li, Yongfeng Zhang, Bo An, and Qingsong Wen. A survey on trustworthy llm agents: Threats and countermeasures. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, pp. 6216–6226, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3736561. URL https://doi.org/10.1145/3711896.3736561.

Betul Yurdem, Murat Kuzlu, Mehmet Kemal Gullu, Ferhat Ozgur Catak, and Maliha Tabassum. Federated learning: Overview, strategies, applications, tools and future directions. *Heliyon*, 10(19):e38137, 2024. ISSN 2405-8440. doi: https://doi.org/10.1016/j.heliyon.2024.e38137. URL https://www.sciencedirect.com/science/article/pii/S2405844024141680.

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

Feng Zheng, Yuze Sun, and Bin Ni. Fedaeb: Deep reinforcement learning based joint client selection and resource allocation strategy for heterogeneous federated learning. *IEEE Transactions on Vehicular Technology*, 73(6):8835–8846, 2024. doi: 10.1109/TVT.2024.3359860.

Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.07.098. URL https://www.sciencedirect.com/science/article/pii/S0925231221013254.

A Appendix

A.1 Semantic analysis of K-Agent results

Figure 2 depicts the overview. K-Agent is implemented on the central server and is triggered at the beginning of each communication round t, consisting of three interconnected modules: Plan, Memory, and Action, as illustrated in Figure 1. The first step of the process (1) consists of defining the selection algorithm that will be used during training. This definition is performed before the start of training, and the same algorithm is used in all rounds. In the next step (2), in each communication round, the agent is triggered to define the ideal value for K, which operates on the client and federation metadata highlighted in step (3). In this part, the tools allow the agent to collect information about the training metadata of each client (e.g., training time, model performance, link quality, etc.)

and the federation as a global average of each attribute. Then, in step (4), the agent inserts the hyperparameter into the defined algorithm, and the system executes the solution. The server then, in step (5), sends the global model to the selected clients, continuing the traditional flow.

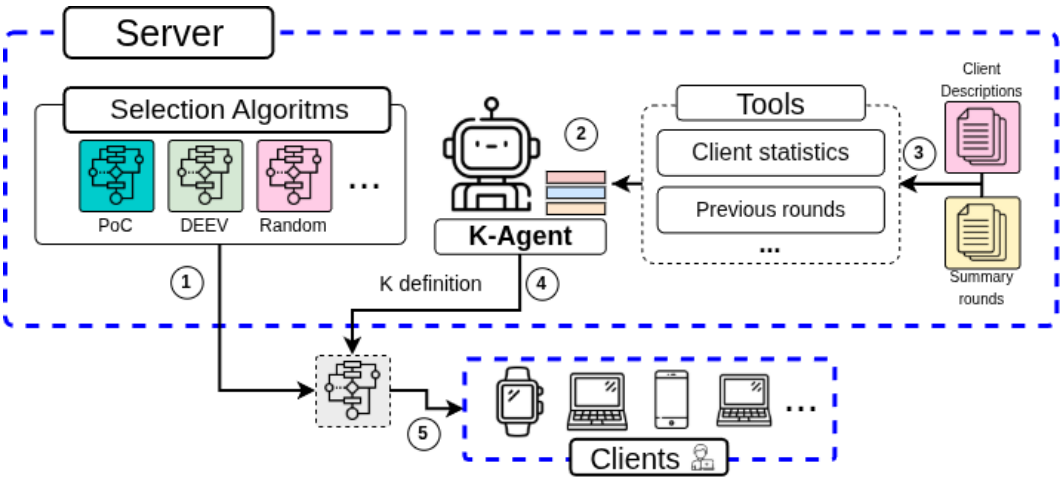

Figure 2: Pipeline of K-Agent

Figure 3 illustrates that the superiority of K-Agent transcends metric performance, being based on its contextual adaptability. By diverging from the fixed strategy in round 25, the agent demonstrates a dynamic perception of the federation's state that static methods lack. When the system detects imminent instability, maintaining the value of $K = 5$ acts as a variance control mechanism, stabilizing the global gradient, before resuming a more aggressive exploration of clients after round 30. This smooth transition between conservatism and exploration is what ensures not only faster convergence but also superior operational robustness in heterogeneous environments.

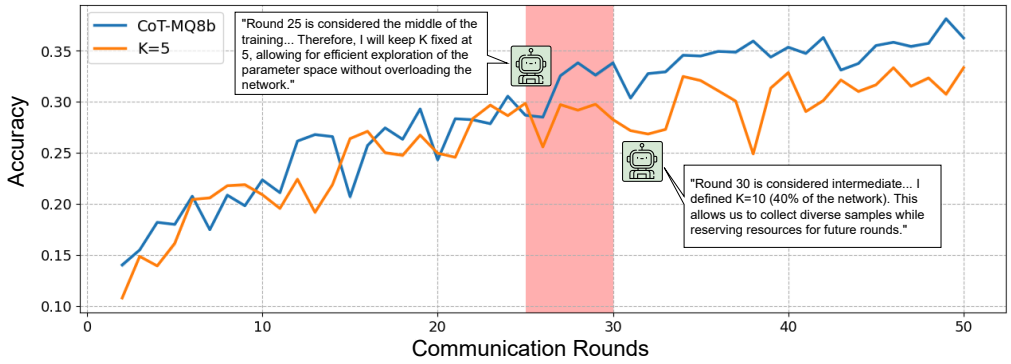

Figure 3: Convergence of PoC: The interval highlights the dynamic adaptation of the agent's strategy.

