FlexiAI: A Multi-Agent Adaptive AutoML System Based on LLMs

Prithvi Pratahkal
Department of General Engineering
San Jose State University
San Jose, USA
prithvi.pratahkal@sjsu.edu

Sapna Sangmitra
Department of General Engineering
San Jose State University
San Jose, USA
sapna.sangmitra@sjsu.edu

Sheetal Khatri
Department of General Engineering
San Jose State University
San Jose, USA
sheetal.khatri@sjsu.edu

Sneha Lingadalli
Department of General Engineering
San Jose State University
San Jose, USA
snehashivakumar.lingadalli@sjsu.edu

Ahmed Hambaba
Department of General Engineering
San Jose State University
San Jose, USA
ahmed.hambaba@sjsu.edu

Abstract

Automated Machine Learning (AutoML) workflows remain complex, often requiring extensive human intervention for model selection, hyperparameter tuning, and data preprocessing. To address these inefficiencies, we present FlexiAI, an LLM-based multi-agent system designed to automate and accelerate ML pipeline development. FlexiAI translates natural language commands into formal workflow procedures, while distributing tasks across specialized agents responsible for model selection, feature engineering, optimization, and evaluation. Each agent operates in parallel within a decentralized architecture, sharing results through a meta-reasoning shared memory that enables cross-workflow collaboration and adaptive orchestration. This design minimizes bottlenecks, scales to large datasets, and improves system robustness through layered reliability verification. By combining parallel distributed optimization with context-aware reasoning, FlexiAI empowers both experts and non-experts to efficiently construct high-quality ML pipelines, reducing time and effort while increasing scalability, adaptability, and transparency.

1 Introduction

1.1 The evolution of machine learning integration into business processes and the automation problem

Healthcare, finance, and e-commerce industries are employing automation and gaining insights through data analysis (1). For instance, healthcare models can issue early warnings and e-commerce helps per real-time pricing and recommendation systems. Meanwhile, ML for finance aids in real-time fraud detection. Automating processes becomes burdening and error prone as developments call for more sophistication. Major hurdles are presented in coming up with advanced solutions that require automation to handle ML project lifecycles. The current situation in which ML processes

are manually managed wastes expert resources on trivial tasks, takes a lot of time, and risks model reproducibility and quality (2).

1.2 Limitations of current AutoML systems

Although AutoML systems have been developed to resolve some of the issues faced by machine learning systems, a great number of solutions still fall short of meeting diverse user needs. Industry-standard AutoML systems are typically characterized by fixed and rigid structural frameworks that are offered "patchwork modularity," failing to allow for genuine interactions defined and guided by the user (3). Such inflexibility hampers efforts aimed at Domain Customization or Knowledge Engineering as user workflows cannot be tailored to address specific needs. Besides, the "blackbox" problem is prevalent as traditional AutoML frameworks are criticized for their lack of transparency, making them unexplainable and difficult to trust (2). These systems tend to automate certain functions such as feature engineering or hyperparameter tuning, but if not explicitly directed, can struggle with nuanced datasets or specialized tasks, potentially undoing time savings. All manual ML pipeline optimization and the excess human involvement that undermine model performance, reduce quality, lead to highly technical systems, as well as defer automation tools, remains present even with some frameworks based on automation (2).

1.3 FlexiAI: a multi-agent adaptive AutoML system based on LLMs

This document introduces FlexiAI: a new system that combines large language models (LLMs) with a multi-agent system (MAS) to address these gaps. With FlexiAI, users can initiate automation of sophisticated ML processes through natural language commands. With FlexiAI, users can issue commands to automate ML processes from model training through deployment which makes complex ML workflows more user-friendly (3). FlexiAI aims to increase productivity, augment model reliability and strengthen operational adaptability by transforming natural language instructions into ML pipelines. The system broadens inclusivity by allowing non-specialists (4). The latest ML workflows integrated with LLMs boost automation in nearly all ML pipeline stages and suggest major advancements in productivity augmentation. Simultaneously, LLM-based MAS moves away from siloed model paradigms towards collaborative approaches where intelligent agents work together to complete sophisticated problems. This positions FlexiAI not simply as a further refinement of existing systems, but as a transformative interface redefinition of user engagement controllers with AutoML systems towards more fluid interactions in ML development. This strategy also helps resolve the "growing need for Comprehensible ML approaches" (1) considering a "shortage of talent in the AI and ML industries" by enfranchising and enabling self-sufficient teams. The work presented here describes the structure and core goals of FlexiAI, which captures it user-centric design integration prototype (5).

