Quarterly Newsletter

CHAIR iDANAE

INTELLIGENCE · DATA · ANALYSIS · STRATEGY

2Q25

GenAI: an approach to multi-agent sytems

UNIVERSIDAD POLITÉCNICA DE MADRID

POLITÉCNICA

Management Solutions

Making things happen

Introduction

Large Language Models (LLMs) are deep neural networks based on the transformer architecture that process and generate text by learning from vast amounts of data. These models have excelled at tasks such as translation, summarization, and content creation [1]. Indeed, the incorporation of LLMs within some AI systems has allowed to develop engines and tools capable of executing tasks, mainly related to the use of natural language, that could not be conducted automatically, or that outperform existing AI systems, changing the way natural language tasks are approached, and offering new capabilities in reasoning, generation, and information retrieval.

As these models grow and new applications are developed, they have been increasingly integrated into more complex AI systems, including the combination with features, such as the use of Retrieval-Augmented Generation (RAG)[1], encoders, or other elements to better perform the tasks and achieve the objective.

However, the complexity of the activities to be assumed and executed by an AI system constantly requires more powerful structures, that combine expert knowledge and capabilities with different nature of solutions. One option to address this requirement is to construct a system where different agents (LLM, database managers, interpreters, models, etc.) can interact among themselves, capitalizing the efficiency of each one, but ensuring the effectiveness of the system without compromising its security.

In this solution, multiple autonomous agents work together to solve problems, make decisions, and achieve goals. These systems, called "Multi-agent Systems" (MAS), mimic the collaborative nature of human societies, leveraging the strengths and capabilities of individual agents to create a more robust and adaptive solution. By enabling agents to communicate, negotiate, and coordinate their actions, MAS are paving the way for advancements in areas such as robotics, smart grids, and complex simulations.

This newsletter aims to explore how MAS can enhance the quality, accuracy, and robustness of AI systems. To this end, in section 2 main technical components of a MAS are described, and in section 3 a practical application has been developed to exemplify the possibilities of these systems and show some metrics that can be used to measure the performance. Finally, in section 4 some reflexions on the challenges of MAS for business are included.

[1]RAG combines the strengths of LLMs with external knowledge sources to enhance the quality and relevance of generated content. By integrating retrieval mechanisms, RAG systems can access and incorporate up-to-date information from databases and documents, resulting in more informed and contextually appropriate outputs.

Multi-agent systems: a brief review

Concept

An AI system is considered a Multi-agent system (MAS) when multiple autonomous agents work together to solve problems, make decisions, and achieve goals. In a simple version, a MAS is composed of multiple instances of LLMs (agents) that work together to solve more complex problems, adapting to a human-like behaviour to make decisions [6]. For this, each agent is assigned a role in a specific domain and then all agents communicate and cooperate to get the best answer [7]. For instance, in an article-writing MAS, one agent could be the planner who collects the content to write the article, another agent could be the writer who writes the article, based on a expert-knowledge database, and a third agent could be the editor that reviews the article. Eventually, all these three agents debate about each other responses and cooperate to write the best possible article.

According to the communication strategy across agents, different alternatives can be designed for a MAS to be structured [7], [8]:

- **Cooperative agents:** all agents have the same goal. They work together and exchange information to get to a common solution.
- **Debate or mixed agents:** each agent has their own point of view or goal. Thus, they argument and critique each other's answers to get to a common and more refined solution.
- **Competitive agents:** they are like debate agents, but they compete for the best point of view instead of ending in common solution.
- **Hierarchical agents:** it is an approach in which agents are organized in a hierarchical structure (usually a tree) to enhance task decomposition among them. Then, agents in parent nodes assign tasks to agents in child nodes.

Building a MAS

When building a MAS several modules can be designed and developed [9], to ensure that all functions are coordinated in an efficient way:

1. **Profile module:** it refers to the module in which the roles are assigned to each agent. Roles are mostly assigned manually; however, they could be automatically created and assigned by LLMs or extracted from a database that contains possible human-like roles.
2. **Memory module:** it is necessary to keep track of the users' queries and LLM generations within a context (short-term) or to have a record of more information over time (long term). Then, memory can be stored in various data formats and data structures such as databases, embeddings, or even natural language (human-readable structures).

3. **Planning module:** it tries to make agents acquire a more human-like behaviour when decomposing tasks to get to their goals. Then, planning can be centralized, in which one agent controls the planning process of all the agents, or decentralized, which each agent plans its workflow independently. The planning module can be controlled by feedback. When there is no feedback, the tasks can be decomposed and executed sequentially, they can be organized into a hierarchical structure (trees), or they can rely on an external planner. If the module includes feedback, it can be obtained from the environment, from humans or from other agents or LLMs.

4. **Action module:** it manages all the agents' decisions and outcomes. Thus, it considers how the goals are being completed, what tools need to be used and which the consequences of the different agents' actions are.

