COMPARATIVE ANALYSIS OF SINGLE AND MULTI-AGENT LARGE LANGUAGE MODEL ARCHITECTURES FOR DOMAIN-SPECIFIC TASKS IN WELL CONSTRUCTION

Vitor Brandão Sabbagh

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Geraldo Bonorino Xexéo

Rio de Janeiro
Julho de 2025

COMPARATIVE ANALYSIS OF SINGLE AND MULTI-AGENT LARGE LANGUAGE MODEL ARCHITECTURES FOR DOMAIN-SPECIFIC TASKS IN WELL CONSTRUCTION

Vitor Brandão Sabbagh

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Geraldo Bonorino Xexéo

Aprovada por: Prof. Nome do Primeiro Examinador Sobrenome
Prof. Nome do Segundo Examinador Sobrenome
Prof. Nome do Terceiro Examinador Sobrenome
Prof. Nome do Quarto Examinador Sobrenome
Prof. Nome do Quinto Examinador Sobrenome

RIO DE JANEIRO, RJ – BRASIL
JULHO DE 2025

Brandão Sabbagh, Vitor

Comparative Analysis of Single and Multi-Agent Large Language Model Architectures for Domain-Specific Tasks in Well Construction/Vitor Brandão Sabbagh. – Rio de Janeiro: UFRJ/COPPE, 2025.

XII, 64 p.: il.; 29, 7cm.

Orientador: Geraldo Bonorino Xexéo

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2025.

Referências Bibliográficas: p. 47 – 52.

1. Large Language Models. 2. Agents. 3. Oil Well Construction. I. Bonorino Xexéo, Geraldo. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

To Carolina, my life partner.

Acknowledgements

I would like to thank everyone. [family, friends, etc]

I extend my gratitude to the well construction engineering experts, Marcelo Grimberg, Rafael Peralta, and Lorenzo Simonassi, whose expertise and dedication significantly contributed to this research.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

COMPARATIVE ANALYSIS OF SINGLE AND MULTI-AGENT LARGE LANGUAGE MODEL ARCHITECTURES FOR DOMAIN-SPECIFIC TASKS IN WELL CONSTRUCTION

Vitor Brandão Sabbagh

Julho/2025

Orientador: Geraldo Bonorino Xexéo

Programa: Engenharia de Sistemas e Computação

Apresenta-se, nesta tese, a aplicação de grandes modelos de linguagem (LLM) no setor de petróleo e gás, especificamente em tarefas de construção e manutenção de poços. O estudo avalia o desempenho de uma arquitetura baseada em LLM de agente único e de múltiplos agentes no processamento de diferentes tarefas, oferecendo uma perspectiva comparativa sobre sua precisão e as implicações de custo de sua implementação. Os resultados indicam que sistemas multiagentes oferecem desempenho melhorado em tarefas de perguntas e respostas, com uma medida de veracidade 28% maior do que os sistemas de agente único, mas a um custo financeiro mais alto. Especificamente, a arquitetura multiagente incorre em custos que são, em média, 3,7 vezes maiores do que os da configuração de agente único, devido ao aumento do número de tokens processados. Por outro lado, os sistemas de agente único se destacam em tarefas de texto para SQL (Linguagem de Consulta Estruturada), especialmente ao usar o Transformador Pré-Treinado Generativo 4 (GPT-4), alcançando uma pontuação 15% maior em comparação com as configurações multiagentes, sugerindo que arquiteturas mais simples podem, às vezes, superar a complexidade. A novidade deste trabalho reside em seu exame original dos desafios específicos apresentados pelos dados complexos, técnicos e não estruturados inerentes às operações de construção de poços, contribuindo para o planejamento estratégico da adoção de aplicações de IA generativa, fornecendo uma base para otimizar soluções contra parâmetros econômicos e tecnológicos.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

COMPARATIVE ANALYSIS OF SINGLE AND MULTI-AGENT LARGE LANGUAGE MODEL ARCHITECTURES FOR DOMAIN-SPECIFIC TASKS IN WELL CONSTRUCTION

Vitor Brandão Sabbagh

July/2025

Advisor: Geraldo Bonorino Xexéo

Department: Systems Engineering and Computer Science

This article explores the application of large language models (LLM) in the oil and gas sector, specifically within well construction and maintenance tasks. The study evaluates the performances of a single-agent and a multi-agent LLM-based architecture in processing different tasks, offering a comparative perspective on their accuracy and the cost implications of their implementation. The results indicate that multi-agent systems offer improved performance in question and answer tasks, with a truthfulness measure 28% higher than single-agent systems, but at a higher financial cost. Specifically, the multi-agent architecture incurs costs that are, on average, 3.7 times higher than those of the single-agent setup due to the increased number of tokens processed. Conversely, single-agent systems excel in text-to-SQL (Structured Query Language) tasks, particularly when using Generative Pre-Trained Transformer 4 (GPT-4), achieving a 15% higher score compared to multi-agent configurations, suggesting that simpler architectures can sometimes outpace complexity. The novelty of this work lies in its original examination of the specific challenges presented by the complex, technical, unstructured data inherent in well construction operations, contributing to strategic planning for adopting generative AI applications, providing a basis for optimizing solutions against economic and technological parameters.

Contents

List of Figures **x**

List of Tables **xii**

1 Introduction **1**
1.1 Objectives . . . 3
1.2 Business Scope Delimitation . . . 4
1.3 Thesis Structure . . . 4

2 Literature Review **5**
2.1 AI in the Exploration and Production (E&P) industry . . . 5
2.2 Large Language Models . . . 6
2.3 Q&A tasks . . . 6
2.4 Text-to-SQL tasks . . . 7
2.5 LLM-based applications . . . 7
2.5.1 Retrieval-Augmented Generation (RAG) . . . 7
2.5.2 Intelligent Agents . . . 8
2.5.3 Multi-Agent Setup . . . 8
2.5.4 LLM-as-judge . . . 9

3 Experimento 1 **11**
3.1 Methodology . . . 11
3.1.1 Data Preparation. . . . 11
3.1.2 Multi-Agent Architecture. . . . 16
3.1.3 Evaluation. . . . 16
3.2 Results . . . 17
3.2.1 Truthfulness. . . . 18
3.2.2 Performance. . . . 19
3.2.3 LLM Cost . . . 19
3.3 Discussion . . . 20
3.3.1 General Performance. . . . 20
3.3.2 Cost-Performance Analysis. . . . 20

3.3.3 Model Performance Variations. 21
3.3.4 Economic Efficiency. 21
3.3.5 Challenges and Limitations. 22
3.3.6 Practical Implications. 24
3.3.7 Future Directions. 25

4 Experiment 2 **27**
4.1 Methodology 28
4.1.1 Experimental Workflow 28
4.1.2 Data Sources 30
4.1.3 System Architecture 31
4.1.4 Experimental Setups 33
4.1.5 Execution Details 35
4.1.6 Evaluation Metrics 36
4.1.7 Limitations 37
4.1.8 Summary 37
4.1.9 Quality Assessment with LLM-as-a-Judge 38
4.2 Results and Discussion 39
4.2.1 Performance 39

5 Conclusões 1 **44**

6 Conclusões 2 AAA **46**

References **47**

A Experiment 2 **53**
A.1 Code for LLM-as-a-Judge 53
A.2 Results 56
A.2.1 Precision 56
A.2.2 Recall 63

List of Figures

1.1 Sample of drilling & completion learned lesson partial document. (translated from Portuguese) 2

3.1 Schematic of the LLM-based agent interacting with an environment containing tools for task-specific operations, and the Human Agent interface for user interaction and feedback. 13
3.2 Chat setup with one User Proxy WU *et al.* (2023) and one Assistant. 13
3.3 Decision process of the agent. 14
3.4 Chat setup with one Chat Manager and a group of LLM agents. . . . 16
3.5 Multi-agent decision process. 16
3.6 Truthfulness and standard deviation in Q&A tasks by LLM model and agent configuration. 18
3.7 Truthfulness and standard deviation in Text-to-SQL tasks by LLM model and agent configuration. 18
3.8 Performance and standard deviation in Q&A tasks by LLM model and agent configuration. 19
3.9 Performance and standard deviation in Text-to-SQL tasks by LLM model and agent configuration. 19
3.10 Average LLM costs and Truthfulness per completed task according to setup and model. 20

4.1 Linear-Flow architecture. PT1 indicates Prompt for Tool 1 and so on. 33
4.2 Linear-Flow with Router architecture. 34
4.3 Single-Agent architecture 35
4.4 Multi-Agent setup with one supervisor and 4 specialist agents. 35
4.5 F1 score by model and configuration. 40
4.6 Average F1 score by configuration. 40
4.7 Average F1 score by model. 40
4.8 F1 Score distribution by model and configuration of agents 41

4.9 Best F1 score by question index and configuration. 41
4.10 Best F1 score by question index and model. 42
4.11 Precisão por modelo e configuração. 42
4.12 Precisão por pergunta e configuração. 42
4.13 Recall by model and configuration. 43
4.14 Recall por pergunta e configuração. 43

A.1 Best precision by model and configuration. 56
A.2 Best precision by question index and configuration. 56
A.3 Best precision by question index and model. 57
A.4 Facet histogram of precision by model. 57
A.5 Facet histogram of precision by model (best of 3). 58
A.6 Histogram of all precisions. 58
A.7 Line plot of precision by question index and model. 59
A.8 Boxplot of precision by model and configuration. 60
A.9 Precision by model. 60
A.10 Precision by model and configuration. 61
A.11 Line plot of precision by question index and configuration. 61
A.12 Scatter plot of precision vs. total time. 62
A.13 Scatter plot of precision vs. total token count input. 62
A.14 Swarm plot of precision by model and configuration. 63
A.15 Violin plot of precision by model and configuration. 63

List of Tables

3.1 Query examples used in this study. 12
3.2 Query sample with inputs, outputs, and evaluations. 15
3.3 Results on Q&A and Text-to-SQL tasks, including standard deviation (Std). The best metrics are highlighted with **bold and underline**. The second best are highlighted with **bold**. 17

Chapter 1

Introduction

In the dynamic changing of the oil and gas (O&G) industry, digital transformation has emerged as a key element to achieve operational efficiency, sustainability, and competitiveness. At the forefront of this transformation are Large Language Models (LLMs), which have the potential to process unstructured queries, map out alternatives, and advise users on possible actions KAR and VARSHA (2023). We also note the advantage of increased engagement, cooperation, accessibility, and ultimately profitability. These models redefine paradigms in knowledge management and information retrieval and impact a variety of other areas ECKROTH and GIPSON (2023), making it crucial to adopt these technologies to remain competitive.

A study conducted by DELLACQUA *et al.* (2023), in collaboration with the Boston Consulting Group demonstrates that in knowledge-intensive tasks, consultants equipped with access to LLMs like GPT-4 not only completed tasks more efficiently (25.1% more quickly on average) but also with substantially higher quality, achieving results more than 40% better compared to those without AI assistance DELLACQUA *et al.* (2023). Increase in productivity of knowledge workers was 12% on average. A major oil company spent in 2023 $2.8B with employee compensation. A potential increase of 12% in knowledge workers productivity, given they represent 60% of all employee, could represent $204M annual savings in this scenario. Broader economic indicators predict significant transformations due to generative AI (Gen-AI) across various industries. A report from Goldman Sachs HATZIUS *et al.* (2023) highlights that Gen-AI is poised to increase global GDP by nearly 7%, increasing productivity growth by 1.5 percentage points over the next decade. This economic uplift is expected due to AI's ability to automate complex workflows and create new business opportunities, significantly impacting employment and productivity sectors worldwide.

Expanding on the broader discussion on data utilization within organizations, an important issue is the challenge of extracting relevant information from extensive databases SINGH *et al.* (2023). Initially, the challenge of knowing, finding, and ac-

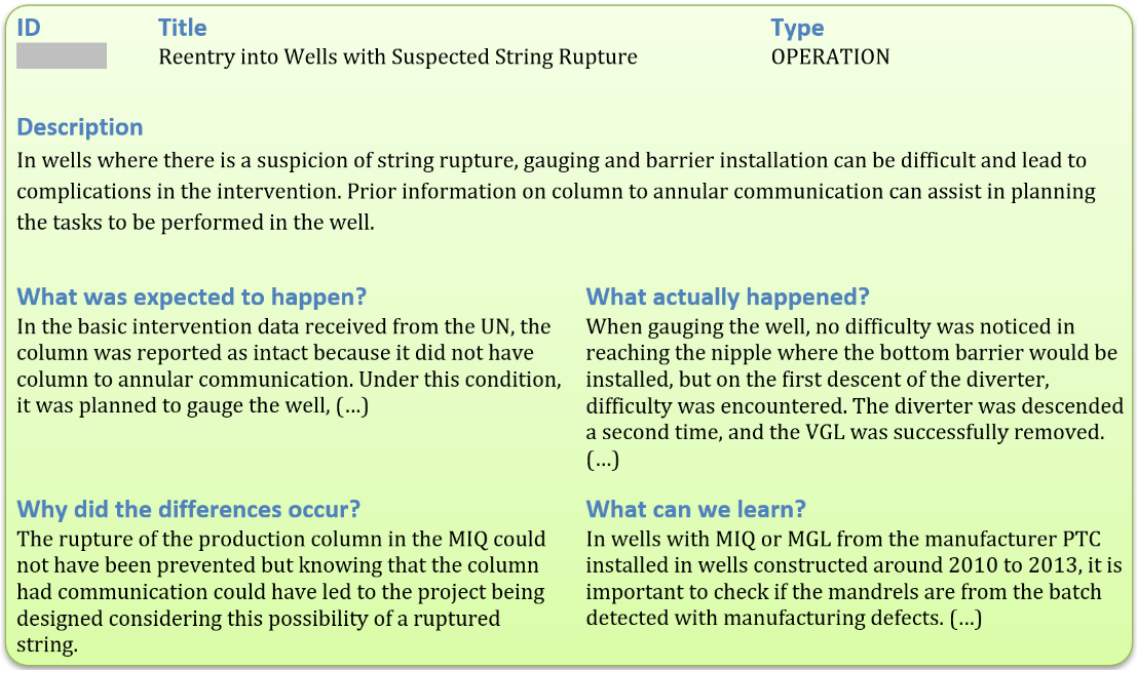

Figure 1.1: Sample of drilling & completion learned lesson partial document. (translated from Portuguese)

cessing data poses a significant obstacle to decision-making processes. Collaborators at O&G companies often face the intensive task of manually searching large data repositories to find useful information.

Focusing specifically on the activities of drilling and completion of offshore and onshore wells, a major challenge lies in the inherently complex and technical nature of the data involved, which can be from various types: operations, projects, technologies, supply chains, and others. Inefficiency in leveraging large volumes of unstructured data worsens these challenges, as observed by SINGH *et al.* (2023). A significant amount of the data generated and collected in this sector is unstructured, ranging from text reports and emails to images and videos of exploration and production activities. Examples include hundreds of daily operational reports from drilling rigs, well execution projects, nonproductive time (NPT) reports, and operational lessons learned documents, as illustrated in Figure 1.1. As a result, valuable information can remain untapped, and the potential to find insights, informed decision-making and innovation is significantly compromised. SINGH *et al.* (2023) showcases the capabilities and potential of Generative AI-enabled chatbots for the O&G sector, particularly in enhancing drilling and production analytics to achieve better business outcomes. The author concludes that companies that adopt these technologies in the coming years will see clear advantages.

However, the deployment of such technologies presents limitations and introduces challenges, including biased data, hallucinations, lack of explainability, and

logical reasoning errors, among others HADI *et al.* (2023), which require a balanced approach to harness their potential in a responsible manner. Although previous research has focused mainly on the broader applications of AI in industry, the novelty of our research lies in its original examination of the specific challenges and solutions presented by the complex, technical and unstructured data inherent in O&G operations. By comparing single- and multi-agent systems, this study fills a knowledge gap, providing empirical insights into the effectiveness of different Gen-AI architectures in a domain where such studies are scarce.

The adoption of these technologies by a major oil company underscores their potential to revolutionize data analysis and management, presenting an opportunity for deeper exploration and application.

1.1 Objectives

This research directly addresses challenges faced by major oil companies. By investigating the comparative advantages and limitations of various Gen-AI architectures, including single and multi-agent systems, for Q&A and Text-to-SQL tasks, this study aims to identify the most efficient and cost-effective solutions. The specific objectives of this research are to assess the suitability and effectiveness of multi-agent systems based on LLMs for complex, domain-specific tasks in well engineering, aiming to streamline information access and decision-making. The study will compare single-agent and multi-agent AI systems in terms of their ability to address well engineering queries. Finally, it will map the potential obstacles and limitations associated with deploying Gen-AI applications.

The insights gained from this research will directly contribute to O&G companies strategic goals by improving access to well engineering information and automated data analysis tasks. A comprehensive understanding of the challenges and limitations associated with Gen-AI will enable informed decisions about its adoption, maximizing the return on investment.

To achieve these objectives, this research was conducted through two distinct experimental phases. The first, carried out in 2024, focused on a foundational comparison between single and multi-agent architectures, revealing key insights into their performance, cost, and limitations such as hallucination and context interpretation. The rapid evolution of generative AI frameworks and models prompted a second, more advanced experiment in 2025. This second phase built upon the initial findings, also employing non-agentic workflows as baseline and a more rigorous, quantitative evaluation methodology to address the challenges identified in the first experiment and automated evaluation based on the concept commonly reffered to as "LLM-as-judge" (GU *et al.* (2025)).

1.2 Business Scope Delimitation

To contextualize the scope of this study, it is necessary to understand the life cycle of an oil field, which begins with Exploration and progresses to the Development of Production, followed by effective production, and culminates in decommissioning BADIRU and OSISANYA (2016). Gen-AI has the potential to impact each of these phases, but the focus of this work lies in the operations of the development and maintenance stages.