1.4 Contributions

The core contributions of FlexiAI include: An LLM-powered novel multi-agent framework for the automation of AutoML workflows end-to-end, which responds to free-text commands and enables advanced self-organization and decomposition among ML agents. A Shared Memory Module that permits contextual reasoning and cross-workflow meta-learning that allows for more optimal decisions based on the historical data and insights by the agents. An autonomous orchestrator for coordinating bounded distributed control of execution across modular services, along with sequential validation of emerging workflows to secure integrity, trust, and safety for flows as well as their LLM specific vulnerabilities. Greater accessibility and interpretability of instructional tasks and comprehensive model-driven evaluations reports through natural language, elevating user interactability and system explainability.

1.5 Structure of the paper

The rest of the paper is structured in the following way: In Section II, the FlexiAI architecture is described including its primary building blocks, and the relations between them. Section III analyzes innovations which FlexiAI proposes. In Section IV, we study the system's credibility along with its likely influence and sample scenarios demonstrating how it could be used. Introduction and outline of further research is given in section V, which is the concluding section of the paper.

2 FlexiAI architecture

2.1 Overview: a distributed multi-agent framework

FlexiAI is a distributed modular multi-agent system. Each agent in the system is an LLM-powered specialist for a certain stage or function in the ML workflow (6). This systemic architectural decision provides some advantages highlighted below. Modularity enhances maintainability by providing greater flexibility as components are now decoupled and can be individually developed, updated, or scaled. Each agent is able to specialize in a designated ML task, such as data preprocessing or model evaluation, and thus acquire in-depth knowledge of the particular area. The distributed attribute enables horizontal scalability and resilience. Multi-agent systems are designed to be scalable for complex problem solving since they are adaptable to changes in the environment by addition, removal, or modification of agents. Moreover, the loss of some control components tends to result in less impact on overall system performance and function, which enhances robustness and system dependability. What stands out in this architecture is the dynamic composability, where automated workflows are customized in real-time based on user input and evolving context rather than executed from a pre-defined static sequence.

2.2 Core components and their relationships

The interconnected core components that comprise the FlexiAI system are critical to support the automation of the entire machine learning workflow, each contributing a key function:

Natural language user interface & task planner LLM: Interaction with FlexiAI systems occurs through natural language commands, for example, "On this CSV dataset, train a fraud detection model" (4). The Task Planner LLM takes the user's high-level instructions as a challenge to solve. It uses its natural language understanding and reasoning capabilities to break down the user's intention into an ML pipeline or workflow at rest (4). This methodology removes obstacles to self-service, similar to how NLIDBs lowered entry barriers to data querying by removing the need for domain expertise in databases.

Specialized LLM-driven ML agents: The execution of the ML task is organized hierarchically using a set of specialized agents executed in sequence for different stages of the ML lifecycle, such as Data Preprocessing, Feature Engineering, Model Selection, Model Training and Evaluation (7). These are not simple, scripted agents: they are LLM-driven, hence capable of reasoning and possessing knowledge in relevant domains. For example, a Feature Engineering Agent could consult an LLM to propose appropriate transformations based on data attributes or generate explainable features.

Orchestrator module: As highlighted in the (5), the Orchestrator Module serves as the primary hub for control in the distributed system. It oversees task scheduling in accordance with the validated sequence, apportions those tasks to relevant domain agents, and manages the required data transfer for seamless information exchange. A crucial function of the Orchestrator is fault tolerance, which involves error capturing, non-response agent handling, and automatic retry or fallback processes. While some systems like MetaOrch use neural networks for agent selection, FlexiAI's Orchestrator concentrates on a dynamically executable pre-verified plan adjusted through multi-agent interactions and feedback loops.