Although the creation and of MAS has been very recent, some frameworks used to build agents have already been developed. Following, there are some of the frameworks widely used [10]:

- **LangGraph:** designed by Langchain, it uses a workflow that is based on directed acyclic graphs (DAGs) to model MAS, in which tasks and functions are contained withing each node of the graph.
- **Autogen:** designed by Microsoft, it is based on creating agents with different roles. Then these agents cooperate and communicate with each other to fulfil their tasks. However, it only supports short-term memory, only keeping track of recent interactions in the context window.
- **CrewAI:** similarly to Autogen, agents with specific objectives are created to achieve a common goal. Unlike Autogen, CrewAI does support long-term memory, allowing it to keep track of past interactions. Moreover, it is built on top of LangChain, which allows it to use and build more personalized tools.

A practical application: a brief review

Within the iDanae Chair, a MAS system has been developed to test a MAS architecture. The goal of the MAS is to create an expert Q&A application, where a user can ask questions on ESG risk management for the banking sector (with a focus on climate and environmental risks), including regulation, modelling techniques, data used, etc. In this section, the steps taken to develop the system (from the dataset creation to the evaluation) are outlined, a comparison to other simpler approaches is included, and some improvements at each stage are highlighted.

Developing the systema

Dataset creation

The foundation of this MAS is a structured dataset consisting of questions and answers (Q&A) on ESG risk management. To this end, each entry in the dataset contains a question related to different topics on ESG risk management and a ground-truth answer based on official documentation (such as the European Guidelines on risk management from the European Banking Authority, the ECB supervisory expectations), modelling practices applied in the banking sector, scientific knowledge on modelling and data science techniques, or information of available databases. This dataset was manually created and revised to facilitate retrieval-based learning, thus serving as a benchmark to assess the accuracy of the responses generated by the system.

RAG development

The first implementation involved a basic RAG system, which was built to retrieve relevant ESG-related information and generate responses based on the retrieved chunks. It followed the following stages:

- **Database Creation:** Qdrant [11], a high-performance vector database, was used to store and retrieve document embeddings. ESG guidelines and documents were chunked into smaller sections and stored as vector embeddings for efficient retrieval. These vectors were obtained using the all-MiniLM-L6-v2 model [12], an embedding model developed by Microsoft, designed for efficient semantic search, clustering, and text similarity tasks.
- **Query Processing:** The system was created using Amazon Bedrock from Amazon Web Services (AWS) [13] and the Claude-3-Haiku model [14]. Then, the query input was converted into an embedding and compared against stored embeddings to retrieve relevant chunks, and finally the model generated a response based on the retrieved document chunks.

Once created, with the aim to enhance response accuracy, an advanced RAG was implemented by incorporating a re-ranking model (amazon.rerank-v1:0) [15]. After retrieving multiple document chunks based on the query, the re-ranker model was applied. This model assigned relevance scores to each chunk based on embedding similarity. Only chunks with scores above 0.6 were selected as input for the LLM. This process eliminates low relevance chunks and improves the contextual accuracy of responses.

Development of the MAS

The final enhancement was the creation of a Multi-Agent System to improve the robustness of ESG-related responses. The MAS was built using AutoGen [16], a tool that helps set up and manage AI-powered agents working together. With AutoGen's GroupChat setup, the agents could easily communicate, share tasks, and improve responses step by step.

The initial approach used a single-agent system with AutoGen to generate responses. This system consisted of only one agent responsible for retrieving documents responses. For this, the agent used a tool that included the reranker model to retrieve the relevant ESG documents and then formulated an answer to the query.

Then, the system was extended into a MAS using AutoGen's GroupChat, allowing multiple agents to collaborate in a structured process. The system was configured to run autonomously, that is, without human intervention, and without allowing an agent to speak twice in a row, ensuring a balanced participation. The MAS consisted of the following agents replying sequentially in the given order:

Retrieval Augmented Generation (RAG)

While LLMs have proven to be powerful, their entire reliance on training data can lead to inaccurate or outdated responses. To mitigate this, techniques like Retrieval-Augmented Generation (RAG) have emerged as an innovative solution.

RAG is a technique that combines the text generation ability of LLMs with the retrieval of relevant information (chunks) from documents stored in an external database. In this way, the LLM uses the retrieved chunks as an additional context to generate a more precise and/or elaborate answer, therefore reducing hallucinations [2].

The approach that RAGs follow in order to manage these chunks is having the so-called retriever. This retriever is composed of two modules, one for building and one for querying. In the building module, the text documents are cut into chunks of a pre-defined length and then each chunk is encoded into vector embeddings (numeric representations of the words in each chunk depending on their context and definition) to be stored in the vector database. Then, the querying module encodes the LLM's query into vector embeddings and uses similarity search techniques in the stored embedded chunks to find the information that better aligns with the query [3].

Several types of RAG systems can be developed [2]:

- Basic or naive RAG: it uses the previously defined logic to extract the relevant chunks from the database. Then, these chunks are added as an additional context in the LLM prompt, that is, the LLM is asked to generate an answer to the query using the retrieved information. This workflow is illustrated in Figure 1.
- Advanced RAG: it uses the same structure as the basic RAG, but with the difference that the chunk search is optimized to retrieve more refined embeddings. This can be achieved by improving the embedding methods, data structures and queries of the retriever and by evaluating the obtained chunks to check which ones of them are the most significant. The second method uses a re-ranking concept in which an LLM (usually one that is specifically designed for re-rank tasks) takes as an input all the retrieved chunks and, using similarity metrics of the embeddings, generates a score for each one of them depending on how well they answer the query. Then, only the chunks with the highest score are chosen to be introduced as the context of the regular LLM to generate an answer. One of the most effective models for this task is Cohere's ReRank [5], which uses deep learning techniques to accurately prioritize the most relevant chunks, significantly improving retrieval performance in RAG architectures. This workflow is illustrated in Figure 2.