Additionally, the agent-based architecture introduces a layer of auditability and explainability. Through the analysis of reasoning logs, it is possible to decode the causal logic behind each change in the value of K, transparently. This traceability allows developers to audit the model's heuristics in real time, validating whether the choice for greater customer diversity in advanced rounds was motivated by model stability or by specific performance metrics. Thus, K-Agent establishes a new paradigm where training effectiveness is inseparable from clarity about the system's behavior.

A.2 Compare single LLM vs agent

To illustrate some of the use cases discussed in the previous sections, we designed some experiments with the agentic system. We decided to conduct an experiment comparing the agentic solution

with Federated Average (FedAvg). We compared three different techniques: i) using only LLM with all client descriptions inserted in the model's context window; ii) an agent with ReAct architecture Yao et al. (2023b) with five tools. These two solutions were compared with the classic FedAvg solution in order to understand the potential that agents can achieve.

Setup: The models used were gpt-4o-mini from the OpenAI API. For the experiments, since the objective of this experiment is simply to understand the potential of LLMs, we used a simple MNIST database with a DNN.

Experiment 1: To understand the potential of language models for client selection, we conducted an experiment comparing three strategies: i) FedAvg, i.e., random selection; ii) Selection using an LLM as a selector, for which all client information and the sequence of instructions for selection are input. iii) In addition, we used a simple ReAct architecture that queries client information through three tools (*filter*, *get_stats*, *get_info_by_cid*). The simulation was performed with 10 clients for 25 rounds in a Non-IID scenario. During the experiments, we tested between 5 and 50 clients with dynamic and fixed selection, which will be explained during each experiment.

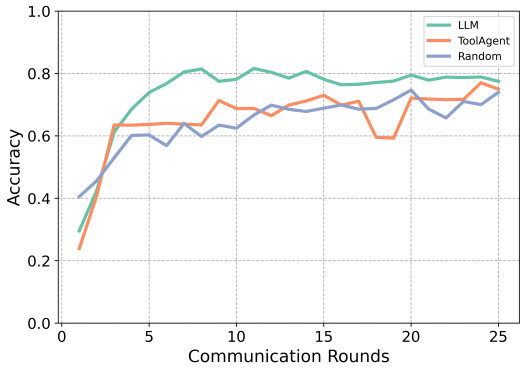

Figure 4: Experiment comparing raw LLM to random selection and ToolAgent.

As we can see in Figure 4, the best-performing method was LLM. This is based on the model having the full context of client information, while the agent must decide how to call the functions to filter clients and then perform the analysis of which ones to select. By analyzing the justifications for both strategies, we realized that they were able to distinguish between training and evaluation performance, in addition to using it to their advantage, selecting clients with poorer performance in the evaluation. LLM predominantly took into account other attributes, such as training time, which the Tool agent did not consider in any round, despite the information being provided. The Tool agent had confusion about the meaning of some attributes. Both considered criteria such as diversity and fairness, but the Tool agent demonstrated greater coherence in its choices, as clients with lower performance were selected to maintain diversity in training.

Table 2: The number of tokens used according to the increase in the number of clients by LLM.

N° Clients	Approachs	Completion Tokens	Prompt Tokens	Total Cost ($)	Total Tokens	Accuracy
5	LLM	178 ± 52	**1336 ± 15**	**0.000308 ± 0.000031**	**1515 ± 54**	0.766264 ± 0.105471
	Tool Agent	**164 ± 122**	6193 ± 1963	0.000804 ± 0.000241	6358 ± 2072	**0.792612 ± 0.101721**
10	LLM	**218 ± 56**	**2259 ± 28**	**0.000470 ± 0.000035**	**2478 ± 64**	0.693686 ± 0.066431
	Tool Agent	289 ± 218	8367 ± 3103	0.001149 ± 0.000431	8657 ± 3297	**0.785554 ± 0.085655**
25	LLM	254 ± 67	**5035 ± 73**	**0.000908 ± 0.000041**	**5289 ± 97**	0.637390 ± 0.090126
	Tool Agent	**248 ± 244**	8056 ± 3275	0.001080 ± 0.000484	8305 ± 3493	**0.882115 ± 0.146230**
50	LLM	241 ± 68	9652 ± 145	0.001593 ± 0.000044	9894 ± 154	0.728411 ± 0.126316
	Tool Agent	**219 ± 223**	**7185 ± 2343**	**0.000961 ± 0.000325**	**7404 ± 2539**	**0.797055 ± 0.138048**

However, using language models alone is not scalable, as demonstrated in the Table 2. For this experiment, we ran each approach three times, increasing the number of clients, to understand the scalability of the strategies. As we can see, as we increase the number of clients, the number of

tokens increases, along with its cost for the pure method LLM. While the agent tool starts with a high value, the growth is slower. With 50 clients, the cost of LLM exceeds that of the agent. Although the standard deviation indicates a similar cost, the potential for improvement is promising with the improvement of the action and query tools, providing information intelligently and reducing costs. In addition to the cost reduction, we also observed a performance gain when using the models coupled to tools when we scale the number of clients, as the model becomes less confused.

Combined with sophisticated techniques such as ToT, long-term and short-term memory, and even advanced retrieval techniques and tools, agents represent promising approaches for the dynamic management of federated systems.