Shared memory module & comparator agent: The Shared Memory Module is a critical centralized store which aids contextual reasoning for learners (5). It contains pipeline histories, model configurations, datasets, specialized summaries, significant feature quantifications, and other evaluation metrics. With this information, agents have access to contextual history which enables them to make decisions in a more informed manner. The Comparator Agent (5) works with this module to assess new executions alongside shared memory's previous run data, calculating accuracy, duration, and other relevant resources.

Layered cerification system (initial & stage-level verifiers): FlexiAI uses a layered verification system to maintain workflow integrity and accuracy. The Initial Verifier checks the high-level workflow outline created by the Task Planner LLM for structural logic, executability, and criteria coverage (4)). After this, Stage-Level Verifiers (Verifier 2) are added into each ML agent or module

Table 1: Key FlexiAI agents and their core responsibilities

Agent/Module Name	Core Responsibility	Key LLM Capabilities Leveraged	Interaction with Other Components
Task Planner LLM	Interprets natural language user commands; decomposes tasks into ML workflow plans.	Natural language understanding, reasoning, planning, structured output generation.	User Interface, Initial Verifier, Orchestrator
Initial Verifier	Validates logical coherence, execution plausibility, and parameter completeness of planned workflows.	Logical reasoning, constraint checking.	Task Planner LLM, Orchestrator
ML Agents (e.g., Data Preprocessing, Feature Engineering, Model Selection, Training, Evaluation)	Execute specific ML pipeline stages (e.g., data cleaning, feature generation, model training).	Domain-specific knowledge retrieval, code generation (for transformations), decision support.	Orchestrator, Shared Memory, Stage-Level Verifiers
Orchestrator Module	Manages task sequencing, agent assignment, data flow, fault tolerance.	Workflow management logic, state tracking, decision-making for retries/fallbacks.	All Agents, Verifiers, Shared Memory
Shared Memory Module	Centralized storage for pipeline history, metadata, model configs, evaluation metrics.	Data storage and retrieval.	All Agents, Comparator Agent, Optimization Agent
Comparator Agent	Compares current execution outputs (accuracy, resources) with historical data.	Pattern recognition, comparative analysis.	Shared Memory, Optimization Agent
Optimization Agent	Proposes enhancements to model quality or features based on analysis.	Problem-solving, recommendation generation, understanding of optimization strategies.	Comparator Agent, Orchestrator (for re-execution)
Stage-Level Verifiers	Ensure task-specific correctness, compatibility, and alignment within each ML agent/module.	Constraint checking, logical consistency verification at a granular level.	ML Agents
Report Generation Agent	Synthesizes comprehensive reports on pipeline structure, decisions, performance.	Natural language generation, data summarization, visualization integration (e.g., SHAP, ROC).	Orchestrator, Shared Memory

(5). These verifiers check the finer details like data transfer stage compatibility, metrics, and models relevancy to the tasks at hand.

Optimization agent & report generation agent: The Optimization Agent examines the outcomes given by the Comparator Agent and other system parameters to suggest benchmarks or alterations aimed at improving the model, feature set, and overall pipeline usability (5). It has the capability of initiating controlled re-execution of the relevant pipeline sub-stages. The Report Generation Agent has the role of summarizing the entire machine learning (ML) process in detailed yet easily comprehensible forms (7).

2.3 Assumptions

The design of FlexiAI relies on several underlying assumptions to ensure feasibility and correct operation:

- **Users can articulate goals through prompts:** End-users are assumed to express their requirements in natural language to allow accurate intent recognition and workflow generation.