Additionally, there exists an adaptive retrieval process for RAGs in which the LLM decides whether it is sufficient to use its own knowledge to generate an answer or if context retrieval is needed for refining it. In this context, LLMs act as autonomous agents to make decisions on their operations.

Figure 1. Basic RAG architecture [4]

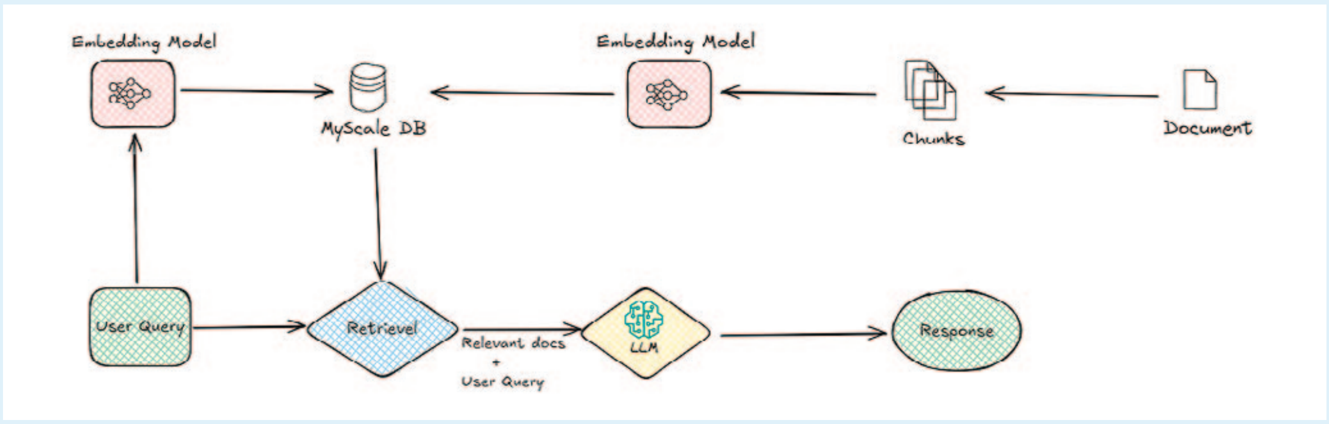
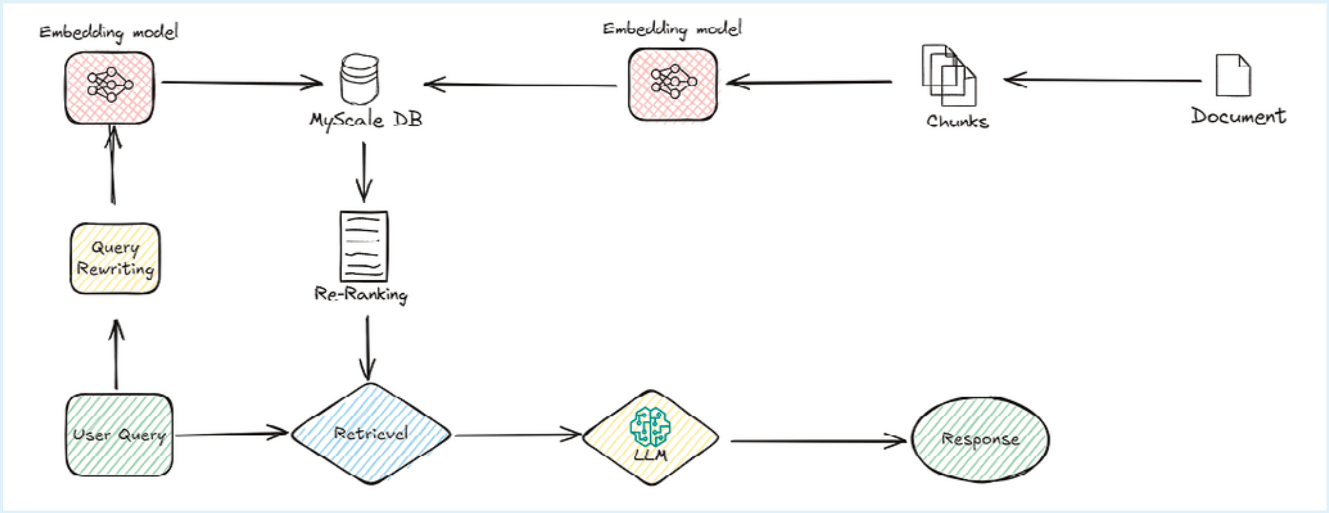


Figure 2. Advanced RAG architecture [4]



- **Context Tool:** retrieved relevant ESG documents from Qdrant using RAG and re-ranking.
- **Researcher:** extracted context and generated an initial response based on retrieved data.
- **Critic:** evaluated the researcher's response, identifying inconsistencies or missing elements without generating an answer itself.
- **Synthesizer:** Combined feedback from both agents into a concise and well-structured final answer, and ensured the final output is concise, accurate, and free of unnecessary critique.