Well construction is a highly specialized activity involving drilling and completion of wells for hydrocarbon extraction THOMAS (2004). In this context, Gen-AI can be applied in various ways. For example, a chatbot could manage knowledge by answering queries about operations and well projects by retrieving information from the organization's databases. Additionally, LLM-based agents could be used in executive project review to ensure that drilling or completion operations comply with the organization's standards and adhere to best operational practices. Moreover, Gen-AI could perform inference in unstructured databases to extract specific information from text reports and obtain structured data.

This business scope emphasizes the importance of Gen-AI in the construction and maintenance of wells.

1.3 Thesis Structure

****SERÁ FEITO POR ÚLTIMO****

Chapter 2

Literature Review

2.1 AI in the Exploration and Production (E&P) industry

The use of AI in the Exploration and Production (E&P) industry has been extensive. In the last decades the majority of AI applications in the industry involve data mining and neural networks BRAVO *et al.* (2014). One example is the work by GUDALA *et al.* (2021) on the optimization of heavy oil flow properties, through the use of a neural networks to optimize flow-influencing parameters. Another development was a deep learning workflow proposed by GOHARI *et al.* (2024), with the generation of synthetic graphic well logs through the application of transfer learning. These developments illustrate the potential of AI in improving processes and the accuracy and efficiency of data analysis RAHMANI *et al.* (2021).

Natural Language Processing (NLP) stands at the intersection of computer science and linguistics, representing a domain within artificial intelligence aimed at enabling computers to understand and process human language in a way that is both meaningful and effective LIDDY (2001). This field integrates a diverse range of computational techniques to analyze and represent text at various levels of linguistic detail, striving to emulate human-like language understanding. As an active area of research, traditionally NLP employs multiple layers of language analysis, each contributing uniquely to the interpretation and generation of language, which finds practical applications across various sectors LIDDY (2001). In the O&G industry, the management of unstructured data such as texts, images, and documents is crucial, with Natural Language Processing (NLP) and Machine Learning playing key roles. Research by ANTONIAK *et al.* (2016) and CASTIÑEIRA *et al.* (2018) has explored the use of NLP to analyze risks and drilling reports.

2.2 Large Language Models

Large Language Models (LLMs) are advanced neural network-based models designed to understand and generate human-like text. They leverage the Transformer architecture introduced in the seminal paper "Attention is All You Need" by VASWANI *et al.* (2017). This architecture relies on self-attention mechanisms, allowing the model to weigh the importance of different words in a sentence effectively.

The emergence of LLMs has made it possible to comprehend and produce textual information. These systems are expected to revolutionize various industries by supporting complex decision-making processes. GPT models OPENAI *et al.* (2023), in particular, leverages its vast training data to provide human-like responses MOSSER *et al.* (2024), which can be highly beneficial in contexts requiring natural language understanding and generation.

As highlighted by SINGH *et al.* (2023), the integration of LLM-based solutions, such as conversational chatbots, offers an approach to optimizing operations across various business segments, including drilling, completion, and production. SINGH *et al.* (2023) uses LLMs models to extract, analyze, and interpret datasets, enabling generation of insights and recommendations.

Despite its widespread impact, language models are not without its limitations. In many industry-specific applications, the critical information required is often proprietary, not shared with third parties, and thus absent from the training data of these LLMs MOSSER *et al.* (2024). This gap means that GPT models might not have access to the most up-to-date or sensitive information needed for certain tasks. Moreover, due to their probabilistic nature, LLMs can experience hallucinations, producing confident yet incorrect or nonsensical responses based on user input OPENAI *et al.* (2023).

2.3 Q&A tasks

Question and Answer (Q&A) represent a method for facilitating knowledge transfer between individuals within organizations (ISKE and BOERSMA, 2005). Conceptually, Q&A systems are designed to connect individuals who possess specific knowledge with those seeking that knowledge through a structured question-and-answer format. The role of Q&A in the documentation landscape, as exemplified by platforms like Stack Overflow, highlights their significance in technical disciplines (TREUDE *et al.*, 2011). This understanding can guide organizations in making more informed decisions about implementing such systems to enhance knowledge transfer and organizational learning (ISKE and BOERSMA, 2005).

2.4 Text-to-SQL tasks

Text-to-SQL tasks in the context of artificial intelligence involve the automatic translation of natural language questions or commands into structured SQL (Structured Query Language) queries (QIN *et al.*, 2022). This is an important area in natural language processing (NLP), allowing users to interact with databases using plain language rather than needing to know how to write complex SQL queries.

The arrival of advanced language models like GPT-3 and GPT-4 (OPENAI, 2023) has marked a significant leap in Text-to-SQL applications (SINGH *et al.*, 2023), demonstrating remarkable capabilities in handling these tasks. This can be attributed to their extensive training on diverse datasets (DENG *et al.*, 2021), which include not only large amounts of text but also structured data like tables and code, enabling the model to understand the intricate relationships between language and data structures. The study by (DENG *et al.*, 2023) introduces a pre-training framework for text to SQL translation, emphasizing the alignment between text and tables in Text-to-SQL tasks.

2.5 LLM-based applications

2.5.1 Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) technique combines LLMs with information retrieval to generate accurate and up-to-date responses, as introduced by LEWIS *et al.* (2020). It employs a search in a database to find relevant information, overcoming the inherent limitations of LLMs that rely solely on the prior knowledge embedded in the language model during the training phase. With the ongoing evolution of information retrieval, which has moved from term-based methods to more semantic approaches leveraging deep learning and large datasets to tackle more complex challenges.

As elucidated by LEWIS *et al.* (2020), RAG unites the strengths of pre-trained parametric and non-parametric memory, using a dense vector index and a semantic retriever. As demonstrated by LI *et al.* (2022) in their analysis, RAG is surpassing traditional generative models in terms of performance across a variety of tasks. The study provides a detailed survey on this topic, emphasizing the fundamental concepts and its applicability in specific contexts.

New tools have been developed to facilitate the implementation of RAG solutions. LIU *et al.* (2023) present a toolkit that integrates augmented retrieval techniques into LLMs, including modules for query rewriting, document retrieval, passage extraction, response generation, and fact-checking, enabling the creation

of more factual and specific responses. The recent study by ZHAO *et al.* (2023) extends this horizon by examining the incorporation of multimodal knowledge into generative models, exploring the integration of diverse external sources such as images, code, tables, graphs, and audio, to enhance the grounding context and improve usability. It also explores potential future trajectories in this emerging field, marking a relevant contribution to the evolving narrative of RAG and its applications.

2.5.2 Intelligent Agents

According to RUSSELL (2020), an agent is something that performs actions. When it comes to computerized agents (in our case, AI-based), these agents are expected to do more: operate autonomously, perceive the environment, persist over time, adapt to changes, create, and strive to achieve goals. The agent program implements the agent function. There is a variety of basic agent program designs that vary in efficiency, compactness, and flexibility. The appropriate design of the agent program depends on the nature of the environment. In this work, a goal-based agent design was implemented, which acts to achieve defined goals (RUSSELL, 2020). Other possible types include simple reflex agents, which directly respond to perceptions, while model-based reflex agents maintain an internal state to track aspects of the world that are not evident in the current perception. Finally, there are utility-based agents, which try to maximize their expected "happiness" (RUSSELL, 2020).

2.5.3 Multi-Agent Setup

As demonstrated by XI *et al.* (2023), the pursuit of Artificial General Intelligence (AGI) has significantly benefited from the development of LLM-based agents, capable of sensing, decision-making, and acting across diverse scenarios. His study outline a foundational framework for such agents, consisting of brain, perception, and action components, which can be customized for various applications including single-agent scenarios, multi-agent systems, and human-agent collaboration . The comprehensive survey underscores the crucial role of LLMs in moving towards AGI, suggesting a promising horizon for operational efficiency and decision-making processes in complex organizational settings (XI *et al.*, 2023).

LI *et al.* (2024a) demonstrated that, through a sampling and voting method, the performance of LLMs scales with the number of instantiated agents. Another open-source framework is AutoGen (WU *et al.*, 2023), that enables the creation of LLM multi-agent applications, allowing for customization across various modes including. It supports diverse applications in fields such as mathematics, coding, and operations research, demonstrating its effectiveness through empirical studies (WU *et al.*, 2023).

2.5.4 LLM-as-judge

The LLM-as-Judge paradigm represents a significant shift in the evaluation of NLP systems in general, utilizing a language model as a scalable proxy for human evaluators (LI *et al.* (2024b)). This approach was developed to overcome the semantic shallowness of traditional metrics like BLEU or ROUGE and the logistical challenges of extensive human annotation (ZHENG *et al.* (2023)). By providing a "judge" LLM with a clear rubric and context, it can perform nuanced assessments of qualities like coherence, relevance, and factual accuracy (LI *et al.* (2024b)). This method has proven effective for complex, open-ended tasks where simple string matching is insufficient, with models like GPT-4 demonstrating over 80% agreement with human preferences in benchmarking studies (ZHENG *et al.* (2023)).

For evaluating Retrieval-Augmented Generation (RAG) systems, the LLM-as-Judge framework can be adapted to produce structured, quantitative assessments. In this application, the judge LLM is tasked with comparing the RAG-generated answer against a ground-truth dataset. By leveraging a meticulously crafted prompt that defines the classification criteria, the judge can systematically categorize each output into classes such as True Positive (TP) (factually consistent with the ground truth), False Positive (FP) (introduces unsupported information), True Negative (TN) (a correct refusal to answer), or False Negative (FN) (missing relevant information). This structured approach moves beyond subjective scoring towards a more objective, task-specific evaluation, similar to methodologies that use specialized judge models trained on fine-grained feedback for enhanced reliability (KIM *et al.* (2024)).

The primary advantage of this methodology is its ability to translate qualitative, AI-driven judgments directly into a confusion matrix. By aggregating the classifications across an entire evaluation dataset, it becomes possible to calculate standard, interpretable metrics such as precision (Equation 2.1), recall (Equation 2.2), and F1-score (Equation 2.3). This process establishes a robust and replicable pipeline for benchmarking the factual accuracy of a RAG system at scale. While it is important to acknowledge the potential for inherent biases in LLM judges (GU *et al.* (2025)), studies show high correlation with human-expert evaluations (LI *et al.* (2024b)), making it a powerful tool for iterative development and system comparison.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.2}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.3}$$

Chapter 3

Experimento 1

3.1 Methodology

This section describes the approach and tools employed to investigate the effectiveness of a language model-based agent in responding to specific queries within the domain of well construction and maintenance. Firstly, the preparation, selection, and utilization of the data sources are described, explaining how each contributes to the knowledge base from which the agent derives its responses.

3.1.1 Data Preparation.

This experiment was conducted in the well construction department of a major oil company. The choice of tasks was focused in technical knowledge management and data analysis. Examples of queries used in the experiment are listed in Table 3.1. The data sources for executing such tasks were chosen to cover a range of operational scenarios in the activity of well construction and maintenance: Operational Knowledge Items, Operational NPTs (Non-Productive Time), and a Collaborator Finder.

Operational Knowledge Items: during drilling, completion, and workover interventions, documents called Knowledge Items are written by specialists, as depicted in Fig 1.1, which can be of 4 types: Technical Alert, Learned Lesson, Good Practice, and Well Observation. This is a tool for knowledge management, considering the large number and variety of specialists involved and well operations performed.

Operational NPTs (Non-Productive Time): the second data source refers to data on anomalies that occurred during well interventions, containing information such as the title, description of the event, the well where it occurred, type of operation, responsible sector, involved rig, lost time in hours, and start and end dates of the event. These data are critical for the industry, as NPTs represent periods when

Task category	**Query example**
Q&A	How does the presence of silica in the composition of cement paste affect its thermal stability at high temperatures? What are the main challenges and risks associated with through tubing plug and abandonment in highly deviated wells? What can cause hydrate formation in the Tree Running Tool connector during the HCR (High Collapse Resistance) hose flush before connecting to the Wet Christmas Tree? What can cause the Down Hole Safety Valve to remain open due to hydrate formation in the control lines? What can cause damage to thread protectors and sealing areas of pin ends of pipes stored at the coating yard? What can cause high drag and torque off-bottom during the drilling of a well with high deviation? What precautions should be taken when performing a top check of the abandonment plug in wells with higher inclination? What are the critical factors to consider when choosing a base fluid for manufacturing a viscous support plug? What are the best practices for managing drilling parameters during cement cutting to avoid premature bit wear?
Text-to-SQL	What was the longest-lasting NPT on rig number 05? How many NPTs occurred on rig number 06 during August 2023?

Table 3.1: Query examples used in this study.

the operation of drilling, completion or maintenance is interrupted due to some technical or logistical problem. The identification and analysis of these events are essential for continuous process improvement, cost reduction, and increased operational efficiency. By understanding the causes and circumstances of these incidents, organizations can develop strategies to prevent them in the future, optimizing operation time.

Collaborator Finder: the third data source used in the experiment is a collaborator finder, an important tool within an organization for consulting and managing employee data. This system allows for quick search and identification of employees through information such as name, workplace, company, registration, and role. The importance of this tool for the experiment lies in the possibility of cross-referencing employee data with other information sources for a more complete response by the agent.

Each of these sources provides inputs for the agent to offer a more accurate and updated view of operations and organizational structure. A set of documents and records was randomly selected from each database, over which questions were formulated. For each document, up to 3 questions were generated, resulting in a

dataset of tasks of the Q&A and Text-to-SQL types. Some examples are described in Table 3.1.

In this work, a goal-based agent RUSSELL (2020) was implemented with the goal of accurately responding to various queries. The agent operates within an environment equipped with multiple tools for task-specific operations, as shown in Figure 3.1, and interfaces with users to receive queries.

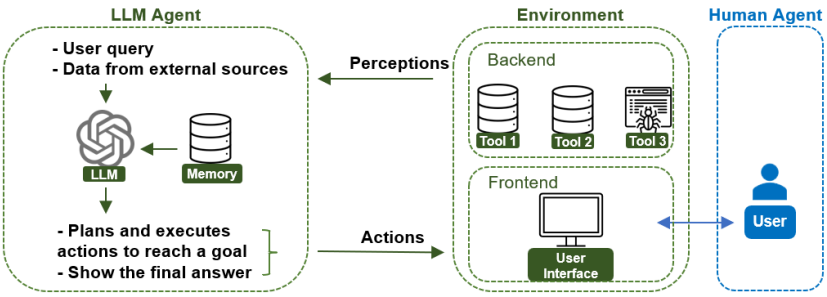

Figure 3.1: Schematic of the LLM-based agent interacting with an environment containing tools for task-specific operations, and the Human Agent interface for user interaction and feedback.

Initially, a configuration of agents was implemented as described in Figure 3.2 using AutoGen Framework WU *et al.* (2023) with an architecture that allows information retrieval and user interaction. This system consists of:

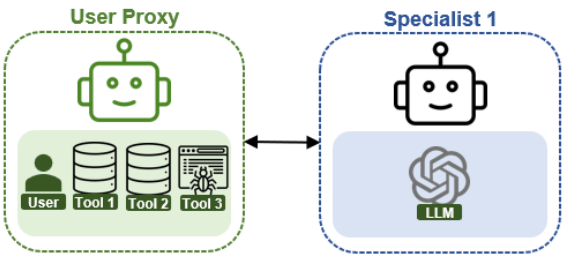

Figure 3.2: Chat setup with one User Proxy WU *et al.* (2023) and one Assistant.

- **User Proxy:** represents the interface with the user and with tools to access external databases. The modular nature of the tools allows the User Proxy to be customized and expanded based on the variety of data sources and the specific requirements of the application domain.
- **Agent:** powered by LLMs such as GPT-4 and GPT-3, is the analytical engine of the system. This agent interprets the queries received from the User Proxy and formulates responses.

For each question in the dataset, the agent's decision-making process is executed as described in Figure 3.3, initially selecting the appropriate tool to respond to a query and, finally, compiling the retrieved information to provide a final answer.

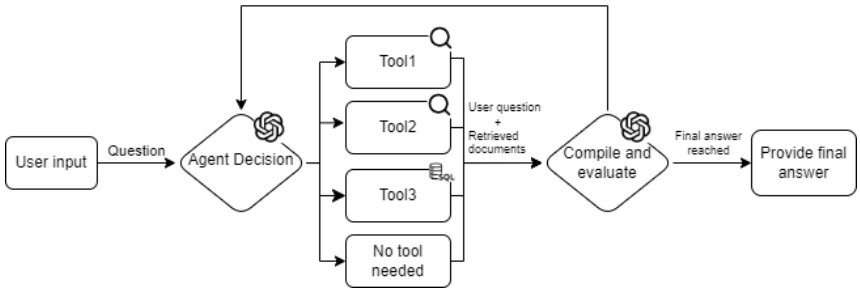

Figure 3.3: Decision process of the agent.

In this experiment, three tools were considered in the decision-making process:

- **Tool 1 - Knowledge Items Search:** a tool for searching for learned lessons that may be relevant to the query.
- **Tool 2 - Employee Search:** functionality that allows the search for information related to collaborators of an organization.
- **Tool 3 - NPT SQL Query:** interface for executing SQL queries on a database of operational NPTs.

In parallel, there is a pathway that allows the LLM Agent to provide a direct response, without the need to resort to other tools, presumably used when the agent already possesses the necessary information. Finally, the agent presents the final response to the user, which is the product of a language model processing, taking as inputs the user's query and relevant information retrieved and included in the context of the prompt.

The Table 3.2 provides a detailed analysis of the performance and accuracy of different models, specifically GPT-3.5-turbo and GPT-4, when queried about the impact of silica on the thermal stability of cement paste at high temperatures. The table compares both single-agent and multi-agent setups, evaluating the final outputs based on truthfulness, performance, and specialist comments. For each query, the table highlights the relevance and accuracy of the information provided by the models, including any extraneous or unrelated sections noted by the specialists. This comprehensive comparison allows for an in-depth assessment of the models' capabilities in generating accurate and relevant responses to technical questions.