- **Trusted execution environment:** All agents function in a pre-defined, controlled, and non-hostile environment.
- **Assumption of available structured tabular data:** Input data is expected in structured formats (e.g., CSV) to minimize manual preprocessing.
- **Stable inter-agent communication:** Communication between agents is reliable, low-latency, and loss-free.
- **Clear and non-hostile data inputs:** Datasets are assumed free from adversarial noise, poisoning, or manipulative inputs.
- **Consistent and centralized memory is accessible:** The Shared Memory Module provides a consistent and up-to-date context accessible to all agents.
- **Task-specific reasoning capabilities of LLM modules:** Integrated LLMs are assumed to be sufficiently knowledgeable to perform tasks such as model selection, feature engineering, and hyperparameter tuning.
- **Adequate hardware resources:** It is assumed that sufficient CPU, GPU, and memory resources are available for model training and evaluation.
- **Accessible workflows and historical results:** All past runs and evaluations are logically stored and retrievable for dynamic learning and comparisons.
- **Minimal user configuration:** Users are assumed to prefer passive interaction, with the system handling end-to-end automation.
- **Security and privacy at deployment:** Security measures (e.g., encryption, access control) are assumed to be managed at deployment rather than prototype level.
- **Ethical and legal compliance:** Data and models are assumed to adhere to privacy laws and ethical guidelines, permitting their processing and use.

2.4 Illustrative workflow execution (high level)

Below is a description of a high-level illustrative steps carried out while executing workflow in FlexiAI. User Command: The user issues a high-level ML task command in free form natural language. For instance, "Predict customer churn using dataset X and optimize for F1 score." Workflow Planning Initial Verification: The Task Planner LLM translates the provided command into a structured workflow plan. Initial Verifier cross-checks the plan for coherence, logical consistency and whether it is achievable. Orchestration Task Dispatch: The verified plan is handed to the orchestrator who organizes all components to fulfill the plan. They divide the higher level tasks into more granular subtasks which are then sent to the relevant specialized, LLM-driven agents. Agent Execution Stage-Level Verification: Every agent interacts with the Shared Memory regarding context and data to carry out an assigned sub-task. Operations and outputs for each agent are confirmed by Stage-Level Verifiers for integrity and coherence. Feedback Optimization: The Feedback computation agent performs a comparison evaluation of the results against historical performance records. The Optimization Agent, using the data, can initiate stage reruns or suggest improvements. Reporting: The automated reporting process is performed by Report Generation Agent which creates a summary document detailing the process, results, and insights as a comprehensive report for the user. The whole process is set up to be iterative with the various feedback loops allowing the continuous enhancement of the ML models and workflows. Particularly, the Shared Memory and Comparator/Optimization loop introduces a workflow level meta-learning framework, thus letting FlexiAI learn from prior diverse task and dataset executions cross-task and dataset diverse execution history instead of just optimizing one pipeline instance.

3 Key innovations of FlexiAI

3.1 Intelligent automation via LLM-powered agents

FlexiAI's intelligence is based on LLM-powered agents, which form its core. These agents, unlike rule-based automation, are capable of complex multi-step reasoning, planning, and execution with different ML tasks. For instance, an LLM-powered Model Selection Agent can perform more than relational database lookups. Utilizing its knowledge of ML, it interprets the dataset's attributes, the

task, and even subtle user cues derived from the natural language prompt (4) to make model selection decisions. LLM agents focused on scientific discovery have learned to automate hypothesis- or experiment-design as well as data analysis using domain-specific knowledge and toolsets, which is parallel to the way FlexiAI agents automate the stages of an ML pipeline. This feature of "intelligent self-organization and decomposition" enhances the degree of context sensitivity in automation.

3.2 Adaptive modularity and reconfigurable orchestration

Multi-agent systems augmented with an intelligent Orchestrator Module provide FlexiAI with adaptive modularity and reconfigurable orchestration (4), (5). This implies that workflows are adaptive and can be constructed and adapted in real time considering the specific problem at hand, the available data, and even previously generated intermediate results. The modular flexibility means that new agents can be introduced (for new ML techniques or data modalities), or existing ones updated or replaced, with no need for a systemic overhaul. The Orchestrator undergoes evolution from the system perspective where automation is performed sequentially in an automation pipeline to a reactivity-driven distributed ML orchestration system that integrates and manages advanced dependencies and real-time variable changes.