Evaluation and performance

To evaluate the effectiveness of the RAG and the MAS systems, a structured evaluation framework was implemented. An LLM was used to automatically measure the similarity between real answers from the dataset and the responses generated by both RAG and MAS systems. This similarity ranking determined whether the most relevant information was being retrieved. Additionally, the Word Error Rate (WER) [17] was calculated between the real answers and the generated responses. This measures the discrepancy between a generated response and the reference text by measuring the percentage of words that need to be inserted, deleted, or substituted to match the correct answer.

Analysing the results

To evaluate the effectiveness of the RAG and the MAS systems, a structured evaluation framework was implemented. An LLM was used to automatically measure the similarity between real answers from the dataset and the responses generated by both RAG and MAS systems. This similarity ranking determined whether the most relevant information was being retrieved. Additionally, the Word Error Rate (WER) [17] was calculated between the real answers and the generated responses. This measures the discrepancy between a generated response and the reference text by measuring the percentage of words that need to be inserted, deleted, or substituted to match the correct answer.

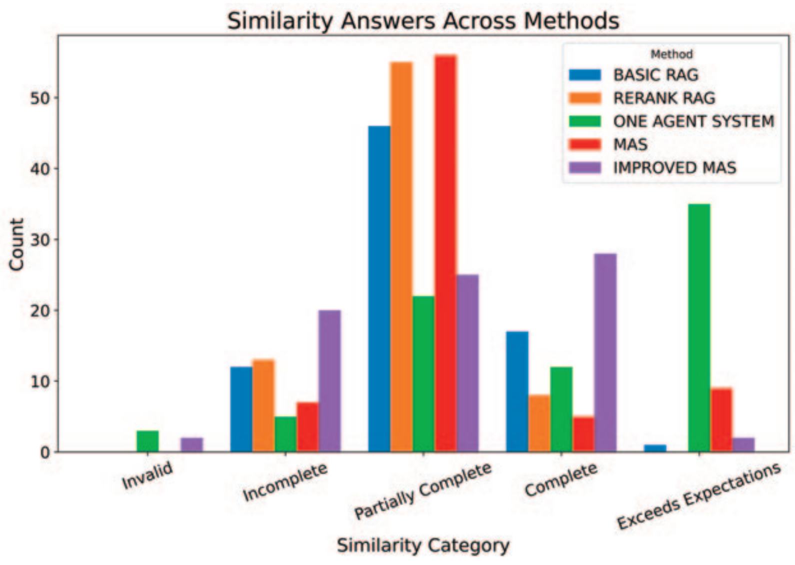
Similarity answers

Figure 3 shows how the LLM that was used for answer evaluation classified the systems' answers. It divided these answers into five categories: Invalid, Incomplete, Partially Complete, Complete, and Exceeds Expectations. These categories reflect how well the answer captures the intent and content of the reference response.

The basic RAG generated a large number of Partially Complete answers, which lacked completeness and missed key points. A smaller portion of answers were judged Complete, and only a few reached the Exceeds Expectations category. The presence of both Incomplete and Invalid answers indicates that this system struggled with consistency and often failed to retrieve relevant content.

The advanced RAG followed a similar pattern to the basic RAG, with most answers classified as Partially Complete. The number of Complete responses was slightly lower than in the basic

Figure 3. Evaluation RAG similarity answers for all the systems.



version and Exceeds Expectations answers were not present. While re-ranking helped reduce irrelevant outputs, answers were not as deep, that is, they were more focused mostly on the retrieved context.

The single-agent system showed greater variability. Many responses were classified as Exceeds Expectations, reflecting its ability to generate richer answers. However, it also produced several Partially Complete and Complete responses, along with occasional Incomplete or Invalid ones. The system's lack of internal feedback likely contributed to this inconsistency, where some responses were well-developed and others missed the main point.

The MAS improved the consistency of responses compared to the single-agent setup. Most answers were still Partially Complete, but there was an increase in both Complete and Exceeds Expectations outputs, while Incomplete and Invalid responses were minimal. The collaboration between agents, particularly the use of a Critic and Synthesizer, helped enhance semantic completeness. However, the high proportion of Partially Complete answers suggests issues with inter-agent coordination.

The improved MAS had the most balanced performance. It produced a higher proportion of Complete answers while reducing the number of Incomplete and Invalid outputs. This improvement can be linked to better prompt design and clearer agent instructions, which helped to address role ambiguity and task alignment problems.

WER comparison

The comparison between Basic RAG and the re-ranked version reveals key differences in output quality.

- The WER of the basic RAG shows the retrieved context is sometimes only loosely related to the question, resulting in answers that are too generic or off-topic. As a result, the generated responses require many edits, leading to high WER values.