Table 3.2: Query sample with inputs, outputs, and evaluations.

Input query	Model	Setup	Final output	Truth. fulness	Performance	Specialist Comments
How does the presence of silica in the composition of cement paste affect its thermal stability at high temperatures?	GPT-3.5-turbo	Single-Agent	The presence of silica in the composition of cement paste can affect its thermal stability at high temperatures due to some characteristics of silica. Silica is an inorganic material that has a stable crystalline structure at high temperatures. When exposed to high temperatures, the silica present in the cement paste can undergo structural transformations, resulting in changes in the properties of the paste. One of the main changes that can occur is the thermal expansion of the cement paste due to the formation of unstable silica compounds. This thermal expansion can lead to cracks and fissures in the structure, compromising the material's integrity and stability. Moreover, the presence of silica in the composition of cement paste can affect the material's thermal resistance. Silica has low thermal conductivity, which can lead to a reduced ability of the cement to withstand high temperatures. This can result in a decrease in the material's strength and, consequently, a reduction in its thermal stability. In summary, the presence of silica (...)	4	5	Created a section that is irrelevant to the question: "Moreover, the presence of silica in the composition of cement paste can affect the material's thermal resistance. (...)"
		Multi-Agent	The presence of silica in the composition of cement paste has a positive effect on thermal stability at high temperatures. Silica is added to cement to mitigate the retrogression of cement strength when exposed to temperatures above 230°F (110°C). Deep wells or those with a high geothermal gradient may require the use of silica cement to ensure the thermal stability of the paste. The use of silica cement is particularly important in surface casing cementing, where the temperature can exceed the critical limit. (...)	2	2	Retrieved an unrelated Lesson Learned on the subject (about plugs).
	GPT-4	Single-Agent	The presence of silica in the composition of cement paste increases thermal stability, especially at high temperatures. Silica is used to transform the cement into a refractory material, capable of withstanding extreme conditions. The cement (...). However, (...).	3	4	Although it is true, the item on the side is not relevant to the question: "However, it is important to note that (...)."
		Multi-Agent	Silica is added to cement at a ratio of 35% by weight of cement to mitigate the retrogression (loss of compressive strength) of the cement in the well when subjected to temperatures above 230ºF (110ºC). (...)	5	4	Cited an unrelated (but true) Lesson Learned about plugs, Lesson Learned LA7005.

3.1.2 Multi-Agent Architecture.

A second architecture that employs multiple agents was implemented, each having a distinct tool to engage with external data-sources, as illustrated in Figure 3.4. This architecture also starts with the user's input. However, as depicted in Figure 3.5, the subsequent 'Speaker selection' process determines the specialized agent evaluated as most suitable for answering the user's question.

When a query falls within the direct LLM knowledge, the pathway 'No tool needed' is selected, and the corresponding agent responds without further tool engagement. The selected agent then 'Compiles and evaluates' the gathered information in the context of the user's query, ensuring a response that is both accurate and context-aware. The final step, 'Provide final answer', is where the multi-agent system converges to deliver the final, coherent answer to the user.

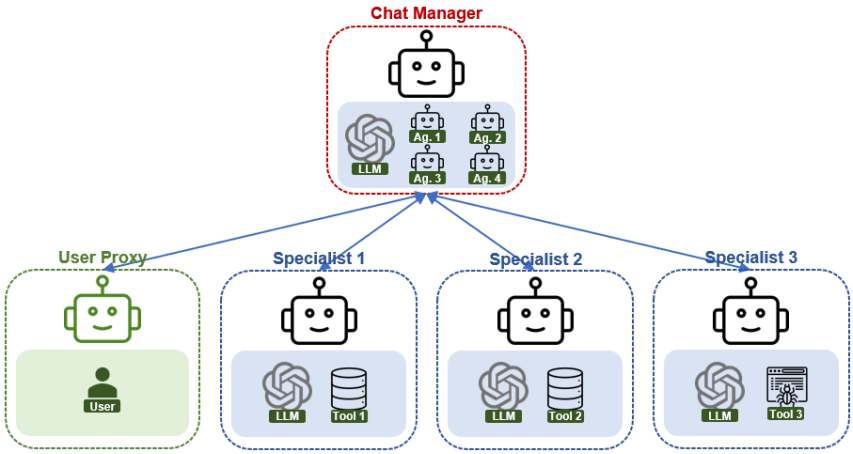

Figure 3.4: Chat setup with one Chat Manager and a group of LLM agents.

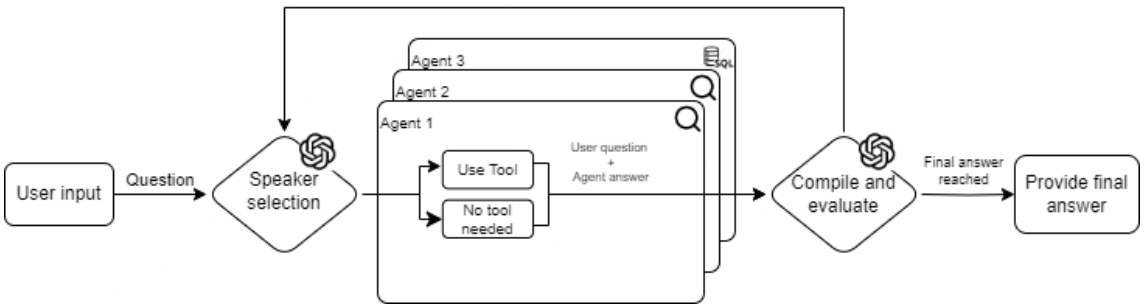

Figure 3.5: Multi-agent decision process.

3.1.3 Evaluation.

For the evaluation process, in line with the evaluation conducted by LI *et al.* (2023), a group of 3 specialist engineers analyzed 33 questions and their corresponding

answers for each configuration. The experts assessed each Q&A pair based on the following predefined metrics:

- **Truthfulness:** a metric to gauge the extent of divergence from factual accuracy.
- **Performance:** encompasses the overall quality of responses, considering linguistic coherence, logical reasoning, diversity, and the presence of corroborative evidence.

The final grade was determined by averaging the scores of the all entries for each configuration. This evaluation ensured a comprehensive assessment of the models' capabilities.

3.2 Results

This section provides an analysis of the data collected and answers the research questions. The results are presented in Table 3.3 and are organized according to the objectives of the study, with each objective being addressed in detail.

The third metric, LLM Cost, represents the financial cost associated with using OpenAI's API for the language models in each configuration. This metric is measured in US dollars and reflects the computational resources required for each task.

Table 3.3: Results on Q&A and Text-to-SQL tasks, including standard deviation (Std). The best metrics are highlighted with **bold and underline**. The second best are highlighted with **bold**.

Task	**Single-Agent**					**Multi-Agent**				
Model	**LLM Cost**	**Truth.**	**Std**	**Perf.**	**Std**	**LLM Cost**	**Truth.**	**Std**	**Perf.**	**Std**
Q&A										
GPT-3.5-turbo	0.005	2.94	1.48	3.94	1.09	0.02	4.09	1.22	3.82	0.98
GPT-4	0.12	**3.88**	1.41	**4.06**	1.30	0.45	**4.57**	0.79	**4.43**	0.79
Text-to-SQL										
GPT-3.5-turbo	0.009	4.13	1.41	4.44	1.03	0.02	**4.29**	1.20	**4.29**	1.33
GPT-4	0.10	**4.56**	0.96	**4.63**	0.81	0.51	3.20	1.99	3.70	1.89

The comparative analysis between single and multi-agent setups for RAG, using GPT-3.5-turbo and GPT-4 models, revealed insights regarding the metrics of truthfulness, performance, and costs of the language model.

3.2.1 Truthfulness.

In assessing the truthfulness metric, significant differences are noted between the single and multi-agent settings in both Q&A and Text-to-SQL tasks. The results are illustrated in Figures 3.6 and 3.7. For Q&A tasks, GPT-4 in a multi-agent configuration significantly exceeded the performance of the single-agent with a truthfulness score of 4.57 compared to 3.88. The GPT-3.5-turbo model showed distinct results between the two configurations, with the multi-agent surpassing the single-agent with scores of 4.09 and 2.94, respectively. In terms of Text-to-SQL queries, a different outcome was observed. GPT-4 single-agent achieved a score of 4.56, while the same model in the multi-agent configuration obtained 3.20, highlighting a limitation for the multi-agent in this task. Conversely, the GPT-3.5-turbo maintained a more balanced performance between configurations, scoring 4.29 for multi-agent and 4.13 for single-agent.

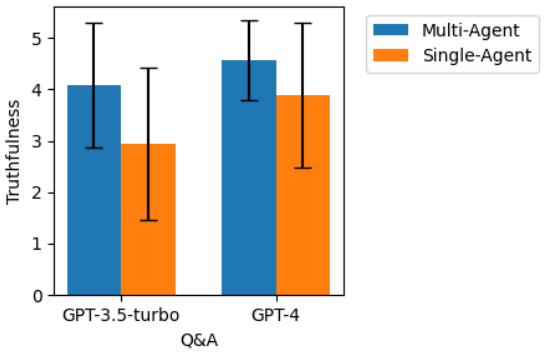

Figure 3.6: Truthfulness and standard deviation in Q&A tasks by LLM model and agent configuration.

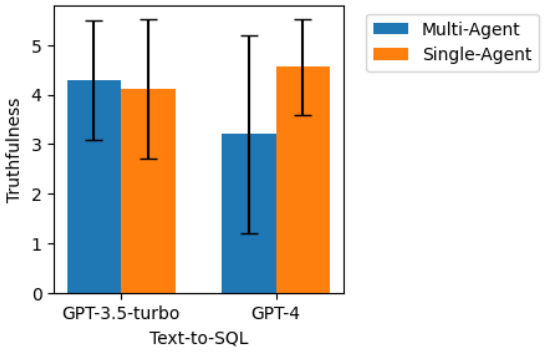

Figure 3.7: Truthfulness and standard deviation in Text-to-SQL tasks by LLM model and agent configuration.

3.2.2 Performance.

The evaluation of LLM performance LI *et al.* (2023) in the tasks of Q&A and Text-to-SQL reveals trends which are similar to the truthfulness results. As shown in Figure 3.8 and 3.9 and summarized in 3.3, the text performance in single and multi-agent setups were compared using GPT-3.5-turbo and GPT-4 models. For Q&A tasks, the multi-agent setup shows a performance boost compared to the single-agent. Notably, the multi-agent GPT-4 achieves a performance score of 4.43, which is higher than the single-agent GPT-4's score of 4.06. This pattern is consistent with the GPT-3.5-turbo, where the multi-agent system also surpasses the single-agent system, scoring 3.82 and 3.94, respectively. These findings emphasize the effectiveness of the multi-agent approach in handling technical user queries.

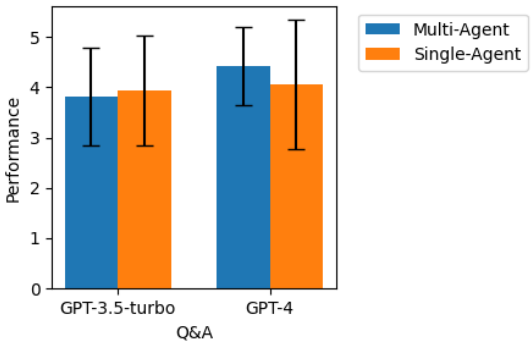

Figure 3.8: Performance and standard deviation in Q&A tasks by LLM model and agent configuration.

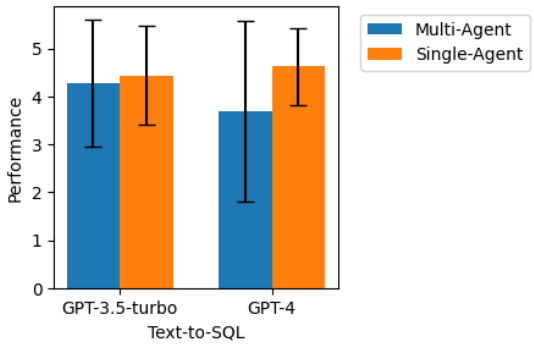

Figure 3.9: Performance and standard deviation in Text-to-SQL tasks by LLM model and agent configuration.

3.2.3 LLM Cost

Language model services are typically composed by a values per token. For instance, GPT-4 model costs US$30.00 (input) and US$60.00 (output) per 1 million tokens received and sent, respectively. The single-agent architecture demonstrated substantially lower costs for both Q&A and Text-to-SQL tasks compared to the multi-agent setup as shown in Figure 3.10. For instance, the average cost of the GPT-4 model OPENAI *et al.* (2023) for a Q&A task was $0.12 per processed question for the single-agent, while the multi-agent recorded an average cost of $0.45. This trend of higher costs for the multi-agent architecture was also maintained for Text-to-SQL tasks, with an average cost of $0.51 for the multi-agent architecture in contrast to $0.10 for the single agent. The higher token count and cost for multi-agent setting is due to the inclusion of intermediate calls, for example, when the "Agent Selector" needs to decide which agent to pass the turn to. All the message history is passed

to the LLM at this stage, substantially increasing the number of tokens submitted and response time.

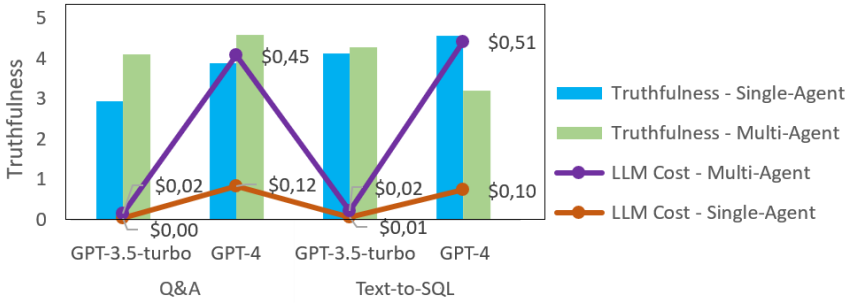

Figure 3.10: Average LLM costs and Truthfulness per completed task according to setup and model.

3.3 Discussion

The comparison between single and multi-agent systems revealed significant differences in terms of performance and cost:

3.3.1 General Performance.

The results indicate that for Q&A tasks in the context of O&G, truthfulness measure was 28% higher with the multi-agent architecture compared to single. However, for Text-to-SQL tasks, this trend was inverted, where the single-agent scored 15% higher.

These findings suggest that for Q&A tasks, the multi-agent setup may be more advantageous in terms of providing truthful information, particularly when utilizing the more advanced GPT-4 model. Conversely, in Text-to-SQL tasks, the GPT-4 model in a single-agent configuration proved more effective. This might imply that the added complexity of managing multiple agents in some tasks does not necessarily lead to improved performance in responses, underscoring the importance of carefully selecting the agent configuration based on the task type and specific features of the language model used.

3.3.2 Cost-Performance Analysis.

While the multi-agent system shows higher truthfulness in Q&A tasks, it is crucial to consider the associated costs. To provide a clearer comparison, let's consider the score/cost ratios. For Q&A tasks using GPT-4, the single-agent configuration yields a ratio of 32.33 truthfulness points per dollar, compared to 10.16 for the multi-agent

setup. This indicates that while the multi-agent system shows a 17.8% improvement in truthfulness, it comes at a 275% increase in cost.

Based on our analysis, we recommend using a multi-agent system for Q&A tasks when the budget allows for it and accuracy is a critical factor. However, decision-makers should consider setting a cost-performance threshold to guide the choice of system configuration, ensuring that the benefits justify the expenses involved.

3.3.3 Model Performance Variations.

Interestingly, our results show that GPT-3.5-turbo outperforms GPT-4 in certain tasks, particularly in the Text-to-SQL multi-agent configuration, despite GPT-4's larger size and more extensive training. This unexpected performance could be attributed to several factors. First, GPT-3.5-turbo may have undergone more specific fine-tuning for structured query tasks, allowing it to excel in Text-to-SQL scenarios. Additionally, GPT-3.5-turbo's training data might be more recent or more relevant to the specific domain of our study. Another possibility is that the smaller model size of GPT-3.5-turbo allows for faster processing and more efficient handling of the multi-agent setup, resulting in better performance in some contexts.

However, it is important to note that GPT-4, when used in a multi-agent setup, demonstrated more consistent truthfulness and performance, as evidenced by its reduced standard deviation in results. This consistency can be particularly advantageous in applications where reliability and accuracy are critical. Multi-agent systems have the advantage of maintaining separate contexts for different aspects of a task. This compartmentalization can lead to better handling of complex, multi-faceted queries, as each agent can focus on its specific context without being overwhelmed by irrelevant information. However, this advantage may be offset in tasks like Text-to-SQL, where maintaining a unified context of the database schema and query structure is crucial, possibly explaining the better performance of single-agent setups in this task. The multi-agent architecture inherently involves multiple stages of information processing, which can serve as natural filtering mechanisms. As information passes from one agent to another, irrelevant or low-quality data may be naturally filtered out, leading to more refined and accurate final outputs. This could explain the superior performance in filtering irrelevant information observed in multi-agent setups.

3.3.4 Economic Efficiency.

The multi-agent architecture incurs significantly higher costs compared to the single-agent system, primarily due to additional intermediate calls to the language model and multiple iterations between agents for action planning. The cost differences be-

tween using GPT-4 and GPT-3.5-turbo are substantial, with GPT-4 being notably more expensive. As detailed in the Cost-Performance Analysis section, the truthfulness per dollar ratio highlights the economic trade-offs between single-agent and multi-agent systems. While the multi-agent system offers improvements in truthfulness, this comes at a considerable increase in cost, impacting the overall economic efficiency.