3.3 Contextual reasoning and cross-workflow meta-learning

Innovation stems from the addition of the Shared Memory Module and the Comparator Agent, which allow for contextual reasoning and Cross-Workflow meta-learning (5). The Shared Memory Module serves as an overarching chronic long-term memory of the system and retains information concerning past executions of pipelines along with their models, features, and pertinent metrics calculated during and after execution. This retrospective context aids agents in better-informed decision-making attuned to the current tasks at hand. The Comparator Module utilize the information collected in their historical databases for assessing their current models with their historical ones. This forms a feedback loop which, in combination with Optimization Agent, allows a sort of meta-learning where the system learns from experience spanning different, and often diverse, workflows.

3.4 Enhanced clarification and flexibility of users through natural languages

The explainability level of FlexiAI systems is remarkably elevated due to its user natural language interface and report generation features. In this case natural language processing is beneficial because it allows users regardless of their knowledge background to express ML problems in their own words which alleviates the most users barrier to entry (1), (4). Moreover, other Shapley-based analyze and evaluation plots (7) offer detailed report generation that gives users clear overviews of the decision process, global steps of pipeline creation, heuristics employed, outcome indicators, and more granular details such as specific model's outputs.

3.5 Security and robustness by design

FlexiAI takes specific measures to incorporate security and robustness by design (8). For example, susceptibilities to prompt injection and generating hallucinations are vulnerabilities pertaining to LLMs. To combat this attack, FlexiAI employs a layered verification approach. These layered agents verify logical completeness, task alignment, and confirm compliance against the set constraints in the system, ensuring safety against nonsensical workflows being executed. The deep blending of these components is what truly brings out the innovation of FlexiAI. The use of LLMs or MAS separately does not drive innovation; it is their synergistic integration that stems a system that is powerful in its reasoning (LLMs), flexible in its operations (MAS modularity and orchestration), easy for interfacing (NLI), all while focusing on being dependable and safe.

4 Feasibility, potential impact, and use cases

4.1 Technical feasibility and implementation strategy

The sections from (5), tell us that the FlexiAI system is feasible from a technological perspective in light of new opportunities emerging from innovation and the availability of its core technologies. A

GPT-4 or Claude based production level LLMs, as well as competing ones, fulfill the sophisticated requirements in natural language understanding and generation for the Task Planner and specialized agents. (5) reveals that agentic frameworks like LangChain and AutoGen provide abstraction over tools which can help speed up the realization of the multi-agent system. The structure in question appears to have been created to aid in the deployment of cloud-native applications, which makes automated scaling and distributed processing possible (5). In addition, safety measures, as well as measures to mitigate hallucinations and logical inconsistencies, will be taken using techniques for steering and constraining LLMs. These include structured prompt templates, input sanitization, and filters that validate the logic of the ML pipeline safeguards (5). The development of FlexiAI is divided into different phases. As outlined in the scope of the project, Phase I includes outlining the key architectural skeleton of the system and building the first version of core modules which include the Task Planner, Orchestrator, critical ML agents, Shared Memory, and Verifiers, as well as implementing natural language driven basic workflow execution (6).

4.2 Illustrative applications and potential impact

Practical cases are as follows (3): Healthcare: Constructing models for early warning systems draws from an organization's EHRs. With these models, a clinician is alerted in advance, allowing him/her to take necessary actions to avert or reduce patient deterioration. Finance: Streaming fraud detection systems have been constructed and infuse the ability to be retrained on live transactional data streams. E-commerce: Automated systems forecast demand with multi-channeled advanced pricing that adjusts in real time, driving higher sales and better inventory management. Aside from these applications, the overarching impact of FlexiAI remains noteworthy: Democratizing ML: Addressing automation gaps, FlexiAI allows for ML task description via natural languages overcoming multi-step automation blocks through natural languages. Accelerating Research and Development: The system serves as an asset for researchers and data scientists. Enhancing Operational Efficiency: With FlexiAI, data-driven operations are more agile and responsive since manual effort, operational costs, and deployment delays linked to ML projects are streamlined.