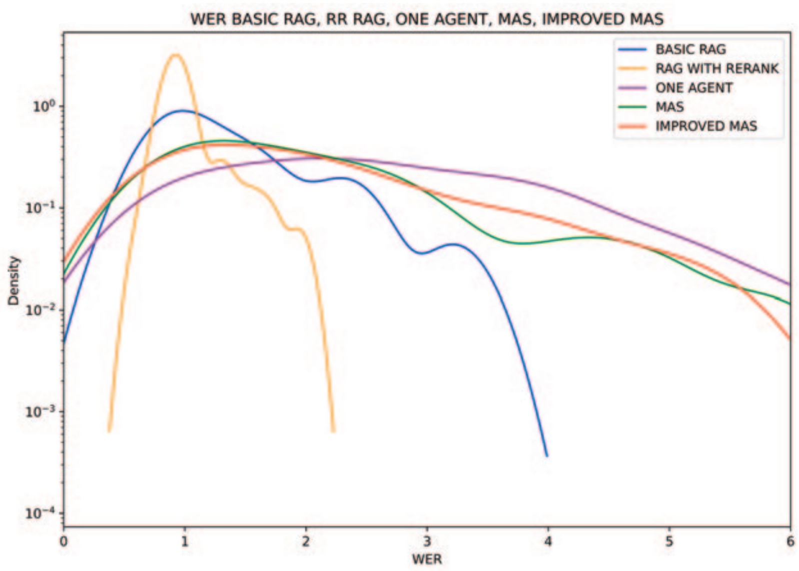
- The advanced RAG shows lower WER across most responses. The reranking process improves the relevance of the retrieved context, which leads to more accurate outputs. However, many of these answers lack depth. In some cases, the system merely reformulates the retrieved chunk without adding meaningful reasoning or synthesis.

The comparison between the agent-based systems highlights the impact of internal feedback, planning, and role definition.

- The single-agent system produces answers with moderate WER. While it can reason beyond retrieved content, the lack of internal review means the responses often contain inconsistencies or unnecessary elaborations. The performance is very variable, and many answers require significant editing to align with the reference.

- The initial MAS shows a clear improvement over the single-agent setup. The use of multiple agents with distinct roles helps to catch issues earlier and refine the answer before final output. As a result, WER values are generally lower. However, some responses remain inconsistent.

Figure 4. TER distribution across the different systems.



- The improved MAS achieves the best WER performance among all MASs. Prompt refinement leads to clearer task definitions and better cooperation between agents. This setup helps avoid misinterpretation and unnecessary content, resulting in concise and accurate responses. However, due to its more complex reasoning skills this system is still outperformed by the advanced RAG.

The previous analysis shows that systems with more Exceeds Expectations answers, like the single-agent and MAS setups, do not achieve the lowest WER. The advanced RAG, despite having mostly Partially Complete answers and no Exceeds Expectations, achieves better WER than the improved MAS. This suggests that producing more complete answers does not directly correspond to requiring fewer edits. To better analyze and understand how semantic quality relates to structural accuracy, all WER distributions were analyzed across the different similarity categories independently, obtaining the following conclusions (see Figure 5):

- The basic RAG shows similar WER values across the Incomplete, Partially Complete, and Complete categories, with no clear difference among them. This suggests that even when responses appear semantically stronger (like classified as Complete), they still require considerable editing. The lack of differentiation likely results from generic and poorly structured document retrieval.

- The advanced RAG presents a different scenario, where Incomplete answers have notably lower WER compared to the Partially Complete and Complete categories. This occurs because incomplete responses are very short, closely matching reference answers at a certain token level despite semantic inadequacy. The WER for more complete answers increases sharply, showing that when responses have greater detail, structural mismatches occur frequently.