For a large company with 40,000 knowledge workers, the choice of model and architecture significantly impacts annual costs. Using GPT-4 in a single-agent configuration could result in an annual cost of approximately $4.38 million, while GPT-3.5 would cost around $438,000. However, when employing a multi-agent architecture, the costs increase substantially. The multi-agent configuration with GPT-4 would escalate the annual cost to $16.425 million, representing a dramatic increase due to the 3.75 times higher token usage. Similarly, GPT-3.5 in a multi-agent setup would cost $1.642 million. These estimates assume an average usage pattern of 10,000 tokens per worker per day and underscore the significant financial implications of adopting a multi-agent system, which, while potentially offering performance benefits, comes with a considerable increase in LLM costs.

In summary, while multi-agent systems and more advanced models like GPT-4 offer improvements in performance, the economic efficiency, as measured by truthfulness per dollar, may favor single-agent systems and less costly models like GPT-3.5-turbo, depending on the specific application and budget constraints.

3.3.5 Challenges and Limitations.

During the evaluation of the agents, several challenges and limitations were identified.

Contextualization and Interpretation. In many cases, the single-agent solution had difficulties understanding the context of the question. For instance, a question about cementing was interpreted in the context of the construction industry, a theme to which the language models were more exposed during the training phase. However, the multi-agent structure, with its well-defined roles, better understood the questions and showed superior performance in Q&A tasks, corroborating the findings of LI *et al.* (2024a).

Filtering Irrelevant Information. The agent often receives irrelevant documents along with important ones in the prompt context, and it is up to the LLM to ignore these. For example, when asked about alternatives to accelerate the curing time of cement paste without compromising its integrity at high temperatures, the RAG system retrieved a document that included information about batch cementing to ensure homogeneity during manufacturing and pumping. While this information

is true, it was not relevant to the specific question asked. In this aspect, the multi-agent solution performed better at discarding such irrelevant information, focusing more accurately on the task at hand. Other possible solutions include improving the accuracy of semantic search by adjusting a minimum threshold for similarity measures or through re-ranking techniques such as those proposed by CARRARO (2024) and SUN *et al.* (2023).

Hallucination. During the evaluation of our system, we encountered instances where the agent produced hallucinated information instead of utilizing the appropriate tool to retrieve accurate data, as in BILBAO *et al.* (2023). For example, when asked, "How many anomalies occurred on rig number 05 during August 2023?" the agent was expected to use the Text-to-SQL tool to query the database. However, it bypassed this tool and generated a fabricated response, stating that 5 anomalies occurred, along with detailed descriptions of fictional events. The correct answer, as retrieved from the database, was that 7 anomalies occurred. This hallucination likely resulted from the agent's reliance on its internal knowledge rather than external data retrieval.

In terms of hallucination statistics, our analysis revealed that for Q&A tasks, hallucinations occurred in 9.6% of cases and 3.8% for partially hallucinated. In contrast, Text-to-SQL tasks exhibited a lower hallucination rate, with only 3.6% of responses containing hallucinated information and 96.4% being accurate. These findings highlight the varying susceptibility to hallucination across different task types, emphasizing the need for targeted strategies to mitigate this issue.

Industry Jargon: Specifically analyzing the activity of drilling and completion of offshore wells, the main challenge is the inherently complex and technical nature of the data involved. There were instances of incorrect interpretation of information, likely due to the use of terms, expressions, and themes specific to well construction, to which the language model had little or no exposure during training phase. A possible solution is the implementation of specialized models, which has been pointed out in gray literature as a trend for the coming years SHAH (2024); MEENA (2023); GHOSH (2023).

Tools vs. Performance: It was identified during the experiments that agents with a high amount of tools showed a decline in overall performance. This can be attributed to the added context to the prompts. As the context length increases, the model's ability to accurately interpret and respond diminishes. This is a limitation of current language models, where longer contexts can lead to a dilution of relevant information and increased difficulty in maintaining coherence and accuracy. This conclusion is currently qualitative, as these metrics were not addressed in this experiment.

Queries Involving Proper Names: In queries involving people's names, it

was not possible to retrieve relevant documents using semantic search. For example, when asked to identify the employee associated with a specific key and list knowledge items they registered in the system, the RAG system incorrectly attributed knowledge items to the wrong author. This highlights the difficulty in accurately retrieving information based on proper names, which can be complicated by variations in accentuation, abbreviation, and formatting. A potential solution to be explored is the use of Self-Query Retriever LANGCHAIN (2023), implementing a hybrid search with metadata filters (including proper names) and semantic retrieval of the rest of the query. It is also suggested, in these cases, to use the LEVENSHTEIN (1966) distance to handle possible variations in the spelling of names. This approach could improve the accuracy of retrieving documents related to specific individuals, ensuring that the correct information is associated with the right person.

3.3.6 Practical Implications.

The findings from our study have significant practical implications for the O&G sector, and potentially for other industries characterized by complex and technical data environments:

- **Enhanced Decision-Making Support:** Our results indicate that multi-agent systems provide a 28% higher truthfulness measure in Q&A tasks. This can be particularly beneficial for decision-making in well engineering, where accurate and truthful information is critical. Implementing multi-agent systems in decision-making processes can lead to more reliable and informed decisions, thereby reducing the risk of errors and enhancing operational safety and efficiency.

- **Balancing Performance and Economic Efficiency:** While multi-agent systems offer superior performance in terms of truthfulness, they come with a cost that is 3.7 times higher on average compared to single-agent systems. This highlights the importance of a strategic approach in selecting agent configurations based on specific tasks and budget constraints. A detailed cost-benefit analysis reveals that for Q&A tasks using GPT-4, the single-agent configuration yields a ratio of 32.33 truthfulness points per dollar, compared to 10.16 for the multi-agent setup. While the multi-agent system shows a 17.8% improvement in truthfulness, this comes at a 275% increase in cost. The efficiency varies significantly by task type; in Text-to-SQL tasks, the GPT-4 single-agent outperforms the multi-agent by 42.5% in truthfulness while costing 80.4% less.

- **Reflection and Critic Agents:** A promising approach to enhance the performance of these agents is the use of reflection SHINN *et al.* (2023), a method where agents verbally reflect on task feedback signals and maintain this reflective text in an episodic memory buffer to improve decision-making in subsequent trials. Critic agents are a way to implement reflection in a multi-agent setup. This type of agent is challenging to apply in Q&A tasks over private technical data, as commercial LLMs (OpenAI, Google Bard, and others) have not been deeply trained in the domain and struggle to provide relevant and precise critiques, reinforcing the trend toward increased use of domain-specific models SHAH (2024); MEENA (2023); GHOSH (2023).

- **Task-Specific Agent Configuration:** The study highlights that the complexity of managing multiple agents does not always lead to better performance. In some cases, a single-agent setup might be more effective. This insight can guide the development and deployment of AI systems, ensuring that the configuration of agents is tailored to the specific requirements of the task, thereby optimizing both performance and cost.

- **Potential for Broader Application:** The insights gained from this study are not limited to the O&G sector but can be applied to other industries with similar technical complexities, such as aerospace, pharmaceuticals, and renewable energy. By adopting multi-agent systems in these industries, organizations can improve decision-making, knowledge management, and operational efficiency, driving innovation and competitiveness.

3.3.7 Future Directions.

This work indicates possible pathways for enhancing RAG architectures in O&G sector.

- **Enhancement of IR Semantic Techniques:** There is a critical need to develop more sophisticated semantic search technologies. Future efforts should focus on enhancing the precision of information retrieval by filtering out irrelevant content more effectively. This will ensure that agents can provide more accurate and contextually appropriate responses, crucial for technical domains such as O&G.

- **Development of Domain-Specific Models:** Specialized models tailored specifically to the O&G and other domains, such as biomedical engineering PAL *et al.* (2024), could significantly improve the handling of specific jargon and complex technical data, while reducing LLM costs AREFEEN *et al.*

(2024). Future research should aim to develop and train these models to better understand and interpret the unique language and data types found in O&G, enhancing the overall accuracy of agent responses.

- **Optimization of Tool Use in Agent Performance:** The relationship between the quantity of tools available to an agent and its performance needs further exploration. Future studies should quantify the impact of tool availability on agent efficacy and efficiency, aiming to optimize tool use without overwhelming the agent or diluting performance quality.

- **Integration of Advanced Name Recognition Techniques:** Queries involving proper names pose a significant challenge in semantic search. Integrating advanced retrieval techniques, such as Self-Query Retrievers LANGCHAIN (2023) and LEVENSHTEIN (1966) distance algorithms, could improve the handling of these queries. Future research should focus on enhancing name recognition capabilities to ensure that agents can accurately retrieve and utilize correct information, especially in scenarios where precision is paramount.

- **Extension to Other Complex Domains:** The potential applications of multi-agent systems are not limited to the O&G sector. Future research should explore the adaptation and implementation of these systems in other complex and technical domains, such as aerospace, pharmaceuticals, and renewable energy. Investigating how these systems can support decision-making in these areas will provide valuable insights into their versatility and adaptability.

- **Hybrid Model Experimentation:** Combining the strengths of single and multi-agent systems could yield significant benefits. Future directions should include experimenting with hybrid models that integrate the robustness and depth of multi-agent interactions with the simplicity and efficiency of single-agent systems. This hybrid approach could potentially offer a balanced solution, maximizing performance while managing costs and complexity.

By pursuing these directions, future research can significantly advance the development of multi-agent systems, not only enhancing their application in the O&G sector but also expanding their utility across various technologically intensive activities.

Chapter 4

Experiment 2

This chapter describes the second experimental methodology used to evaluate LLM agents and workflows for answering questions in the well engineering domain. The experiment integrates multiple LLM configurations, agent architectures, and RAG tools, leveraging Petrobras datasets.

To achieve these objectives, this research was conducted through two distinct experimental phases. The first, carried out in 2024, focused on a foundational comparison between single and multi-agent architectures, revealing key insights into their performance, cost, and limitations such as hallucination and context interpretation. The rapid evolution of generative AI frameworks and models prompted a second, more up-to-date experiment in 2025. This second phase built upon the initial findings, also employing non-agentic workflows as baseline and a more rigorous, quantitative evaluation methodology to address the challenges identified in the first experiment and automated evaluation based on the concept commonly reffered to as "LLM-as-a-judge" (GU *et al.* (2025)).

The use of "LLM-as-a-judge" was driven by the sheer volume of responses to be evaluated. With four configurations, two models, and three executions for each, a total of 24 responses were generated for every question in the dataset. Manually assessing this volume of data would have been impractical. Furthermore, previously used metrics like ‘truthfulness‘ had become obsolete. This metric was highly relevant when models frequently hallucinated, a problem that is far less prevalent in the current generation of LLMs.

4.1 Methodology

4.1.1 Experimental Workflow

Dataset Preparation

The experimental workflow was designed to provide a thorough and reproducible evaluation of language model agents within oil well operations. The process begins with the careful preparation of the dataset, which is composed of questions and corresponding ground truth answers derived from a diverse range of operational records, incident reports, and lessons learned. To ensure the quality and relevance of the data, questions undergo a filtering and preprocessing phase where clarity, diversity, and alignment with real-world scenarios are prioritized. This includes removing duplicates, standardizing terminology, and confirming that each question is properly paired with an accurate answer. The dataset is further validated for completeness and consistency, ensuring it represents the full spectrum of operational challenges, such as safety, cementing, and intervention scenarios.

Model and Setup Selection

Following dataset preparation, the experimental design incorporates a variety of agent architectures. These include approaches where questions are routed to specialized agents, single-agent systems that centralize all reasoning and retrieval, and multi-agent frameworks that leverage collaboration among specialized agents under a supervisory structure. Each of these configurations is evaluated using different language models, allowing for a comprehensive assessment of how model choice and agent setup influence performance. The agents are also provided with access to advanced retrieval tools and domain-specific knowledge bases, enabling them to draw on a broad foundation of operational expertise.

Execution Loop

The core of the experimental workflow is an execution loop depicted in Algorithm 1. For each combination of question, agent setup, and language model, the system systematically loads the relevant data, configures the agent, and executes the workflow. Throughout this process, all responses and intermediate reasoning steps are meticulously logged. This approach not only ensures systematic coverage of all experimental conditions but also provides full traceability for subsequent analysis. The automation of these procedures guarantees consistency and reproducibility, while the comprehensive logging facilitates in-depth evaluation and comparison of agent performance across a range of operational scenarios.

Algorithm 1 Experiment Execution Loop

Require: questions, setups, models
Ensure: results

```
function RunExperiment
    results ← {}
    for all question ∈ questions do
        ground_truth ← question.ground_truth
        for all setup ∈ setups do
            for all model ∈ models do
                agent ← InitializeAgent(setup, model)
                response ← agent.ProcessQuestion(question)
                metrics ← EvaluateResponse(response, ground_truth)
                results[question, setup, model] ← {
                        "response" : response,
                        "metrics" : metrics,
                        "execution_trace" : agent.trace
                }
            end for
        end for
    end for
    return AggregateResults(results)
end function
```

Evaluation and Metrics

Following the execution of all experimental combinations, a comprehensive evaluation framework is applied to assess agent performance. The system calculates a suite of quantitative metrics for each question, setup, and model combination by comparing the generated answers against the established ground truth. These metrics include standard performance indicators such as accuracy, precision, recall, and F1 score, which provide a multifaceted view of response quality. For open-ended questions where binary correctness measures are insufficient, a confusion matrix approach is implemented to capture nuances in answer quality and content coverage. Additionally, the system measures answer size ratio relative to ground truth, offering insights into model verbosity and conciseness. These metrics are then aggregated across different dimensions to enable meaningful comparisons between agent architectures and language models, revealing patterns in performance across various operational scenarios and question types.

Reproducibility and Quality Control

To ensure scientific rigor and reproducibility, the experimental methodology incorporates robust tracking of all environmental variables and configuration parameters. The system maintains detailed logs of the computational environment, in-

cluding software versions, dependency specifications, and hardware characteristics that might influence results. All experimental parameters, from model identifiers to dataset specifications, are systematically recorded alongside the results they generate. Throughout the experimental process, periodic validation checks are performed to maintain data integrity and result consistency, with anomalies flagged for investigation. This comprehensive approach to reproducibility not only facilitates verification of findings but also enables future extensions of the research with comparable baselines. The quality control measures embedded in the workflow ensure that conclusions drawn from the experiments rest on a foundation of methodological soundness and data reliability.

<insert workflow diagram or pseudocode here to illustrate the above stages>

4.1.2 Data Sources

The experimental evaluation relies on a carefully curated collection of data sources that represent the diverse knowledge domains relevant to oil well operations. At the core of the experiment is a comprehensive questions dataset containing structured entries that simulate real-world queries an operator might encounter. This dataset was developed through extensive collaboration with domain experts and analysis of historical operational records. Each entry in the dataset contains a question formulated in natural language, a unique identifier, categorical metadata to facilitate analysis, and a corresponding ground truth answer validated by subject matter experts. The questions span various complexity levels, from factual inquiries to complex reasoning scenarios that require integration of multiple knowledge sources.

To provide the language models with the necessary domain knowledge, the experiment incorporates several specialized knowledge bases that reflect different aspects of oil well operations:

- **Knowledge Bases and Tools**:
  - **Lessons**: A repository of knowledge items capturing insights, best practices, and technical know-how from past oil well operations. These lessons represent institutional memory and expertise accumulated over years of operational experience.
  - **Alertas SMS**: A collection of safety alerts and incident reports documenting past events, near-misses, and accidents, providing critical safety information and preventative measures.
  - **Cronoweb**: A comprehensive database of scheduling information and intervention records, detailing maintenance activities, equipment deployments, and operational timelines.

– **SITOP**: Detailed daily operational logs from drilling rigs, containing technical parameters, operational decisions, and situational reports from active drilling operations.

These knowledge sources were preprocessed to ensure consistency, remove sensitive information, and optimize retrieval performance. The integration of these diverse data sources enables a holistic evaluation of how language model agents navigate the complex informational landscape of oil well operations, from technical specifications to safety protocols and historical precedents.

<insert table or image summarizing datasets and tools here>

4.1.3 System Architecture

The experimental system was implemented using modern Python frameworks specialized for language model orchestration and agent workflows. The architecture leverages the LangChain and LangGraph ecosystems, which provide robust foundations for building complex language model applications with multiple components and state management. This subsection details the modular design of the system, highlighting how different components interact to enable systematic evaluation of language model agents in oil well operations.

Experiment Orchestration

At the core of the system architecture is an experiment orchestration layer responsible for coordinating the entire evaluation process. This component manages the loading of questions from the dataset, systematically iterates through different model and setup combinations, and ensures proper logging of results. The orchestrator maintains experiment state across multiple runs, handles error recovery, and implements checkpointing to allow for resumption of long-running experiments. By centralizing control flow, this component ensures that all experimental conditions are tested consistently and that results are captured in a standardized format for subsequent analysis.

Agent Workflow Frameworks

The system implements multiple agent workflow frameworks to evaluate different approaches to question answering in the oil well domain. These frameworks define the flow of information and decision-making processes within and between language model agents. The implemented workflows include a Linear-Flow with Router (CORTEX) that directs questions to specialized processing paths, a Single-Agent approach that centralizes all reasoning and tool use, and a Multi-Agent Supervisor

framework that coordinates multiple specialized agents. Each workflow is defined declaratively, specifying the sequence of operations, decision points, and information exchange patterns that govern agent behavior during question processing.

Nodes and Tool Integration

The system architecture includes specialized nodes that implement specific reasoning steps and tool-calling logic. These nodes serve as the building blocks of agent workflows, encapsulating discrete functionality such as question analysis, knowledge retrieval, and answer synthesis. The tool integration layer provides agents with access to external knowledge sources through a standardized interface, enabling semantic search over domain-specific corpora, structured data queries, and other specialized operations. This modular approach to tool integration allows for consistent evaluation of how different agent architectures leverage available tools and knowledge sources.

Prompt Engineering and System Messages

A critical component of the architecture is the prompt engineering layer, which defines the instructions and context provided to language models. This includes carefully crafted system messages that establish the role and capabilities of each agent, prompt templates that structure inputs consistently across experimental conditions, and few-shot examples that guide model behavior. The system maintains a library of prompt variants optimized for different tasks within the question-answering workflow, ensuring that each agent receives appropriate guidance while maintaining experimental control.

State Management and Metrics

The architecture incorporates a comprehensive state management system that tracks the progress of experiments, maintains contextual information across agent interactions, and captures intermediate reasoning steps. This component is tightly integrated with the metrics calculation subsystem, which computes performance indicators in real-time as experiments progress. The metrics framework implements various evaluation approaches, from simple accuracy measures to sophisticated semantic similarity calculations, providing multi-dimensional assessment of agent performance. All experimental data, including intermediate states and final results, is persisted in structured formats to enable both immediate feedback and in-depth post-experiment analysis.

<insert system architecture diagram here>

4.1.4 Experimental Setups

To comprehensively evaluate language model performance in well construction operations, the experiment employed multiple agent architectures and model configurations. This subsection details the different experimental setups, highlighting their design principles, operational characteristics, and the rationale behind their selection. The experimental design deliberately incorporates contrasting approaches to agent architecture, enabling comparative analysis of different strategies for complex question answering in specialized domains.

Linear-Flow

The Linear-Flow architecture represents the simplest RAG design, where user input is processed in a strictly sequential manner. In this setup, the user's query is handled by a single LLM step, which carries all the instructions (PT1, PT2, PT3 and PT4, as depicted in 4.1) required for the generation of various types of search queries. These instruction prompts are often quite long, as they are carefully crafted to produce high-quality queries for the vector store. Due to the aggregation of all instruction prompts within a single LLM invocation, the resulting context becomes notably extensive. This can lead to performance degradation as the context length increases [O QUE PODE SER VISTO NO GRAFICO TAL EM CONTRASTE COM O SETUP TAL QUE DIVIDE OS PROMPTS EM PARTES].

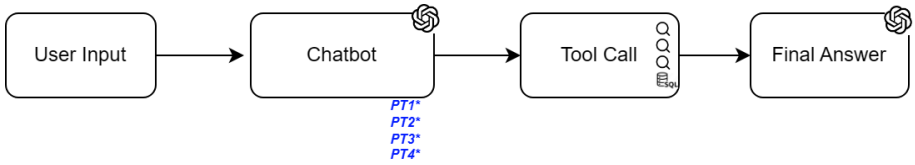

Figure 4.1: Linear-Flow architecture. PT1 indicates Prompt for Tool 1 and so on.

Linear-Flow with Router

[VERSÃO 1] The Linear-Flow with Router architecture implements a sequential processing pipeline with routing, in order to divide and reduce tool instruction prompts. In this setup, questions first pass through a router that analyzes the query content and determines the most appropriate specialized node for the user query at hand.

[VERSÃO 2] The Linear-Flow with Router paradigm extends the basic linear flow by introducing a routing mechanism that enables the distribution of tool instruction prompts into specialized nodes with smaller prompts. User questions first pass through a router that determines the most appropriate specialized node. As illustrated in Figure 4.2, instead of a single node generating different types of retrieval

queries, we have several nodes (or sub-queries), each with its own specialized tool. Each sub-query is then processed independently by separate tool invocations.

This approach offers several advantages:

- **Increased Throughput:** By distributing sub-tasks across multiple tools, the system can handle more complex or multi-faceted user requests efficiently.
- **Specialization:** Each tool can be tailored to address a specific aspect of the user's query, allowing for more accurate and relevant results.
- **Scalability:** The architecture naturally supports scaling, as additional tools can be added to handle more sub-queries or specialized tasks.

In practice, the router acts as an orchestrator, analyzing the user input and generating multiple targeted queries (PT1*, PT2*, PT3*, PT4* in the figure). These queries are dispatched to their respective tools, and the results are aggregated to form the final answer. This method is particularly effective for tasks that can be decomposed into independent components, such as multi-part questions or workflows requiring different types of expertise.

Compared to the standard linear flow, the use of a router introduces additional complexity in query generation and result aggregation but enables a significant boost in system flexibility and performance.

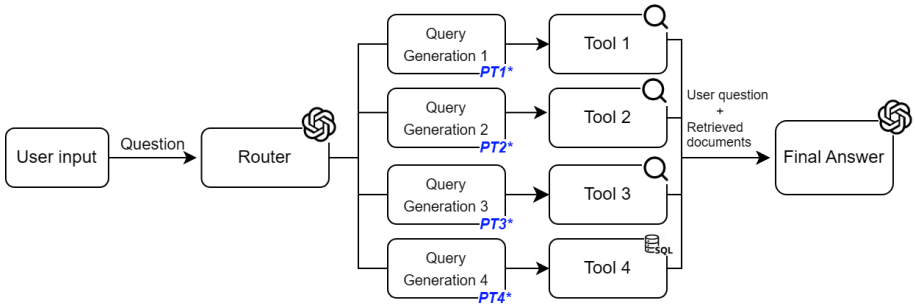

Figure 4.2: Linear-Flow with Router architecture.

Single-Agent

The Single-Agent approach represents a centralized architecture where a single language model agent handles the entire question-answering process. This agent has access to the full suite of retrieval tools and knowledge sources, making independent decisions about which tools to invoke and how to synthesize information into coherent answers. The design emphasizes end-to-end reasoning within a unified context, allowing the model to maintain a consistent understanding throughout the process.

This approach tests the capability of language models to manage complex workflows autonomously, balancing between exploration of different knowledge sources and focused answer generation without the overhead of inter-agent communication.

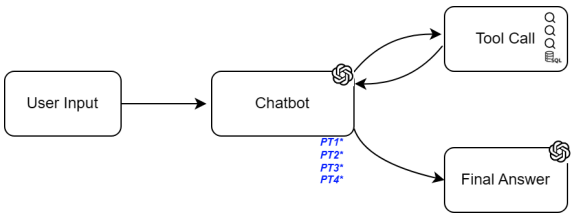

Figure 4.3: Single-Agent architecture

Multi-Agent Supervisor

The Multi-Agent Supervisor setup implements a collaborative approach where multiple specialized agents work together under the coordination of a supervisor agent. Each specialized agent focuses on a specific domain of knowledge or reasoning skill, such as retrieval, analysis, or explanation generation. The supervisor agent orchestrates the collaboration, delegating subtasks to appropriate specialized agents, integrating their contributions, and ensuring coherence in the final answer. This architecture explores the potential benefits of distributed cognition, where complex reasoning is decomposed into manageable components handled by purpose-built agents. The framework includes mechanisms for resolving conflicts between agents and synthesizing potentially divergent perspectives into unified responses.

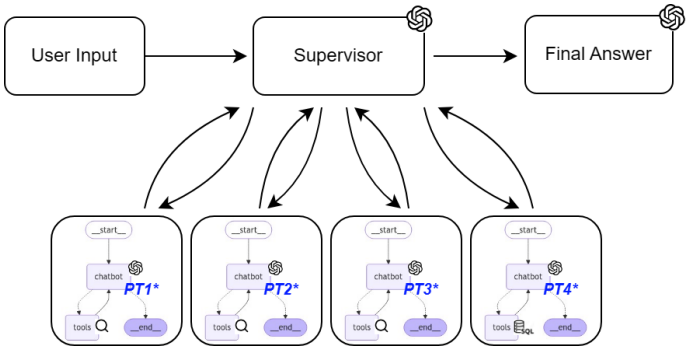

Figure 4.4: Multi-Agent setup with one supervisor and 4 specialist agents.

<insert table summarizing experimental setups and models here>

4.1.5 Execution Details

The experiment was driven by a script without manual intervention during the evaluation process. A main execution loop systematically iterated through all combina-

tions of questions, agent setups, and language models defined in the experimental design.

Tool Integration and Knowledge Access

During execution, the agent systems accessed domain-specific knowledge through a standardized tool interface layer. This layer provided consistent access patterns across all experimental configurations, ensuring that differences in performance could be attributed to agent architecture rather than variations in knowledge availability. The tool integration framework supported a diverse range of knowledge access methods, including semantic search over unstructured text corpora, structured queries against relational databases, and specialized information extraction routines tailored to the oil well operations domain. Each tool invocation was executed within a controlled environment that captured performance metrics such as latency and resource utilization, providing additional dimensions for analysis beyond answer correctness. The standardization of tool interfaces across agent architectures was a critical design decision that enabled fair comparison while still allowing each architecture to implement its own strategy for tool selection and result interpretation.

Comprehensive Logging and Observability

A cornerstone of the experimental methodology was the implementation of comprehensive logging throughout the execution process. The system captured detailed records of each step in the question-answering workflow, from initial question parsing to final answer generation. These logs included intermediate reasoning steps, tool invocations with their inputs and outputs, and internal state transitions within the agent systems. All experimental artifacts were persisted in structured formats that facilitated both automated analysis and manual inspection. The logging system implemented a hierarchical organization that linked high-level metrics to the detailed execution traces that produced them, enabling root cause analysis of performance patterns. This observability infrastructure was essential for understanding not just what results were produced, but how and why different agent architectures arrived at their answers, providing insights into their reasoning processes and failure modes.

<insert code snippet or pseudocode of main execution loop here>

4.1.6 Evaluation Metrics

The evaluation of each experimental run is grounded in a comprehensive set of metrics designed to capture both the correctness and the quality of the system's responses. Standard quantitative measures such as accuracy, precision, recall, and

F1 score are calculated by comparing the answers generated by the agent systems to the established ground truth for each question. These metrics provide a multifaceted view of performance, indicating not only how often the system produces correct answers but also how well it balances false positives and false negatives.

For questions that are open-ended or less amenable to binary correctness, the evaluation framework employs a confusion matrix approach. This allows for a more nuanced assessment, capturing partial correctness and the degree to which the system's response overlaps with the expected content. Additionally, the methodology includes the calculation of the answer size ratio, which measures the verbosity of the generated answer relative to the ground truth. This metric helps to identify tendencies toward overly concise or excessively verbose responses, offering further insight into the models' behavior and suitability for practical deployment.

4.1.7 Limitations

While the experimental methodology strives for rigor and comprehensiveness, several limitations must be acknowledged. One key limitation concerns the coverage of the dataset: although the question set is carefully curated to represent a broad range of operational scenarios, it may not capture the full diversity of real-world challenges encountered in oil well operations. Similarly, the models and agent architectures evaluated are constrained by the available computational resources and the current state of language modeling technology, which may limit their ability to generalize beyond the scenarios tested.

Another limitation arises from the reliance on ground truth answers, which, despite expert validation, may still reflect subjective judgments or incomplete information in certain cases. Furthermore, the evaluation metrics, while robust, may not fully capture qualitative aspects of answer usefulness or clarity, especially in highly technical or ambiguous situations. Recognizing these limitations is essential for interpreting the results and for guiding future research aimed at addressing these gaps.

4.1.8 Summary

This methodology provides a systematic framework for comparing different agent architectures and large language models in the context of complex question answering for oil well operations. By integrating rigorous evaluation metrics, robust reproducibility practices, and a clear acknowledgment of limitations, the approach enables meaningful insights into the strengths and weaknesses of various system designs. The findings derived from this methodology can inform both the deployment of language model agents in operational settings and the ongoing development of

more capable and reliable AI systems for specialized industrial domains.

4.1.9 Quality Assessment with LLM-as-a-Judge

To evaluate the quality of the responses generated by the RAG system, an automated evaluation approach known as *LLM-as-a-Judge* ZHENG *et al.* (2023) was adopted. This method uses a large language model (LLM) as an impartial judge to compare the system-generated response with a reference response, known as the *Ground Truth* (GT). Using an LLM for evaluation allows for a more granular and scalable analysis than manual human evaluation, while also capturing nuances of semantics and content.

The evaluation process was implemented through a structured *prompt* that instructs the LLM-judge to perform an analysis based on the concepts of a confusion matrix. The evaluation flow for each question-answer pair is as follows:

1. **Decomposition into Statements:** The LLM-judge is instructed to decompose both the system's response and the *Ground Truth* response into a set of atomic and objective statements. This allows for a detailed comparison, rather than a holistic and subjective evaluation.

2. **Classification of Statements:** Based on the comparison between the statements, the LLM-judge classifies them into three categories, following the logic of a confusion matrix:

   - **True Positive (TP):** Correct statements made by the system that are also present in the *Ground Truth* response.
   - **False Positive (FP):** Incorrect, irrelevant, or hallucinated statements made by the system that are not in the *Ground Truth* response.
   - **False Negative (FN):** Statements present in the *Ground Truth* response that were omitted by the system.

   The True Negative (TN) category is not applicable in this context, as the goal is to measure the precision and completeness of the information provided.

3. **Calculation of Metrics:** Based on the count of statements in each category (TP, FP, FN), the following information retrieval metrics are calculated:

   - **Precision:** Measures the proportion of correct statements among all statements made by the system. It is calculated as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall:** Measures the proportion of correct statements that the system managed to retrieve compared to the total number of statements in the *Ground Truth* response. It is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1-Score:** Is the harmonic mean of Precision and Recall, providing a single metric that balances both values:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

This methodology allows for a quantitative and objective evaluation of the accuracy and completeness of the system's responses, providing valuable *insights* into its performance in different scenarios.

4.2 Results and Discussion

4.2.1 Performance

In this section, the F1-score charts are presented, which is the main metric for performance evaluation in this work. The corresponding charts for the precision and recall metrics are available in Appendix A.2.

É importante ressaltar que todos os resultados apresentados correspondem à melhor performance obtida em três execuções independentes para cada configuração de agente e modelo. A adoção desta metodologia visa mitigar a variabilidade inerente aos processos estocásticos presentes em muitos algoritmos de aprendizado de máquina. Fatores como a inicialização aleatória de pesos ou a aleatorização de dados podem levar a resultados distintos a cada execução. Ao selecionar o melhor resultado, busca-se apresentar o potencial máximo de cada configuração, reduzindo a chance de que uma performance inferior, causada por um ótimo local ou uma inicialização desfavorável, seja erroneamente interpretada como a capacidade real do modelo.

...

...

...

...

F1 Score

...

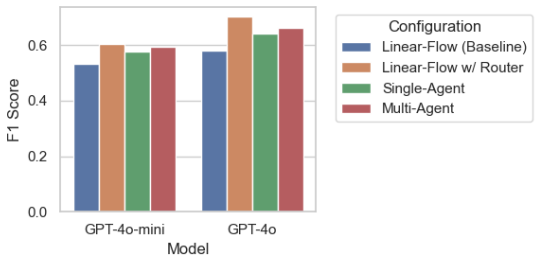

Figure 4.5: F1 score by model and configuration.

...

...

...

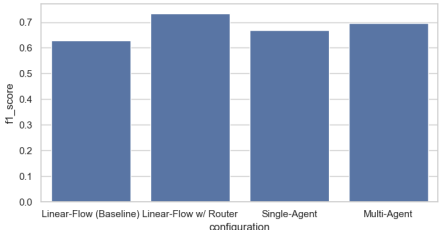

Figure 4.6: Average F1 score by configuration.

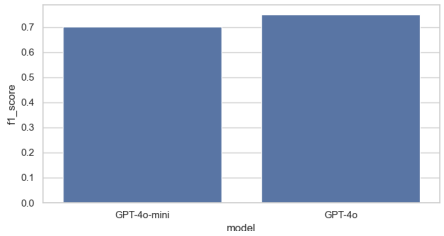

Figure 4.7: Average F1 score by model.

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

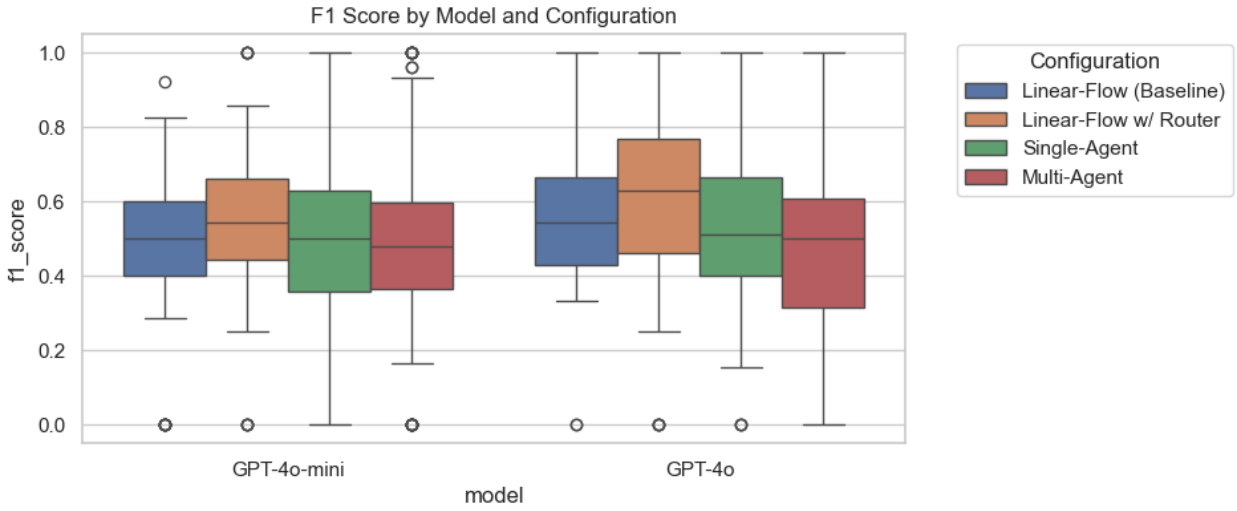

Figure 4.8: F1 Score distribution by model and configuration of agents

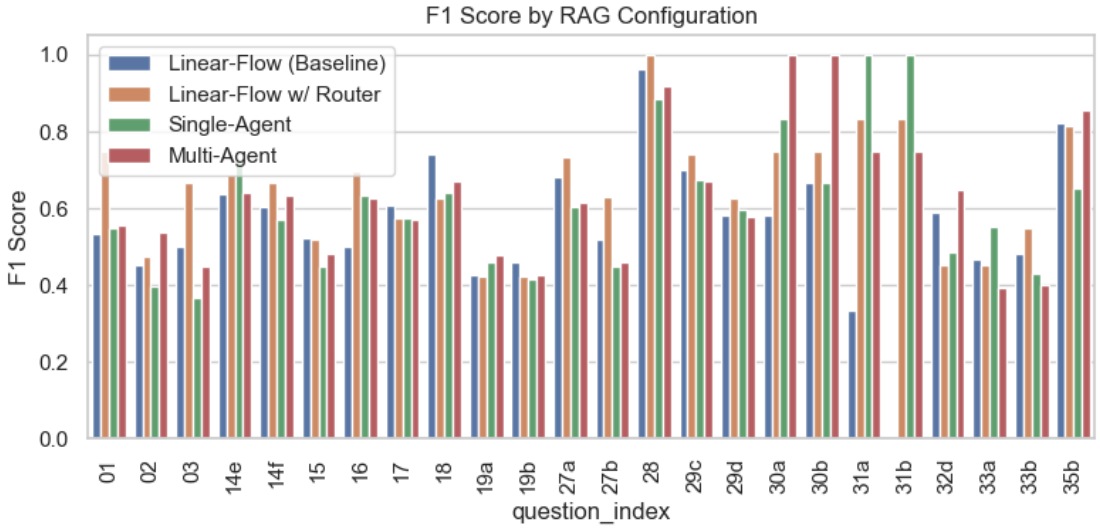

Figure 4.9: Best F1 score by question index and configuration.

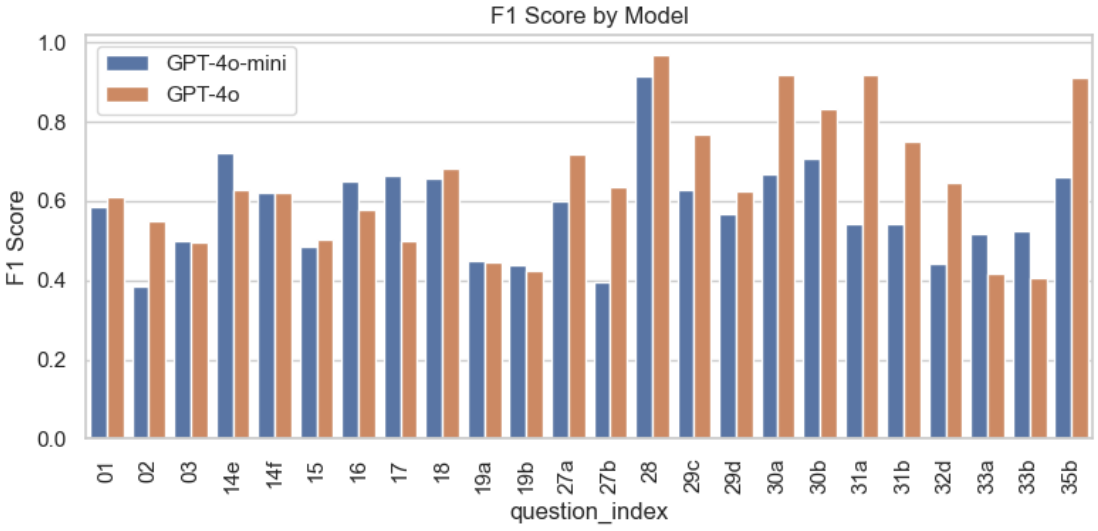

Figure 4.10: Best F1 score by question index and model.

Precisão

...

...

...

...

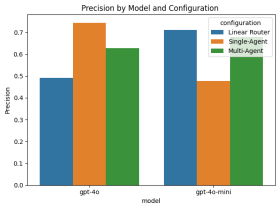

Figure 4.11: Precisão por modelo e configuração.

...

...

...

...

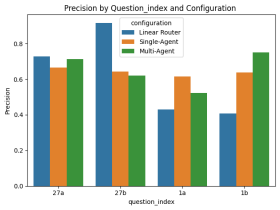

Figure 4.12: Precisão por pergunta e configuração.

...

...

...

Recall

...

...

...

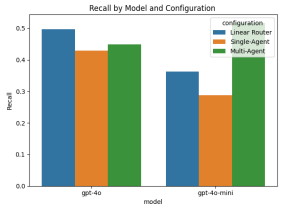

Figure 4.13: Recall by model and configuration.

...

...

...

...

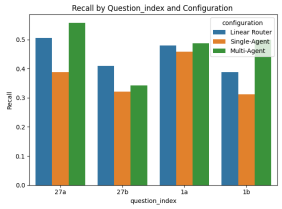

Figure 4.14: Recall por pergunta e configuração.

...

...

...

Chapter 5

Conclusões 1

Os resultados deste estudo destacam o potencial das arquiteturas multiagente baseadas em LLMs no setor de O&G, especialmente no domínio da engenharia de poços. A capacidade de processar e responder a consultas complexas abre caminho para uma transformação digital significativa na área.

Nossa análise comparativa de arquiteturas de agente único e multiagente, utilizando GPT-3.5-turbo e GPT-4, revela um panorama detalhado de trade-offs entre desempenho e eficiência econômica. Os sistemas multiagente demonstram uma veracidade 28% maior em tarefas de perguntas e respostas (Q&A), especialmente com GPT-4, em comparação com sistemas de agente único. No entanto, eles incorrem em custos de LLM que são, em média, 3,7 vezes maiores devido às complexidades da comunicação entre agentes. Em contraste, os sistemas de agente único se destacam em tarefas de Text-to-SQL, apresentando um desempenho 15% melhor do que as configurações multiagente. Essa dinâmica de custo-benefício exige uma consideração cuidadosa ao implementar RAG em cenários do mundo real, onde precisão e restrições financeiras devem ser equilibradas.

Destacamos vários desafios encontrados durante nossos experimentos, incluindo questões de contextualização, necessidade de filtragem de informações mais refinada e a persistência de alucinações. Esses desafios sublinham a necessidade de pesquisas contínuas em áreas como modelos especializados em domínios específicos, técnicas avançadas de busca semântica e arquiteturas híbridas que combinem as forças dos sistemas de agente único e multiagente.

As implicações práticas deste estudo vão além do setor de O&G. Os insights alcançados aqui são aplicáveis a qualquer domínio intensivo em conhecimento que lide com grandes volumes de dados técnicos. Ao focar em aprimorar os mecanismos de recuperação, desenvolver LLMs específicos de domínio e otimizar as interações entre agentes e ferramentas, pavimentamos o caminho para soluções RAG mais eficazes, confiáveis e econômicas em diversos setores.

Os principais pontos do estudo são os seguintes: sistemas multiagente oferecem su-

perior veracidade em tarefas de Q&A, embora a um custo significativamente maior. Arquiteturas de agente único, por outro lado, se destacam em tarefas de Text-to-SQL. Apesar das vantagens, persistem vários desafios, incluindo questões de contextualização, filtragem, alucinação e vocabulário específico de domínio.

Pesquisas futuras devem focar no desenvolvimento de modelos especializados, no avanço das técnicas de recuperação e na exploração de arquiteturas híbridas. As lições aprendidas deste estudo têm implicações mais amplas e podem se estender a outros domínios técnicos complexos. Ao abordar as limitações identificadas neste estudo e abraçar as tendências emergentes em sistemas multiagente e tecnologia RAG, podemos desbloquear seu potencial total, revolucionando a tomada de decisões, a gestão do conhecimento e a eficiência operacional em indústrias complexas em todo o mundo.

Chapter 6

Conclusões 2 AAA

...

References

WU, Q., BANSAL, G., ZHANG, J., et al. "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation", 2023. doi: 10.48550/arXiv.2308.08155.

KAR, A. K., VARSHA, P. S. "Unravelling the Impact of Generative Artificial Intelligence (GAI) in Industrial Applications: A Review of Scientific and Grey Literature", *Global Journal of Flexible Systems Management*, v. 24, pp. 659–689, 12 2023. ISSN: 09740198. doi: 10.1007/s40171-023-00356-x.

ECKROTH, J., GIPSON, M. "Answering Natural Language Questions with OpenAI's GPT in the Petroleum Industry", pp. 16–18, 2023. doi: 10.2118/214888-MS. Available at: <http://onepetro.org/SPEATCE/proceedings-pdf/23ATCE/3-23ATCE/D031S032R005/3301837/spe-214888-ms.pdf/1>.

DELLACQUA, F., SARAN, A., MCFOWLAND, R. E., et al. "Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of AI on Knowledge Worker Productivity and Quality", *Harvard Business School: Technology and Operations Management Unit Working Paper Series*, 2023. doi: 10.2139/ssrn.4573321. Available at: <https://ssrn.com/abstract=4573321>.

HATZIUS, J., BRIGGS, J., KODNANI, D., et al. "The Potentially Large Effects of Artificial Intelligence on Economic Growth". 2023.

SINGH, A., JIA, T., NALAGATLA, V. "Generative AI Enabled Conversational Chatbot for Drilling and Production Analytics", *Day 2 Tue, October 03, 2023*, 10 2023. doi: 10.2118/216267-MS. Available at: <https://onepetro.org/SPEADIP/proceedings/23ADIP/2-23ADIP/D021S065R002/534485>.

HADI, M. U., QASEM AL TASHI, QURESHI, R., et al. "A Survey on Large Language Models: Applications, Challenges, Limitations, and Practical

Usage", 2023. doi: 10.36227/techrxiv.23589741.v1. Available at: <https://doi.org/10.36227/techrxiv.23589741.v1>.

GU, J., JIANG, X., SHI, Z., et al. "A Survey on LLM-as-a-Judge", 3 2025. Available at: <http://arxiv.org/abs/2411.15594>.

BADIRU, A. B., OSISANYA, S. O. *Project Management for the Oil and Gas Industry: A World System Approach*. Boca Raton, FL 33487-2742, CRC Press, 1 2016. ISBN: 9781420094268. doi: 10.1201/b13755.

THOMAS, J. E. *Fundamentos de Engenharia de Petróleo*. 2nd ed. Av. Presidente Vargas 435, Rio de Janeiro, RJ - 20.077-900, Editora Interciência, 2004.

BRAVO, C., SAPUTELLI, L., RIVAS, F., et al. "State of the art of artificial intelligence and predictive analytics in the E&P industry: A technology survey", *SPE Journal*, v. 19, n. 4, pp. 547–563, 2014. ISSN: 1086055X. doi: 10.2118/150314-pa. Available at: <http://onepetro.org/SJ/article-pdf/19/04/547/2099035/spe-150314-pa.pdf/1>.

GUDALA, M., NAIYA, T. K., GOVINDARAJAN, S. K. "Remediation of heavy oil transportation problems via pipelines using biodegradable additives: An experimental and artificial intelligence approach", *SPE Journal*, v. 26, n. 2, pp. 1050–1071, apr 2021. ISSN: 1086055X. doi: 10.2118/203824-PA.

GOHARI, M. S. J., NIRI, M. E., SADEGHNEJAD, S., et al. "Synthetic Graphic Well Log Generation Using an Enhanced Deep Learning Workflow: Imbalanced Multiclass Data, Sample Size, and Scalability Challenges", *SPE Journal*, v. 29, pp. 1–20, 2024. ISSN: 1086055X. doi: 10.2118/217466-PA. Available at: <http://onepetro.org/SJ/article-pdf/29/01/1/3358626/spe-217466-pa.pdf/1>.

RAHMANI, A. M., AZHIR, E., ALI, S., et al. "Artificial intelligence approaches and mechanisms for big data analytics: a systematic study", *PeerJ Computer Science*, v. 7, pp. 1–28, 4 2021. ISSN: 23765992. doi: 10.7717/peerj-cs.488.

LIDDY, E. *Natural Language Processing*. Encyclopedia of Library and Information Science, 2001. Available at: <https://surface.syr.edu/istpub>.

ANTONIAK, M., DALGLIESH, J., VERKRUYSE, M., et al. "Natural language processing techniques on oil and gas drilling data", *Society of Petroleum Engineers - SPE Intelligent Energy International Conference and Exhibition*, 2016. doi: 10.2118/181015-MS.

CASTIÑEIRA, D., TORONYI, R., SALERI, N. "Machine Learning and Natural Language Processing for Automated Analysis of Drilling and Completion Data", pp. 23–26, 2018. doi: 10.2118/192280-MS. Available at: <http://onepetro.org/SPESATS/proceedings-pdf/18SATS/All-18SATS/SPE-192280-MS/1246545/spe-192280-ms.pdf/1>.

VASWANI, A., BRAIN, G., SHAZEER, N., et al. "Attention Is All You Need". 2017. Available at: <https://arxiv.org/abs/1706.03762>.

OPENAI, ACHIAM, J., ADLER, S., et al. "GPT-4 Technical Report", v. 4, pp. 1–100, 2023. doi: 10.48550/arXiv.2303.08774. Available at: <http://arxiv.org/abs/2303.08774>.

MOSSER, L., AURSAND, P., BRAKSTAD, K. S., et al. "Exploration Robot Chat: Uncovering Decades of Exploration Knowledge and Data with Conversational Large Language Models". In: *Day 1 Wed, April 17, 2024*. SPE, apr 2024. doi: 10.2118/218439-MS. Available at: <https://onepetro.org/SPEBERG/proceedings/24BERG/1-24BERG/D011S002R006/544177>.

ISKE, P., BOERSMA, W. "Connected brains. Question and answer systems for knowledge sharing: Concepts, implementation and return on investment". 2005. ISSN: 13673270.

TREUDE, C., BARZILAY, O., STOREY, M.-A. "How do programmers ask and answer questions on the web? (NIER track)". In: *Proceedings of the 33rd International Conference on Software Engineering*, ICSE '11, p. 804–807, New York, NY, USA, 2011. Association for Computing Machinery. ISBN: 9781450304450. doi: 10.1145/1985793.1985907. Available at: <https://doi.org/10.1145/1985793.1985907>.

QIN, B., HUI, B., WANG, L., et al. "A Survey on Text-to-SQL Parsing: Concepts, Methods, and Future Directions", 8 2022. doi: 10.48550/arXiv.2208.13629. Available at: <http://arxiv.org/abs/2208.13629>.

OPENAI. "Embeddings - OpenAI API". 2023. Available at: <https://platform.openai.com/docs/guides/embeddings>.

DENG, X., AWADALLAH, A. H., MEEK, C., et al. "Structure-Grounded Pretraining for Text-to-SQL", pp. 1337–1350, 2021. doi: 10.18653/v1/2021.naacl-main.105. Available at: <http://dx.doi.org/10.18653/v1/2021.naacl-main.105>.

DENG, X., GU, Y., ZHENG, B., et al. “MIND2WEB: Towards a Generalist Agent for the Web”, 2023. Available at: <https://osu-nlp-group.github.io/Mind2Web>.

LEWIS, P., PEREZ, E., PIKTUS, A., et al. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. 2020. Available at: <https://github.com/huggingface/transformers/blob/master/>.

LI, H., SU, Y., CAI, D., et al. “A Survey on Retrieval-Augmented Text Generation”, 2 2022. doi: 10.48550/arXiv.2202.01110.

LIU, J., JIN, J., WANG, Z., et al. “RETA-LLM: A Retrieval-Augmented Large Language Model Toolkit”, 6 2023. Available at: <https://arxiv.org/abs/2306.05212v1>.

ZHAO, R., CHEN, H., WANG, W., et al. “Retrieving Multimodal Information for Augmented Generation: A Survey”, 3 2023. doi: 10.48550/arXiv.2303.10868. Available at: <http://arxiv.org/abs/2303.10868>.

RUSSELL, S. *Artificial intelligence: a modern approach*. Pearson, 2020. ISBN: 0134610997.

XI, Z., CHEN, W., GUO, X., et al. “The Rise and Potential of Large Language Model Based Agents: A Survey”. 2023.

LI, J., ZHANG, Q., YU, Y., et al. “More Agents Is All You Need”, 2024a. doi: 10.48550/arXiv.2402.05120.

LI, H., DONG, Q., CHEN, J., et al. “LLMs-as-Judges: A Comprehensive Survey on LLM-based Evaluation Methods”. 2024b. Available at: <https://arxiv.org/abs/2412.05579>.

ZHENG, L., CHIANG, W.-L., SHENG, Y., et al. “Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena”, 12 2023. Available at: <http://arxiv.org/abs/2306.05685>.

KIM, S., SHIN, J., CHO, Y., et al. “Prometheus: Inducing Fine-grained Evaluation Capability in Language Models”, 3 2024. Available at: <http://arxiv.org/abs/2310.08491>.

LI, C., WANG, J., ZHANG, Y., et al. “Large Language Models Understand and Can Be Enhanced by Emotional Stimuli”, 2023. doi: 10.48550/arXiv.2307.11760.

CARRARO, D. “Enhancing Recommendation Diversity by Re-ranking with Large Language Models”, 2024. doi: 10.48550/arXiv.2401.11506.

SUN, W., YAN, L., MA, X., et al. “Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents”, 2023. doi: 10.48550/arXiv.2304.09542. Available at: <https://arxiv.org/abs/2304.09542>.

BILBAO, D., GELBUKH, A., RODRIGO, A., et al. “A Mathematical Investigation of Hallucination and Creativity in GPT Models”, *Mathematics 2023*, v. 11, pp. 2320, 5 2023. ISSN: 2227-7390. doi: 10.3390/MATH11102320. Available at: <https://www.mdpi.com/2227-7390/11/10/2320/htmhttps://www.mdpi.com/2227-7390/11/10/2320>.

SHAH, B. “Large Learning Models: The Rising Demand of Specialized LLM’s”. 2024. Available at: <https://blogs.infosys.com/emerging-technology-solutions/artificial-intelligence/large-learning-models-the-rising-demand-of-specialized-llms.html>.

MEENA, S. “The Future of Large Language Models: Evolution, Specialization, and Market Dynamics”. 2023. Available at: <https://www.linkedin.com/pulse/future-large-language-models-evolution-specialization-shekhar-meena/>.

GHOSH, B. “Emerging Trends in LLM Architecture,”. 2023. Available at: <https://medium.com/@bijit211987/emerging-trends-in-llm-architecture-a8897d9d987b>.

LANGCHAIN. “Self-query Retriever”. 2023. Available at: <https://python.langchain.com/docs/modules/data_connection/retrievers/self_query/>.

LEVENSHTEIN, V. “Binary codes capable of correcting deletions, insertions, and reversals”, *Cybernetics and Control Theory*, v. 10, 1966.

SHINN, N., CASSANO, F., BERMAN, E., et al. “Reflexion: Language Agents with Verbal Reinforcement Learning”, 3 2023. Available at: <http://arxiv.org/abs/2303.11366>.

PAL, S., BHATTACHARYA, M., LEE, S. S., et al. “A Domain-Specific Next-Generation Large Language Model (LLM) or ChatGPT is Required for Biomedical Engineering and Research”. 3 2024. ISSN: 15739686.

AREFEEN, A., DEBNATH, B., CHAKRADHAR, S. "LeanContext: Cost-efficient domain-specific question answering using LLMs", *Natural Language Processing Journal*, v. 7, pp. 100065, 2024. doi: 10.1016/j.nlp.2024.100065. Available at: <https://doi.org/10.1016/j.nlp.2024.100065>.

Appendix A

Experiment 2

A.1 Code for LLM-as-a-Judge

```
class Confusion_Matrix(TypedDict):  # type: ignore
    true_positive: list[str]
    false_positive: list[str]
    true_negative: list[str]
    false_negative: list[str]

def calculate_metrics(llm, question, history):
    if type(question['Ground Truth']) == str:
        prompt_confusion_matrix = f"""
            Voce recebera os seguintes parametros:
            Pergunta: a pergunta do usuario
            Resposta Ideal: a resposta considerada correta por um humano
            Resposta do sistema: a resposta fornecida pelo sistema baseado em IA

            Pegue a resposta do sistema, separe em afirmacoes e classifique cada afirmacao entre as opcoes abaixo:
            True Positive (TP): as afirmacoes corretas feitas pelo sistema, ou seja, que estao presentes na resposta ideal.
            False Positive (FP): as afirmacoes incorretas ou irrelevantes feitas pelo sistema, ou seja, que nao estao presentes na resposta ideal.

            Pegue a resposta ideal, separe em afirmacoes e classifique cada afirmacao entre as opcoes abaixo:
```

```
20              True Negative (TN): Nao se aplica, deixar vazio.
21              False Negative (FN): as afirmacoes que constam na
     resposta ideal, mas nao foram feitas pelo sistema.
22
23              Importante:
24              - Voce deve gerar listas de afirmacoes para cada
    categoria.
25              - Voce deve quebrar as respostas do sistema e a
    ideal em afirmacoes objetivas.
26              - Se as respostas do sistema ou a ideal
    contiverem frases grandes com mtas afirmacoes, analisar
    cada afirmacao separadamente.
27              - Uma afirmacao nao pode estar em mais de uma
    categoria.
28              - Ignore coisas na resposta do sistema que nao
    sao afirmacoes objetivas, como por exemplo citacoes de
    fontes e links.
29
30              Vamos la!
31
32              ###################
33
34              Pergunta: {{{{
35              {question['Question']}
36              }}}}
37
38              Resposta ideal: {{{{
39              {question['Ground Truth']}
40              }}}}
41
42              Resposta do sistema: {{{{
43              {history[-1].content}
44              }}}}
45          """
46          response = llm.with_structured_output(
    Confusion_Matrix).invoke(prompt_confusion_matrix)
47
48          true_positive_count = len(response.get('true_positive
    ', []))
49          false_positive_count = len(response.get('
    false_positive', []))
```

```
50          true_negative_count = len(response.get('true_negative
    ', []))
51          false_negative_count = len(response.get('
    false_negative', []))
52
53          try:
54              precision = true_positive_count / (
    true_positive_count + false_positive_count)
55          except ZeroDivisionError:
56              precision = 0
57
58          try:
59              recall = true_positive_count / (
    true_positive_count + false_negative_count)
60          except ZeroDivisionError:
61              recall = 0
62
63          try:
64              f1_score = 2 * (precision * recall) / (precision
    + recall)
65          except ZeroDivisionError:
66              f1_score = 0
67
68          print("\\n\\nPrecision: "+str(precision))
69          print("Recall: "+str(recall))
70          print("F1 Score: "+str(f1_score))
71
72          print("\\nAnswer Size Ratio: "+str(question['Answer
    Size (% of GT)']))
73
74          state.precision = precision
75          state.recall = recall
76          state.f1_score = f1_score
77          state.ground_truth = question['Ground Truth']
78          state.statements = response
```

Listing A.1: Código para LLM-as-a-Judge

A.2 Results

A.2.1 Precision

Best Precision by Model and Configuration

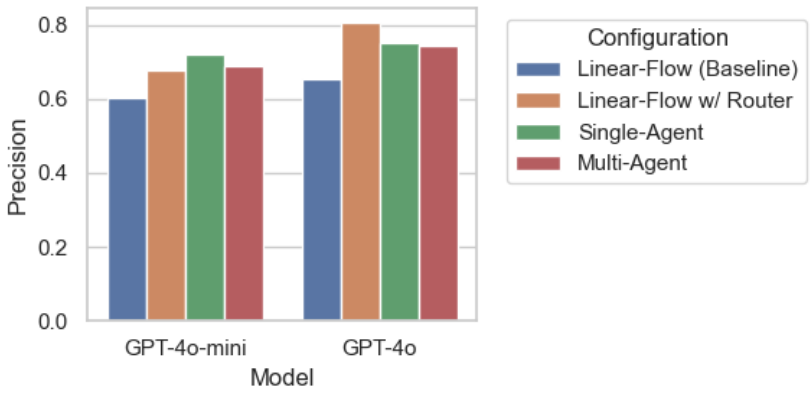

Figure A.1: Best precision by model and configuration.

Best Precision by Question Index and Configuration

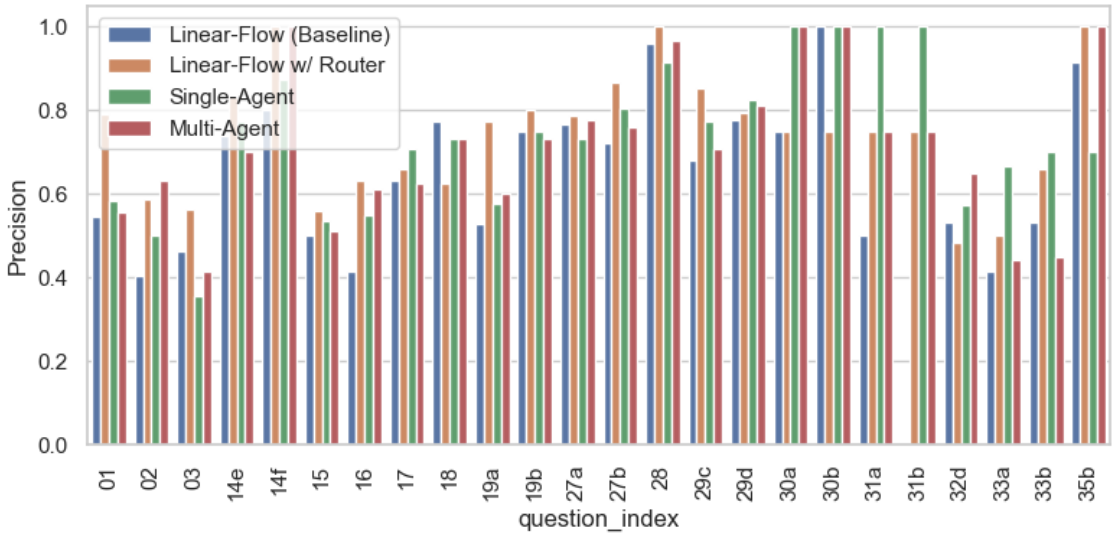

Figure A.2: Best precision by question index and configuration.

Best Precision by Question Index and Model

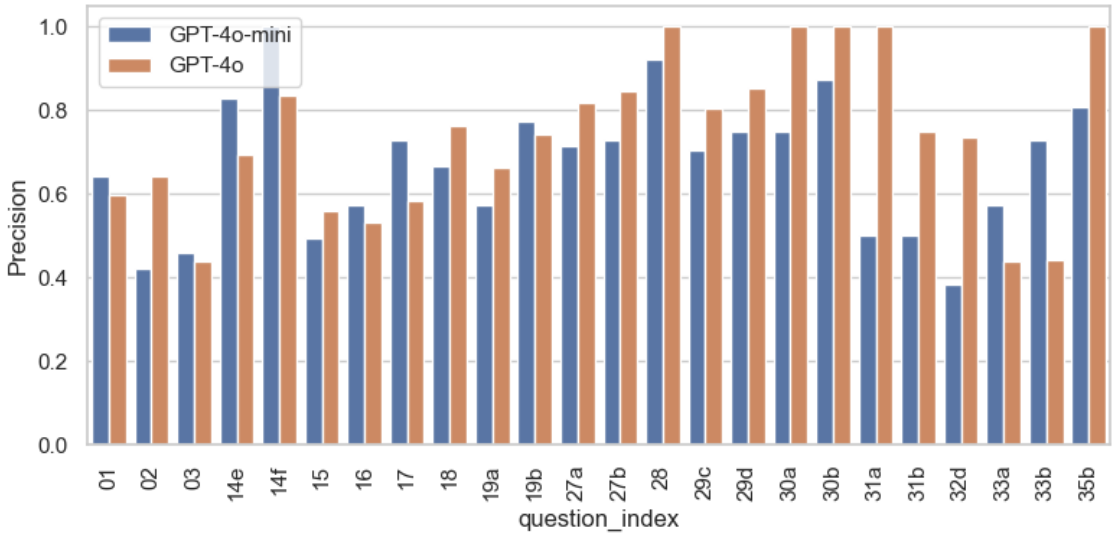

Figure A.3: Best precision by question index and model.

Facet Histogram of Precision by Model

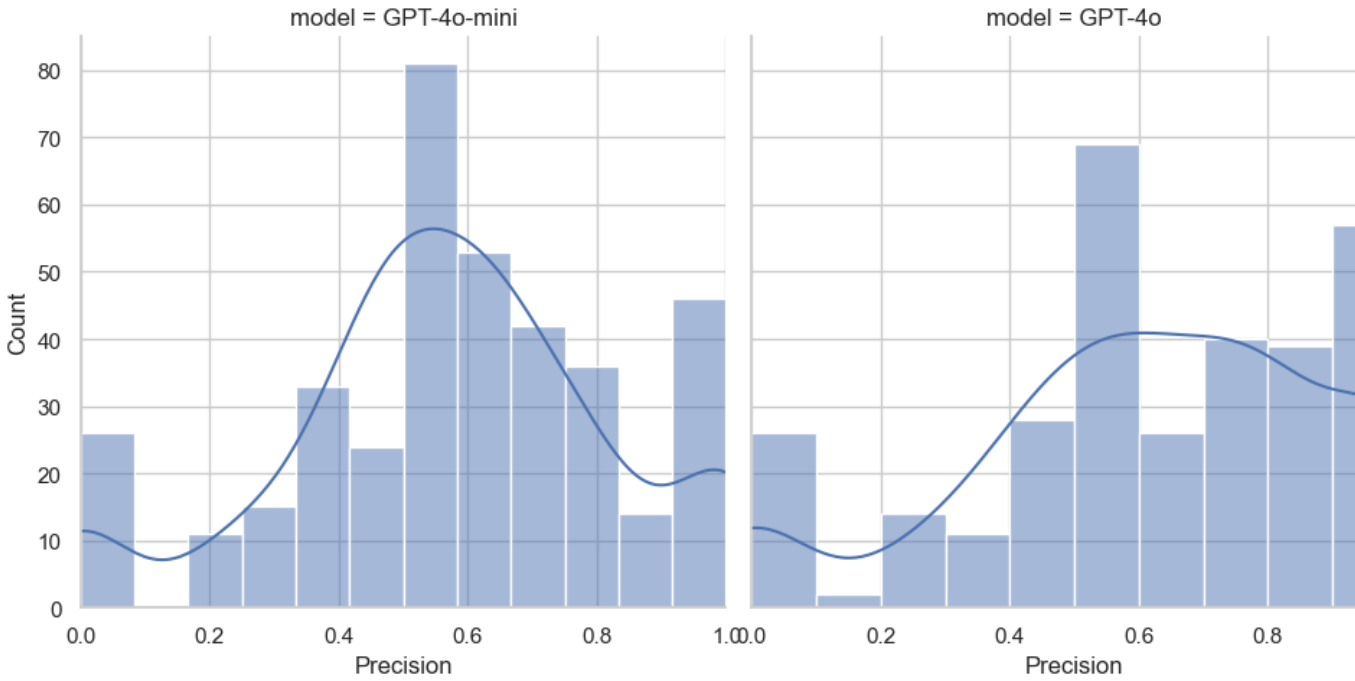

Figure A.4: Facet histogram of precision by model.

Facet Histogram of Precision by Model (best of 3)

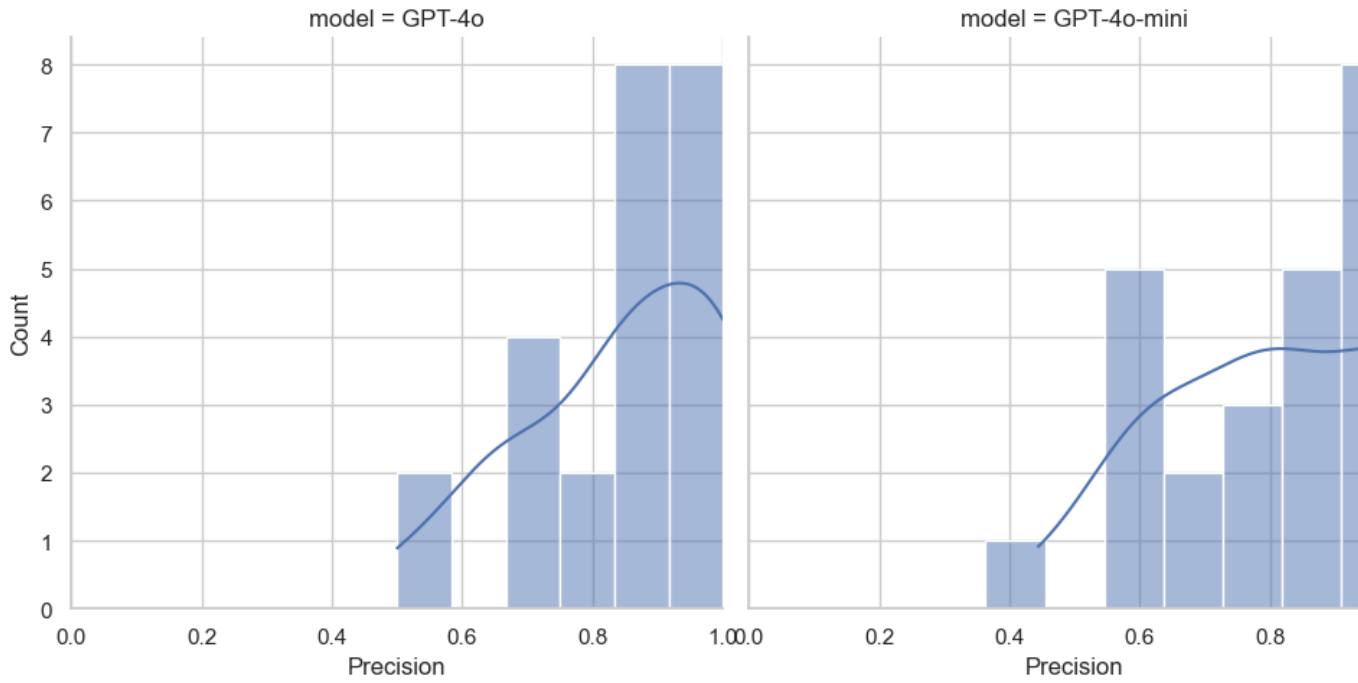

Figure A.5: Facet histogram of precision by model (best of 3).

Histogram of All Precisions

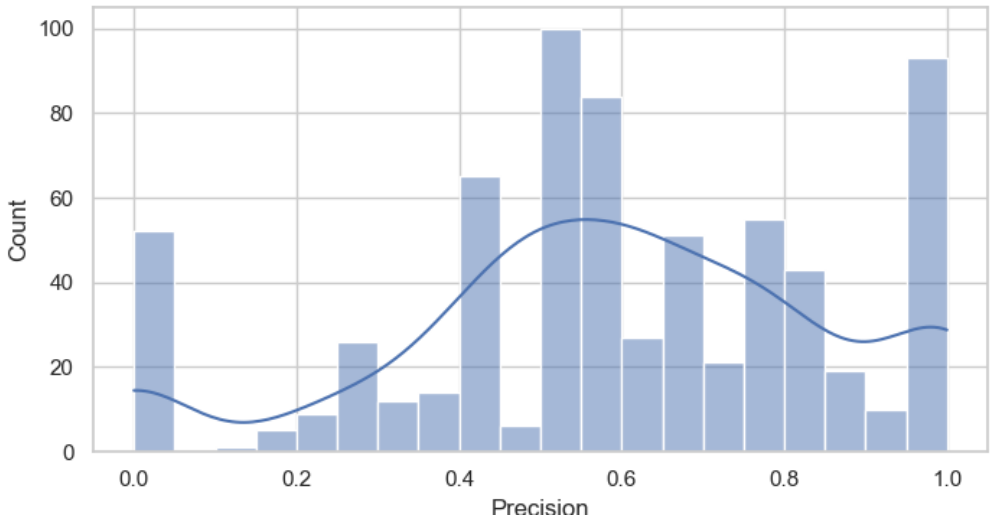

Figure A.6: Histogram of all precisions.

Line Plot of Precision by Question Index and Model

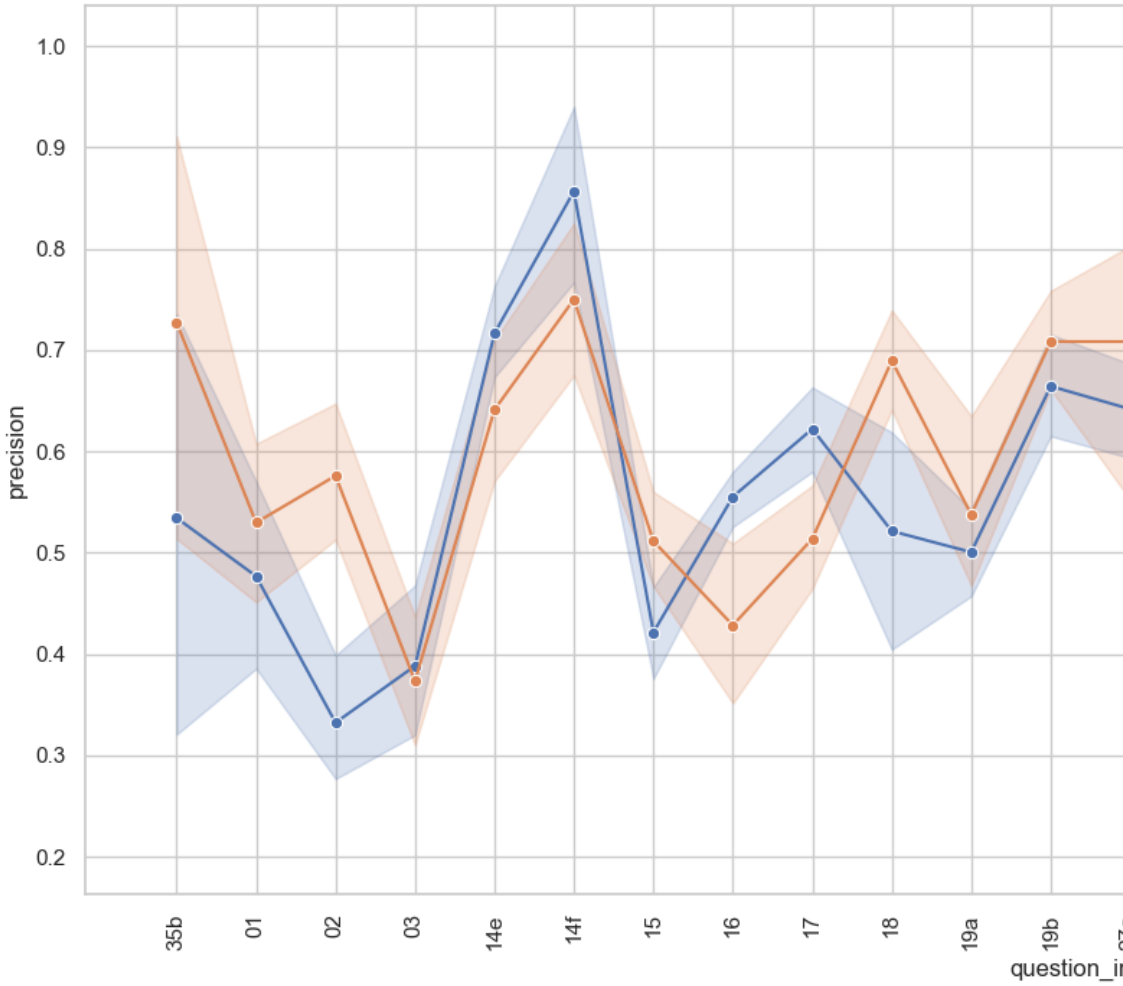

Figure A.7: Line plot of precision by question index and model.

Boxplot of Precision by Model and Configuration

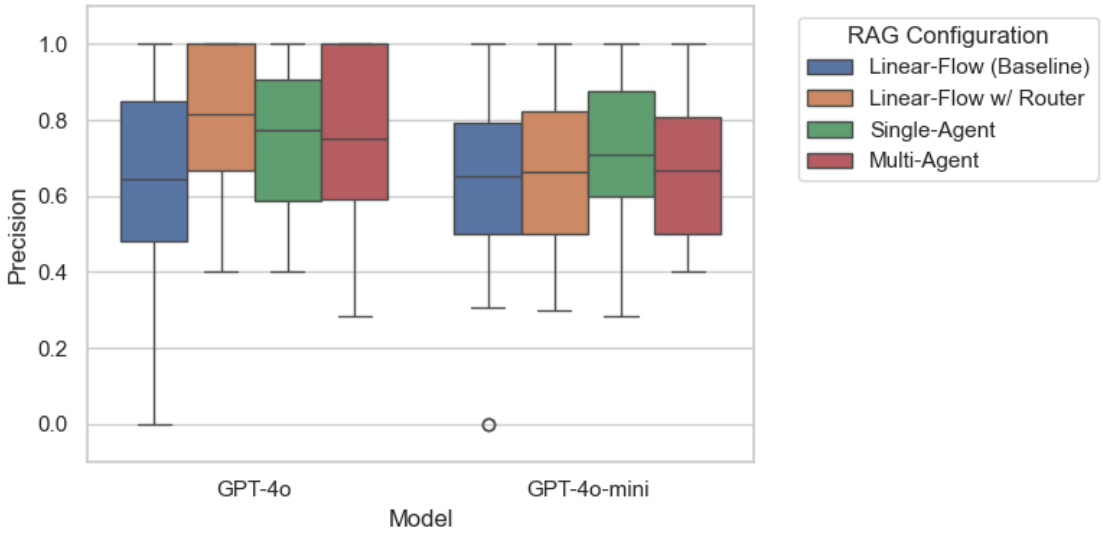

Figure A.8: Boxplot of precision by model and configuration.

Precision by Model

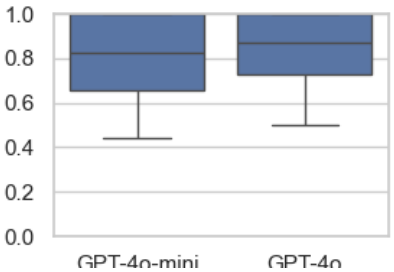

Figure A.9: Precision by model.

Precision by Model and Configuration

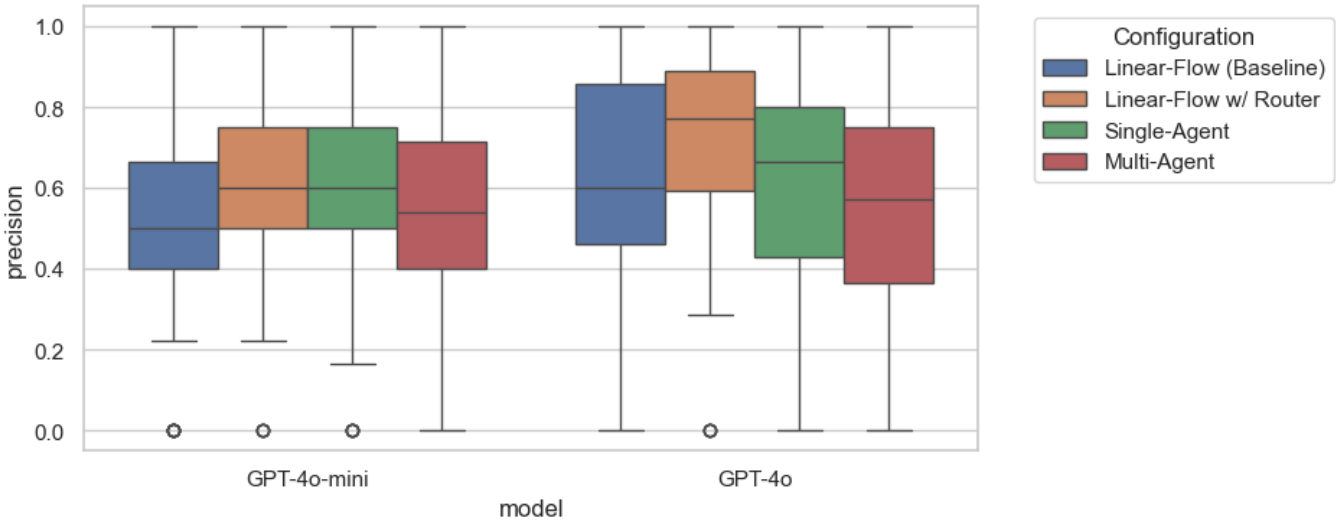

Figure A.10: Precision by model and configuration.

Line Plot of Precision by Question Index and Configuration

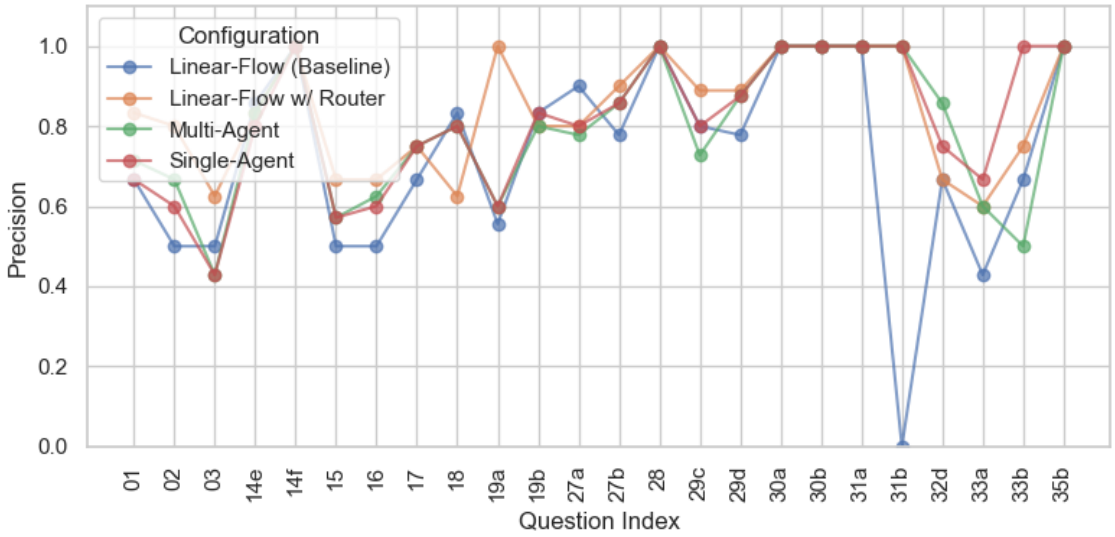

Figure A.11: Line plot of precision by question index and configuration.

Scatter Plot of Precision vs. Total Time

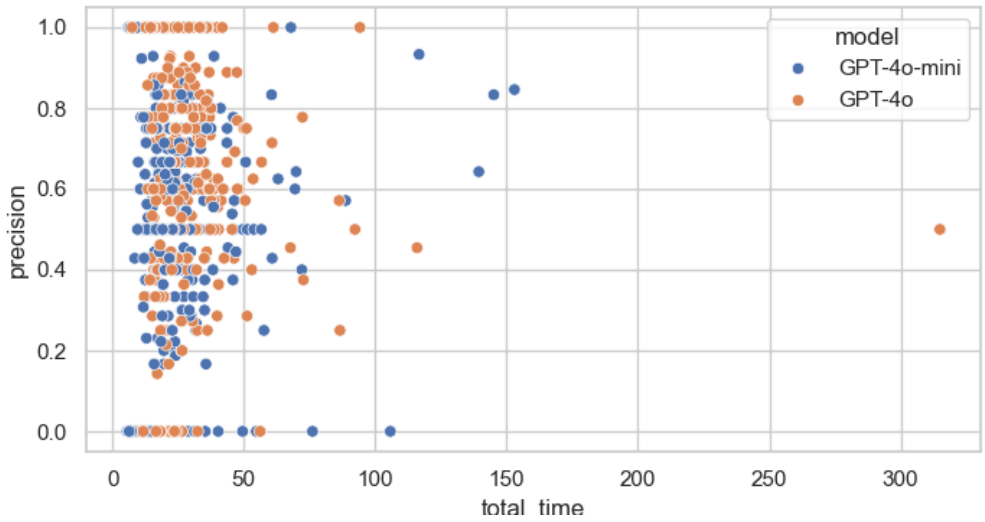

Figure A.12: Scatter plot of precision vs. total time.

Scatter Plot of Precision vs. Total Token Count Input

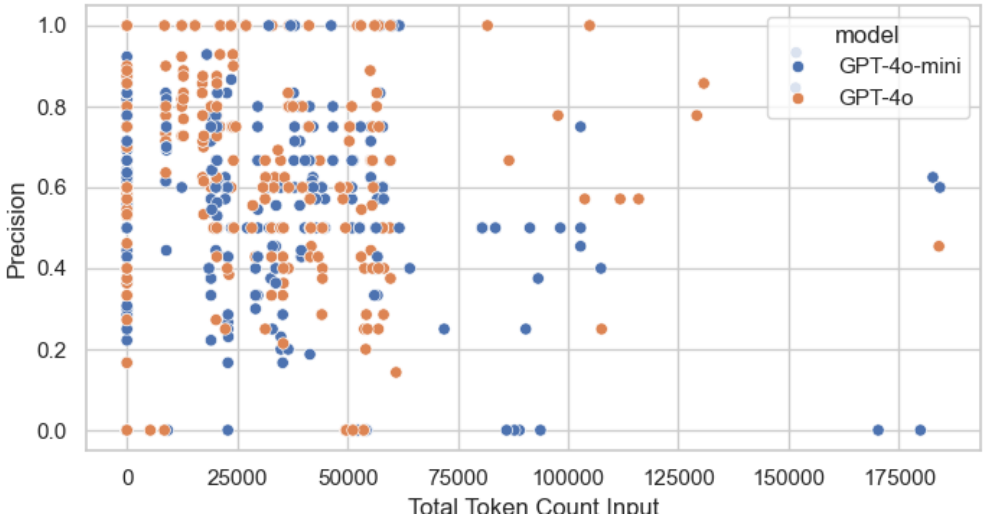

Figure A.13: Scatter plot of precision vs. total token count input.

Swarm Plot of Precision by Model and Configuration

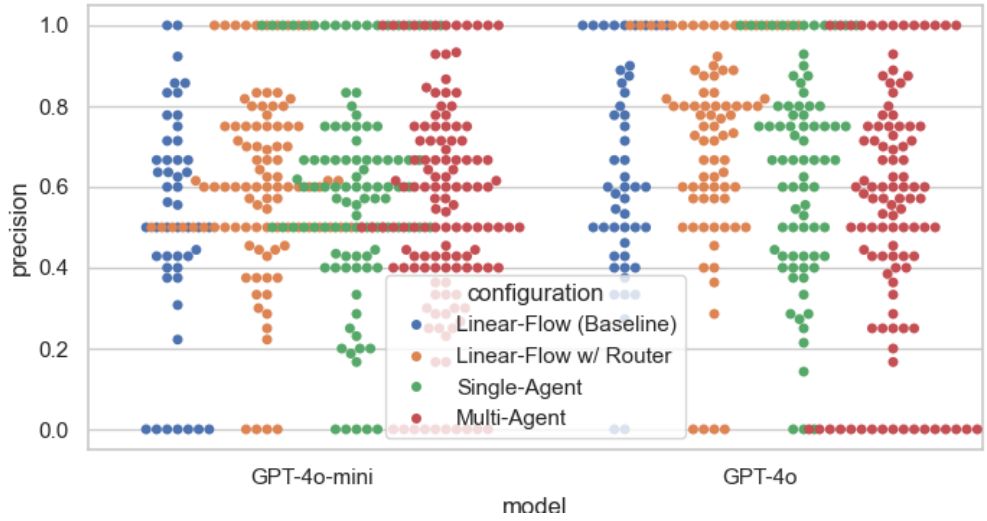

Figure A.14: Swarm plot of precision by model and configuration.

Violin Plot of Precision by Model and Configuration

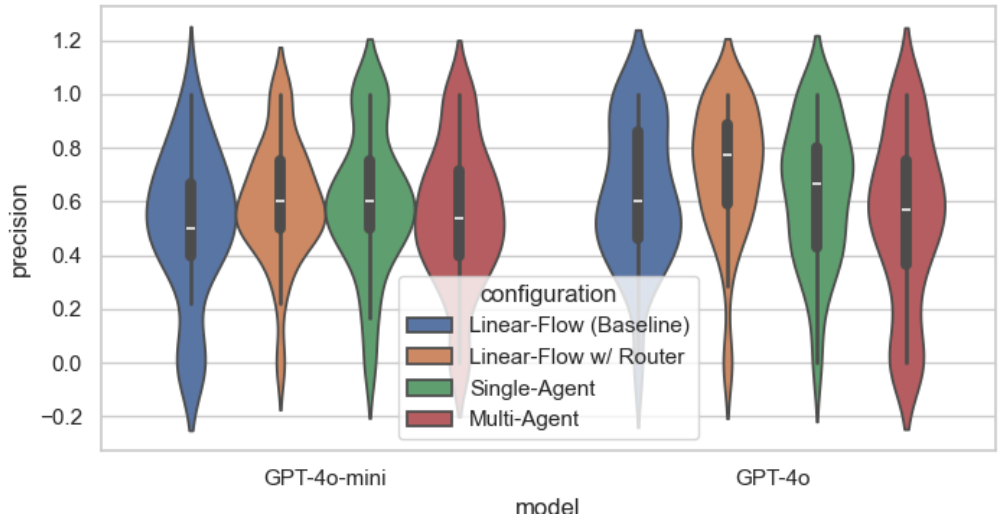

Figure A.15: Violin plot of precision by model and configuration.

A.2.2 Recall

TEXT

...

...
TEXT
...
...