5 Final remarks and prospects for further research

5.1 Overview of advancements with FlexiAI

The problem of ineffective and convoluted ML workflow management is alleviated with FlexiAI's deployment of a novel LLM-based multi-agent architecture. This system allows the automation of processes through natural language commands, turning them into executable, optimized ML workflows. Important features of this system include: modular and adaptive architecture enhancements; the implementation of a Shared Memory Module integrated contextual reasoning cross-workflow meta learning for sophisticated inference and multi-tasking; reinforced layered verification towards more robust reliability; and further accessibility and explainability for users and systems.

5.2 Further work and expanding research proposals

Completing Phase I will capture the basic infrastructure of FlexiAI. Expanding functionality, agility, and resilience will be aimed for in the later stages of development. Prior work includes the following: Improving Agent Functionality. Advanced Orchestration Strategies. The Federated Learning Agent. The Real-time Drift Remediation Agent. Support for multi-modal agents. Behavior of the system is proactive. Broadening Scope as a Research Platform. With these enhancements, FlexiAI seeks to evolve as an intelligent domain-specific platform for self-directed automation and advancing the technology of machine learning.

References

[1] N. D. Lawrence, "Machine learning from innovation to deployment: A strategic research agenda for autoai," 2020. Available:https://mlatcl.github.io/papers/autoai-sra.pdf.

[2] T. Nagarajah and G. Poravi, "A review on automated machine learning (automl) systems," in *Proc. IEEE 5th Int. Conf. for Convergence in Technology (I2CT)*, Bombay, India, 2019, pp. 1–6.

[3] S. Jayanthi and S. P. Devi, "Automated machine learning on high dimensional big data for prediction tasks," in *Proc. 7th Int. Conf. on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 2022, pp. 995–1001.

[4] S. Chen, W. Zhai, C. Chai, and X. Shi, "Llm2automl: Zero-code automl framework leveraging large language models," in *Proc. Int. Conf. on Intelligent Robotics and Automatic Control (IRAC)*, Guangzhou, China, 2024, pp. 285–290.

[5] N. Nascimento, P. Alencar, and D. Cowan, "Self-adaptive large language model (llm)-based multiagent systems," David R. Cheriton School of Computer Science, University of Waterloo, Canada, 2023. Available:https://cs.uwaterloo.ca.

[6] C. R. J. Romero, J. Chang, and J. M. Montoya, "Multi-agent systems powered by large language models: Applications in swarm intelligence," 2025, arXiv preprint arXiv:2503.03800v1. Available:https://arxiv.org/abs/2503.03800.

[7] A. Salhi, A. C. Henslee, J. Ross, J. Jabour, and I. Dettwiller, "Data preprocessing using automl: A survey," in *Proc. Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, Las Vegas, NV, USA, 2023, pp. 1619–1623.

[8] A. Jin, Y. Ye, B. Lee, and Y. Qiao, "Decoagent: Large language model empowered decentralized autonomous collaboration agents based on smart contracts," *IEEE Access*, vol. 12, pp. 119438–119453, 2024.

Agents4Science AI Involvement Checklist

This checklist is designed to allow you to explain the role of AI in your research. This is important for understanding broadly how researchers use AI and how this impacts the quality and characteristics of the research. **Do not remove the checklist! Papers not including the checklist will be desk rejected.** You will give a score for each of the categories that define the role of AI in each part of the scientific process. The scores are as follows:

- **[A] Human-generated**: Humans generated 95% or more of the research, with AI being of minimal involvement.
- **[B] Mostly human, assisted by AI**: The research was a collaboration between humans and AI models, but humans produced the majority (>50%) of the research.
- **[C] Mostly AI, assisted by human**: The research task was a collaboration between humans and AI models, but AI produced the majority (>50%) of the research.
- **[D] AI-generated**: AI performed over 95% of the research. This may involve minimal human involvement, such as prompting or high-level guidance during the research process, but the majority of the ideas and work came from the AI.