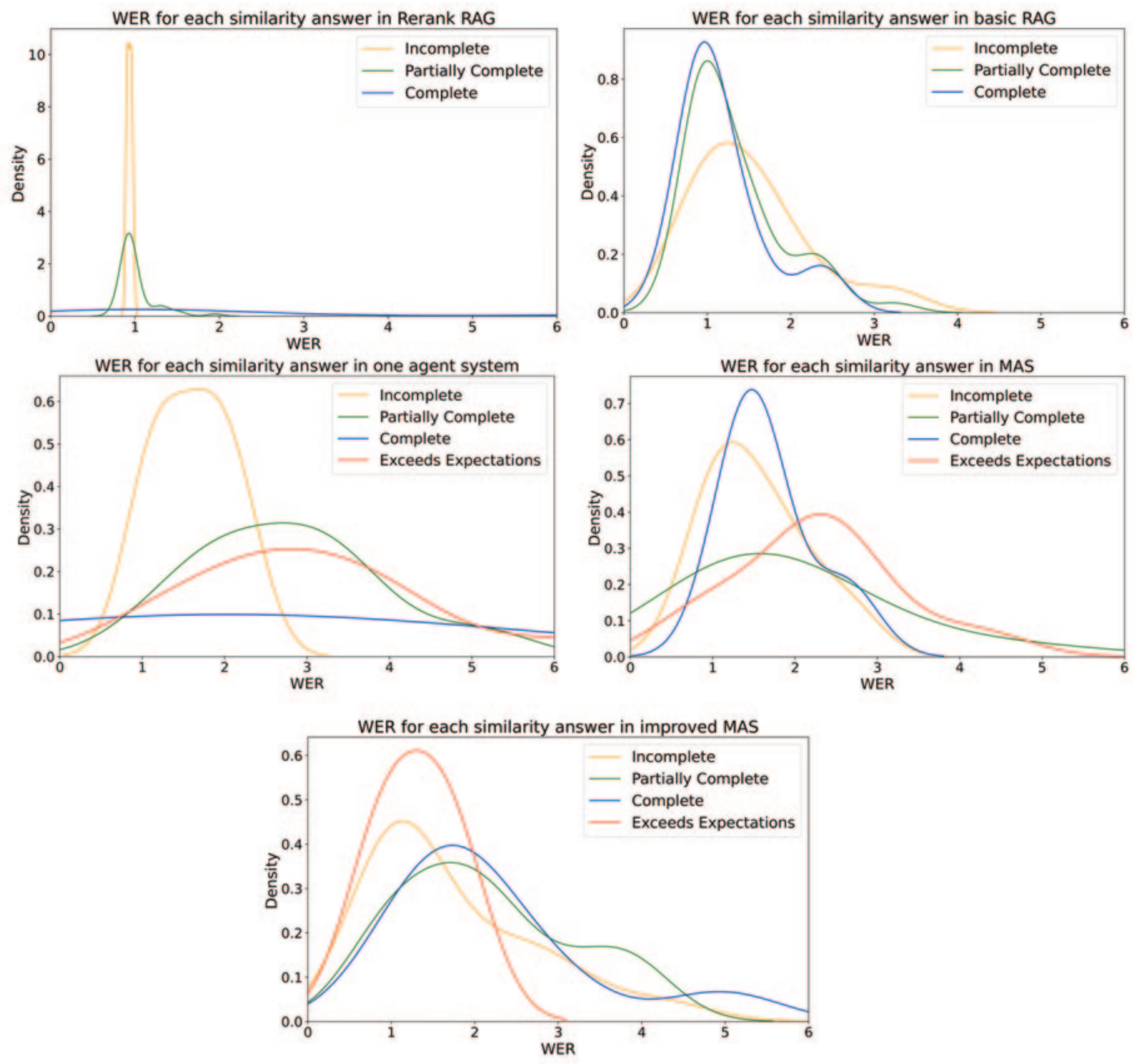
- In the same way, the one-agent system produces Incomplete responses with consistently low WER values. However, as semantic completeness improves (from Partially Complete to Complete and Exceeds Expectations), the WER rises notably. This pattern indicates that richer and more elaborate responses often diverge from the reference answers, suggesting the system struggles with consistent reasoning and structuring.

- In the initial MAS Complete and Exceeds Expectations responses have relatively higher WER compared to Partially Complete and especially Incomplete ones. The higher WER for more elaborate responses, apart from being probably caused by agent miscoordination, highlights again the fact that responses with greater detail contain more structural mismatches.

- The improved MAS exhibits the most balanced relationship between similarity and WER. Unlike the other systems, the Exceeds Expectations answers do not result in the highest WER. In fact, they tend to have lower WER values than Incomplete responses, indicating that the system is able to produce semantically rich and detailed answers while maintaining strong structural alignment. Despite the WER of Incomplete answers being lower than the Complete and Partially complete ones, prompt refinement proved to improve role clarity and coordination among agents.

These results reflect several of the failure modes identified in Cemri et al [18]. In the initial MAS, the high proportion of Partially Complete responses and the increase in WER for Complete and Exceeds Expectations answers point to issues related to agent miscoordination, such as reasoning-action mismatch or agents ignoring each other's input. These problems are consistent with the inter-agent misalignment failures described in the paper. The improved MAS, on the other hand, showed more consistent answers and better WER performance, which can be linked to clearer prompt design and better task specification. These changes helped reduce failures such as disobeyed role specifications and incorrect termination, which were commonly observed in other MAS setups in the study. While the paper highlights that prompt tuning is often not enough to resolve deeper system issues, the improvements observed here suggest that even small refinements in role clarity and communication can lead to more reliable agent behaviour and better overall results.

Figure 5. TER by similarity classification across all the systems.



Challenges for the organisations

IImplementing MAS in corporate environment is an interesting endeavour because such system could interact with data from within the organization autonomously, execute actions without supervision, and even be a method to manage data for clients or employees providing a better product or service. All these advantages translate to reducing operational costs and improve client satisfaction.

Nevertheless, this presents a range of business challenges that can significantly impact the success of such initiatives:

- **Complexity of integration with existing legacy systems:** Many corporations have established IT infrastructures that are deeply embedded in their operations, and integrating MAS with these systems can be technically demanding and costly.
- **Management of data privacy and security:** Multiagent systems often involve the exchange of large volumes of data between agents, which can include sensitive corporate information. Ensuring that this data is securely transmitted and stored is paramount to prevent breaches and maintain compliance with data protection regulations.
- **Scalability and Performance Optimization:** As the number of agents in a system increases, ensuring that the system remains scalable and performs efficiently can be challenging. Each agent requires computational resources, and the interactions between agents can become complex and resource intensive.
- **Skill and Knowledge Gaps: I**mplementing and managing multiagent systems requires specialized knowledge and skills that may not be readily available within the existing workforce. Businesses may face challenges in finding and retaining qualified personnel who are proficient in MAS technologies.
- **Coordination and Conflict Resolution:** In a multiagent system, agents often have to work together to achieve common goals. However, coordinating the actions of multiple autonomous agents can be difficult, especially when their objectives or strategies conflict.
- **Adherence to Industry-Specific Regulations and Standards.** Different industries are governed by various regulations and standards that dictate how data should be handled, processed, and stored. In Europe, the implementation of multiagent systems must also comply with the AI Act, a comprehensive regulation aimed at ensuring the safe and ethical deployment of AI systems. The AI Act imposes stringent requirements on high-risk AI systems, including rigorous testing, documentation, and transparency measures.

Conclusion

Large Language Models (LLMs) have revolutionized natural language processing by enhancing capabilities in reasoning, generation, and information retrieval. As these models expand, they are increasingly integrated into complex systems like Retrieval-Augmented Generation (RAG) and Multi-Agent Systems (MAS), involving multiple autonomous agents collaborating to solve problems and make decisions.

This newsletter explores the concept of a MAS through a simple Q&A application using these approaches. A basic RAG system has been complemented with an advanced RAG, but none of them achieved complete answers, giving rise to missing key points, despite efforts to improve relevance. The MAS system was more consistent, with more complete and excellent answers and fewer invalid ones, thanks to better teamwork among agents, better instructions and clearer roles for the agents.

Notwithstanding, despite these techniques improving the output of GenAI systems, several challenges are still ahead, such as the complexity for the integration with existing legacy systems, the security, data privacy, or technical elements, such as the scalability, the performance or the resolution of possible conflicts among agents, among others. These elements will have to be further addressed for a professional use of these new systems.

References

[1] H. Zhao et al., ‘Explainability for Large Language Models: A Survey’, ACM Trans Intell Syst Technol, vol. 15, no. 2, p. 20:1-20:38, Feb. 2024, doi: 10.1145/3639372.

[2] I. ILIN, ‘Advanced RAG Techniques: an Illustrated Overview’, Medium. Accessed: Feb. 18, 2025. [Online]. Available: https://pub.towardsai.net/advanced-rag-techniques-an-illustrated-overview-04d193d8fec6

[3] RobBagby, ‘Develop a RAG Solution - Generate Embeddings Phase - Azure Architecture Center’. Accessed: Feb. 21, 2025. [Online]. Available: https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/rag/rag-generate-embeddings

[4] MyScale, ‘Naive RAG vs. Advanced RAG’, Medium. Accessed: Mar. 12, 2025. [Online]. Available: https://medium.com/@myscale/naive-rag-vs-advanced-rag-17b38cda44c1

[5] ‘Rerank - Optimize Your Search With One Line of Code’, Cohere. Accessed: Mar. 12, 2025. [Online]. Available: https://cohere.com/rerank

[6] L. Wang et al., ‘A Survey on Large Language Model based Autonomous Agents’, Front. Comput. Sci., vol. 18, no. 6, p. 186345, Dec. 2024, doi: 10.1007/s11704-024-40231-1.

[7] Y. Cheng et al., ‘Exploring Large Language Model based Intelligent Agents: Definitions, Methods, and Prospects’, Jan. 07, 2024, arXiv: arXiv:2401.03428. doi: 10.48550/arXiv.2401.03428.

[8] T. Guo et al., ‘Large Language Model based Multi-Agents: A Survey of Progress and Challenges’, Apr. 19, 2024, arXiv: arXiv:2402.01680. doi: 10.48550/arXiv.2402.01680.

[9] H. Chen, ‘Understand the LLM Agent Orchestration’, SciSharp STACK. Accessed: Feb. 21, 2025. [Online]. Available: https://medium.com/scisharp/understand-the-llm-agent-orchestration-043ebfaead1f

[10] Mastering AI Agents. A comprehensive guite to evaluating AI agents. Galileo.

[11] ‘Qdrant - Vector Database’. Accessed: Mar. 17, 2025. [Online]. Available: https://qdrant.tech/

[12] ‘sentence-transformers/all-MiniLM-L6-v2 · Hugging Face’. Accessed: Mar. 17, 2025. [Online]. Available: https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[13] ‘What is Amazon Bedrock? - Amazon Bedrock’. Accessed: Mar. 17, 2025. [Online]. Available: https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html

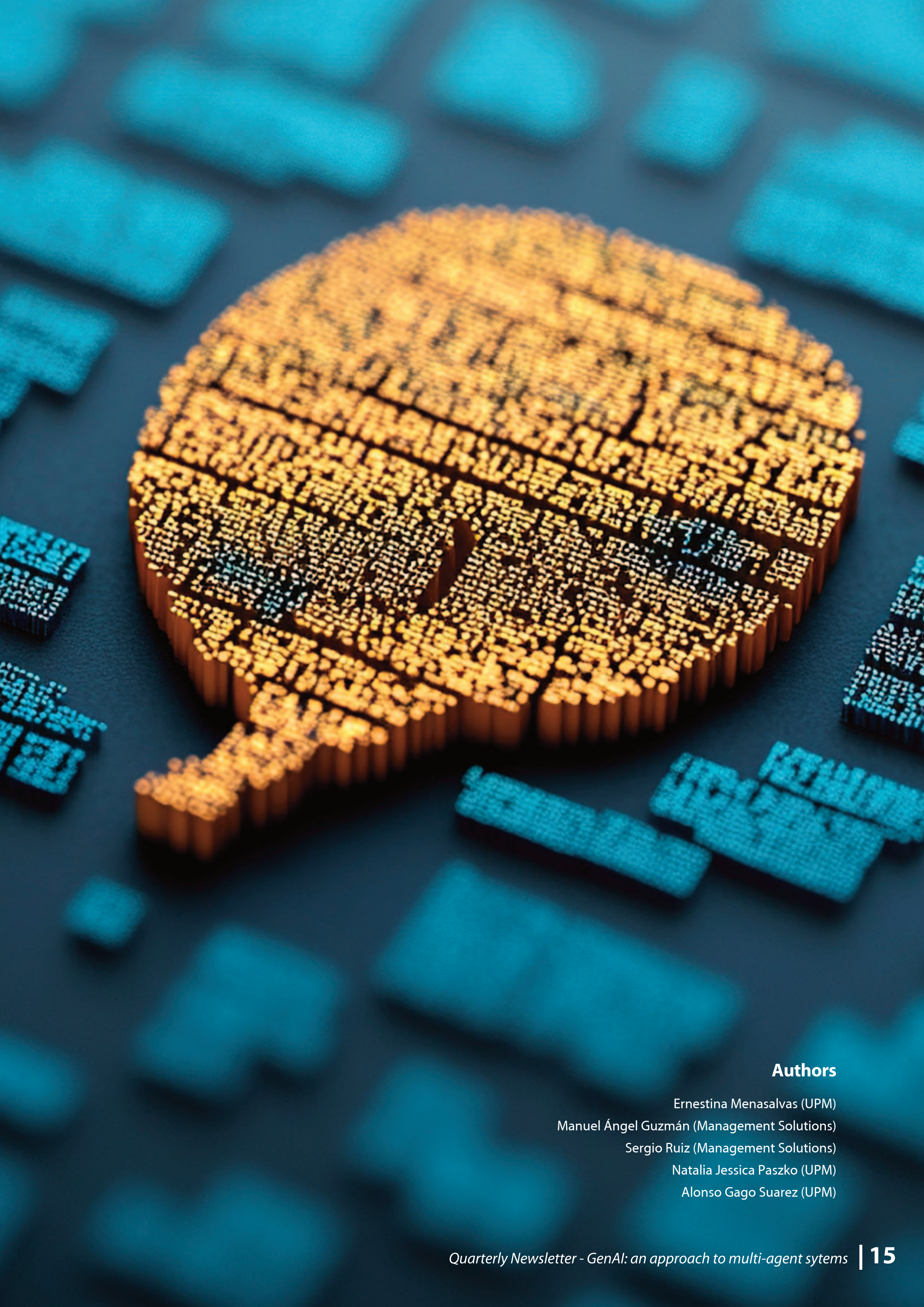
[14] ‘Claude 3 Haiku: our fastest model yet’. Accessed: Mar. 17, 2025. [Online]. Available: https://www.anthropic.com/news/claude-3-haiku

[15] ‘Improve the relevance of query responses with a reranker model in Amazon Bedrock - Amazon Bedrock’. Accessed: Mar. 17, 2025. [Online]. Available: https://docs.aws.amazon.com/bedrock/latest/userguide/rerank.html

[16] ‘Amazon Bedrock | AutoGen 0.2’. Accessed: Mar. 17, 2025. [Online]. Available: https://microsoft.github.io/autogen/0.2/docs/topics/non-openai-models/cloud-bedrock/

[17] R. Massawe, ‘Understanding and Calculating Word Error Rate (WER) in Automatic Speech Recognition using python’, Medium. Accessed: Mar. 17, 2025. [Online]. Available: https://medium.com/@ramadhanimassawe14/understanding-and-calculating-word-error-rate-wer-in-automatic-speech-recognition-using-python-661f18b518a5

[18] M. Cemri et al., ‘Why Do Multi-Agent LLM Systems Fail?’, Mar. 17, 2025, arXiv: arXiv:2503.13657. doi: 10.48550/arXiv.2503.13657.



Authors

Ernestina Menasalvas (UPM)

Manuel Ángel Guzmán (Management Solutions)

Sergio Ruiz (Management Solutions)

Natalia Jessica Paszko (UPM)

Alonso Gago Suarez (UPM)

POLITÉCNICA UNIVERSIDAD POLITÉCNICA DE MADRID

Management Solutions
Making things happen

The Universidad Politécnica de Madrid is a multi-sector and multi-disciplinary Public Law Entity, which carries out teaching, research and scientific and technological development activities.

www.upm.es

Management Solutions is an international consulting firm, focused on business, finance, risk, organization, technology and process consulting, operating in more than 50 countries and with a team of 4,000 professionals working for more than 2,000 clients worldwide.

www.managementsolutions.com

For more information visit
blogs.upm.es/catedra-idanae/