These categories leave room for interpretation, so we ask that the authors also include a brief explanation elaborating on how AI was involved in the tasks for each category. Please keep your explanation to less than 150 words.

1. **Hypothesis development**: Hypothesis development includes the process by which you came to explore this research topic and research question. This can involve the background research performed by either researchers or by AI. This can also involve whether the idea was proposed by researchers or by AI.

   Answer: **[A]**

   Explanation: The research topic and system design hypotheses were conceived and structured entirely by the human authors, drawing on prior literature and domain expertise. AI support was minimal, limited to wording suggestions rather than idea generation.

2. **Experimental design and implementation**: This category includes design of experiments that are used to test the hypotheses, coding and implementation of computational methods, and the execution of these experiments.

   Answer: **[B]**

   Explanation:While Phase A (conceptual architecture) was primarily developed by the human authors, AI tools were used in a supporting role for brainstorming possible design alternatives and clarifying implementation strategies. Phase B (implementation and experiments) is ongoing work and not reported here, but any AI use remains assistive rather than primary.

3. **Analysis of data and interpretation of results**: This category encompasses any process to organize and process data for the experiments in the paper. It also includes interpretations of the results of the study.

   Answer: **[B]**

   Explanation: Since this paper presents Phase A (conceptual architecture) without experimental data, there was no formal data analysis. However, AI tools were occasionally used to assist in interpreting potential system behaviors and comparing design choices, with human authors leading the reasoning and final interpretations.

4. **Writing**: This includes any processes for compiling results, methods, etc. into the final paper form. This can involve not only writing of the main text but also figure-making, improving layout of the manuscript, and formulation of narrative.

   Answer: **[B]**

   Explanation: Human authors drafted the core content, figures, and structure, while AI tools were used in a supportive role for polishing language, improving clarity, and suggesting narrative refinements. The final text reflects human-led integration and domain expertise.

5. **Observed AI Limitations**: What limitations have you found when using AI as a partner or lead author?

Description: AI assistance often produced verbose or repetitive phrasing and occasionally introduced technically inaccurate statements that required human correction. It also lacked the ability to reason deeply about domain-specific design trade-offs, so human oversight was necessary to ensure precision and coherence.

Agents4Science Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the motivation, problem, and FlexiAI solution, and the listed contributions match the conceptual scope of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [NA]

   Justification: The paper focuses on conceptual architecture (Phase A) and does not include a dedicated discussion of limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This Phase A paper does not include formal theorems or proofs; instead, we outline key system assumptions that guide the conceptual design. In Phase B (implementation and experiments), these assumptions will be validated or refined as practical constraints are encountered.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [NA]

   Justification: The paper does not include experiments in this phase; it focuses on conceptual system design, so reproducibility of results is not applicable.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [NA]

   Justification: The paper does not report experiments requiring code or data; it focuses on conceptual architecture design, so open access to code and data is not applicable.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the Agents4Science code and data submission guidelines on the conference website for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [NA]

   Justification: The paper does not include experiments; it is focused on conceptual system architecture, so training and testing details are not applicable.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: The paper does not include experiments; it presents conceptual architecture only, so statistical significance reporting is not applicable.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, or overall run with given experimental conditions).

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: The paper does not include experiments; it focuses on conceptual system design, so compute resource reporting is not applicable.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the Agents4Science Code of Ethics (see conference website)?

   Answer: [Yes]

   Justification: The work follows the Agents4Science Code of Ethics, as it focuses on system design that emphasizes transparency, reliability, and safe use of LLM-based agents without ethical violations

   Guidelines:

   - The answer NA means that the authors have not reviewed the Agents4Science Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: The paper highlights positive impacts such as democratizing ML access and improving efficiency, and it also discusses risks like robustness issues and vulnerabilities in LLM-based systems, along with mitigation strategies.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations, privacy considerations, and security considerations.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies.