Carlos Vinicios Martins Rocha

A Data Annotation Approach Using Large Language Models

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Informática, do Departamento de Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor : Prof. Hélio Côrtes Vieira Lopes
Co-advisor: Prof. Jonatas dos Santos Grosman

Rio de Janeiro
September 2024

Carlos Vinicios Martins Rocha

A Data Annotation Approach Using Large Language Models

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Rio de Janeiro, September 26th, 2024

Carlos Vinicios Martins Rocha

Graduated in Computer Science by the Universidade Federal do Maranhão.

Bibliographic data

Rocha, Carlos Vinicios Martins

A Data Annotation Approach Using Large Language Models / Carlos Vinicios Martins Rocha; advisor: Hélio Côrtes Vieira Lopes; co-advisor: Jonatas dos Santos Grosman. – 2024.

84 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2024.

Inclui bibliografia

1. Informática – Teses. 2. Anotação de Dados. 3. LLM. 4. VQA. 5. Documentos. I. Lopes, Hélio Côrtes Vieira. II. Grosman, Jonatas dos Santos. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Acknowledgments

To God, who has blessed and protected me from all harm and illness throughout my life and has been the strength I needed to persevere and keep trying, regardless of the difficulties.

To my parents, who have always supported me on my academic journey, shielding me from life's challenges that could have hindered my educational and professional progress. They were also the primary financial backers of my entire journey, sacrificing from the little they had to provide me with a quality education and continuous improvement opportunities. To my siblings, who stood by me during tough times and were always there when I needed them.

To my great love, Laryssa, who has been by my side since my undergraduate studies, witnessing my entire academic and personal growth, and making every special moment even more meaningful. I am immensely grateful for your companionship, especially when physical distance separated us, yet you always found a way for us to spend even a few precious minutes together. I am also thankful that you shared the dream of pursuing postgraduate studies with me, far from our hometown, as we lived and shared our days together. I appreciate your love, always thinking of different ways to make every moment as special as possible. And when things were tough, you were by my side, offering emotional support when I needed it the most.

To my friends, Arthur, Pedro, and Boaro, who first ventured into the unknown lands where I now find myself, making the process of settling and adapting much easier. To my friend Venicius, who is practically a brother, accompanying me on this long journey and being there from the first day to the last of my postgraduate studies. To my undergraduate friends, who, in some way, helped or encouraged me to pursue a master's degree. To my high school friends ("*Panelinha*") and university friends ("*Amifos*" and "*Sociedade do Café*"), who have been with me for a long time, sharing moments of joy and achievement, and who also directly contributed to the development of my work.

To my advisor, who supported me throughout the development of my work and gave me the opportunity to join the ExACTa research lab, allowing for better academic and professional growth. To my co-advisors and friends, who embraced my project, helping and guiding me through its technical and theoretical development.

To the Department of Informatics at Pontifical Catholic University of Rio de Janeiro (PUC-Rio), for granting the scholarship that enabled me to pursue postgraduate studies.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Rocha, Carlos Vinicios Martins; Lopes, Hélio Côrtes Vieira (Advisor); Grosman, Jonatas dos Santos (Co-Advisor). **A Data Annotation Approach Using Large Language Models**. Rio de Janeiro, 2024. 84p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Documents are essential for the economic and academic system; however, exploring them can be complex and time-consuming. An approach to surpass this problem is the use of Visual Question and Answering (VQA) models to extract information from documents through natural language prompts. In VQA, as well as for the development of various models, it is necessary to have annotated data for training and validation. However, creating these datasets is challenging due to the high cost involved in the process. To face this challenge, we propose a four-step process that combines Computer Vision Models and Large Language Models (LLMs) for VQA data annotation in financial reports. The proposed method starts with recognizing the textual structure of documents through Document Layout Analysis and Table Structure Extraction models. Then, it uses two distinct LLMs for the generation and evaluation of question and answer pairs, automating the construction and selection of the best pairs to compose the final dataset. To evaluate the proposed method, we generate a dataset for train and evaluate VQA specialized models.

Keywords

Data Annotation; LLM; VQA; Documents.

Resumo

Rocha, Carlos Vinicios Martins; Lopes, Hélio Côrtes Vieira; Grosman, Jonatas dos Santos. **Uma Abordagem para Anotação de Dados Utilizando Grandes Modelos de Linguagem**. Rio de Janeiro, 2024. 84p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Os documentos são essenciais para o sistema econômico e acadêmico; no entanto, explorá-los pode ser uma tarefa complexa e demorada. Uma abordagem para contornar esse problema é o uso de modelos de *Visual Question and Answering* (VQA) para extração de informações de documentos por meio de *prompts* em linguagem natural. No VQA, assim como para o desenvolvimento dos mais variados modelos, é necessário possuir dados anotados para a sua etapa de treinamento e validação. No entanto, criar esses conjuntos de dados é desafiador devido ao alto custo envolvido no processo. Com base nisso, propomos um processo de quatro etapas que combina Modelos de Visão Computacional e *Large Language Models* (LLMs) para a anotação de dados de VQA em relatórios financeiros. O método proposto inicia pelo reconhecimento da estrutura textual dos documentos por meio de modelos de Análise de Layout de Documentos e Extração de Estrutura de Tabelas. Em seguida, utiliza duas LLMs distintas para a etapa de geração e avaliação dos pares de perguntas e respostas geradas, automatizando a construção e seleção dos melhores pares para compor a base final. Para avaliar o método proposto, geramos um dataset para treinar e avaliar modelos especialistas em VQA.

Palavras-chave

Anotação de Dados; LLM; VQA; Documentos.

Table of contents

List of figures

List of tables

List of Abreviations

AP – Average Precision

B3 – *Brasil Bolsa Balcão*

CNN – Convolutional Neural Network

CV – Computer Vision

CVM – *Comissão de Valores Mobiliários*

DLA – Document Layout Analysis

DocVQA – Document Visual Question and Answer

ESMA – European Securities and Markets Authority

GPT – Generative Pre-Trained Transformer

LGPMA – Local and Global Pyramid Mask Alignment

LLaMA – Large Language Model Meta AI

LLM – Large Language Model

mAP – Mean Average Precision

NLG – Natural Language Generation

NLP – Natural Language Process

NLU – Natural Language Understanding

OCR – Optical Character Recognition

QA – Question and Answer

RNN – Recurrent Neural Network

SEC – Securities and Exchange Commission

TATR – Table Transformer

TEDS – Tree-Edit-Distance-based Similarity

TSR – Table Structure Recognition

VQA – Visual Question and Answer

YOLO – You Only Look Once

"I did not choose this path. It was chosen for me."

Ezio Auditore, *Assassin's Creed II.*

1
Introduction

Data annotation is a critical task in developing supervised machine learning models, especially in Natural Language Processing (NLP) and Computer Vision (CV). The quality of these models relies on large volumes of labeled data to learn and perform complex tasks, such as document comprehension and interpretation.

Document Visual Question Answering (DocVQA) exemplifies a task where state-of-the-art deep learning models combine NLP and CV techniques. The primary goal of DocVQA is to enable models to interpret document images by leveraging both textual and visual features, allowing users to interact with document information through natural language queries.

Training DocVQA models requires thousands of annotated and carefully reviewed document pages. This annotation and review process is time-consuming and costly, primarily due to the significant human resources needed. To meet this demand, the literature often employs non-expert annotators through crowdsourcing platforms like Amazon Mechanical Turk (MATHEW et al., 2021; MATHEW et al., 2022). However, using such platforms presents significant challenges and limitations, including inconsistent annotation quality and the additional effort needed for review and validation, which increases both cost and time (KITTUR; CHI; SUH, 2008).

DocVQA models have applications across various domains. In scientific research, they assist in identifying relationships between entities within large text corpora. In the insurance industry, these models help understand complex policies and detect fraud. In finance, they extract metrics from financial reports and provide insights for investors based on text-based market data.

A valuable data source for DocVQA is financial reports, such as those provided by Brazilian stock market, named *Brasil Bolsa Balcão* (B3).[1] These reports are essential for publicly traded companies, ensuring the transparency and credibility required to attract investors. Accurate financial performance assessments and informed decision-making depend on these documents. In the B3, 2,706 listed companies are required by the Securities and Exchange Commission of Brazil, or Comissão de Valores Mobiliários (CVM),[2] to publish annual balance sheets, quarterly reports, and other periodic information. As a result, a vast amount of publicly available financial reports support market

[1]https://www.b3.com.br/
[2]https://www.gov.br/cvm/en

analysis. This scenario is not unique to Brazil; similar regulations exist in many other regions, such as in the United States, where the Securities and Exchange Commission (SEC)[3] oversees financial reporting, and in the European Union, where the European Securities and Markets Authority (ESMA).[4]

Although these financial reports are public, they lack structured annotated data. To create a usable dataset, it is necessary to first address the data annotation challenge. One approach to leverage the abundant unlabeled data is to use Large Language Models (LLMs) as annotators to generate labels in a zero-shot or few-shot manner. While promising, this method introduces noise into the generated labels, especially when dealing with complex tasks and domain-specific datasets (AGRAWAL et al., 2022).

Wang et al. (2021) demonstrated that data annotation with GPT-3 can reduce costs by more than 90% for Natural Language Generation (NLG) and Natural Language Understanding (NLU) tasks. In the financial domain, the work of Aguda et al. (2024) demonstrated that more robust language models like GPT-4 and PaLM 2 outperform crowdworkers in annotating relationships between entities and individuals in financial documents. From a multi-modal perspective, Nguyen et al. (2023) combined a Table Structure Recognition (TSR) model and a language model to generate responses in a Table QA task. For the automatic validation of answers generated by foundational models to open questions, Bai et al. (2024) proposed peer-evaluation method, combining LLMs to assess the quality of the responses.

1.1 Research Goal

Given their remarkable capability on many text-annotation tasks with a reduced cost, LLMs could be addressed by automatically generating an annotated dataset of questions and answers for the DocVQA task.

To take advantage of the generalist capability of LLMs and reduce the time it takes to build an annotated dataset for training expert models in DocVQA, our main research question is:

> ***MRQ***: *How can we use Large Language Models in data annotation for Document Visual Question Answering task?*

Moreover, the approach proposed to solve the *MRQ* must consider the different ways of displaying data in documents, such as tables. Furthermore, the proposed approach must take into account the need for a textual representation

[3]https://www.sec.gov/
[4]https://www.esma.europa.eu/

originating from these representations. Finally, the questions and answers generated by the approach must be reliable and of high quality to ensure that the specialist models learn from them. To achieve this, we address the following research questions:

> ***RQ1****: How can we combine computer vision models and Large Language Models to generate questions and answers from documents?*
> ***RQ2****: How can we evaluate the quality of the generated questions and answers?*

To answer *RQ1* and *RQ2*, we propose a three-stage process. In the first stage, we transcribe the document page by recognizing the characters and the main points of the layout and converting the tables into markup language. In the second stage, we organize and structure all transcriptions and segmentations from the previous stage into a text, which is input into an LLM to generate QA pairs. In the third and final stage, the same structured text is provided to another LLM to assess the quality of the generated pairs, determining whether each pair should be included in the final dataset.

As a result of the transcription stage, we selected the RoDLA model, which achieved an overall mAP of 0.716 on the DLA benchmark dataset. For table structure extraction, we selected the TATR model, which attained a TEDS-Struct score of 0.94 on the benchmark dataset. In the generation stage, we observed significant differences in the variability of QA pairs proposed by each LLM. The best-performing models were Mixtral-8x22b and GPT-4o mini, which demonstrated good question variation, adherence to output formatting, and effective utilization of generated QA pairs. Finally, for question quality evaluation, the models delivered strong performance, with Claude 3.5 Sonnet achieving an F1 score of 0.93 when compared to human validation. However, improvements are still needed in this stage to enhance the filtering of false negatives.

1.2 Contributions

- We present a new approach for generating and evaluating question-answer pairs in documents containing both text and tables.
- We contribute by evaluating different models for Document Layout Analysis and Table Structure Recognition, specifically for extracting information from financial reports of the Brazilian stock exchange.

1.3 Overview

This work is organized as follows: Chapter 2 reviews the related literature that served as a reference for this work. Chapter 3 explains all the concepts and techniques necessary to understand the developed approach. Chapter 4 describes the approach proposed in this work, detailing each stage, from textual data extraction to the evaluation of each pair of questions and answers. Chapter 5 presents the results obtained with the developed approach, discussing the results obtained for each stage. Lastly, Chapter 6 provides a comprehensive overview of the research, evaluates its outcomes, identifies potential shortcomings, and proposes directions for future research.

2
Related Work

In this section, we review the main works related to the generation of questions and answers in documents and the use of large language models (LLMs) for data annotation. We explore approaches that propose the manual creation of datasets, as in (MATHEW et al., 2021), as well as solutions that employ automation to reduce the cost and time of annotation, as in (DING et al., 2023). We also discuss the use of LLMs for automatic annotation generation and evaluation tasks, highlighting studies that evaluate their efficiency and comparing commercial and open source approaches. This review contextualizes the development of our work, which seeks to improve the automatic generation of questions and the evaluation of answers in complex documents, such as financial reports.

2.1
DocVQA Dataset Construction

As with any supervised machine learning task, DocVQA relies on a labeled dataset to train models effectively. High-quality labeled data is essential for teaching models to understand and interpret the diverse textual, structural, and graphical elements found in document images.

Mathew et al. (2021) introduced a dataset consisting of document images from various industries, including elements such as tables, forms, figures, and a range of textual, graphical, and structural components. The dataset was curated from handpicked document pages sourced from the UCSF Industry Document Library and contains 50,000 questions across 12,767 images.

The dataset was annotated through crowdsourcing, using a three-stage process. In the first stage, annotators were tasked with generating up to 10 question-answer pairs based solely on the information in a given document page. The second stage involved validating these pairs, where annotators answered questions generated in the first stage. In the final stage, the authors reviewed the pairs to catch errors missed in earlier steps.

Our approach shares similarities with Mathew et al. (2021) in the annotation process, particularly in the first stage. However, we reduced the number of questions generated per page and modified the validation approach in the second stage, ultimately eliminating the final human review.

Given the extensive human and financial resources required to build a dataset like Mathew et al. (2021), alternative cost-saving strategies have been explored. Ding et al. (2023) proposed an automated question-answer generation process to minimize human annotation efforts and diversify question patterns.

Their automated process focused on the structural elements of documents, such as table counts or semantic information in specific regions. They defined 66 patterns for generating questions for tasks like counting objects, checking their existence, and understanding structural and relational information.

We build on the idea of automated question generation from Ding et al. (2023), but make the process more dynamic. Instead of relying on fixed patterns for controlled scenarios, we leverage the few-shot learning capabilities of LLMs.

2.2 Large Language Model Annotation

Several studies have examined LLMs' potential in data annotation for natural language processing tasks, aiming to identify time-efficient and cost-effective approaches (WANG et al., 2021; AGUDA et al., 2024; ZHANG et al., 2023).

Wang et al. (2021) conducted experiments evaluating GPT-3's performance compared to humans in natural language generation (NLG) and understanding (NLU) tasks, including question generation based on a given text and target answer. They found that their approach reduced labeling costs by up to ten times while achieving human-like performance.

Similarly, Aguda et al. (2024) explored LLM effectiveness in extracting financial relationships, comparing models like GPT-4, PaLM2, and MPT Instruct with human and expert annotations. They evaluated aspects such as time, cost, and reliability, showing that automated annotations were more efficient, though still falling short of expert-level quality, suggesting that human oversight remains necessary for accuracy.

Our work aligns with Wang et al. (2021) in applying LLMs to an NLG task. However, unlike Aguda et al. (2024), which focuses on financial relationship extraction, we concentrate on financial reports, utilizing LLMs for QA annotation.

Financial reports often include complex tables, making it necessary to extract and represent the information in a textual format for effective LLM annotation. Nguyen et al. (2023) proposed a pipeline for answering questions from business table images, extracting the table structure into HTML, converting it to a dataframe, and using it as input for a QA module.

Despite their capabilities, LLMs can produce hallucinations, so assessing the validity and quality of generated questions is essential. Bai et al. (2024) introduced a benchmarking framework where language models both generate and evaluate questions to reduce test leakage and improve automated assessment in open questions. They tested this on several foundation models, including LLaMA and GPT-4, across various domains. Additionally, they proposed a decentralized peer-evaluation method to reduce biases. The results indicated that larger fine-tuned models, such as LLaMA-65B and GPT-4, performed better on complex questions, and decentralized evaluation offered a more balanced process.

We adopt the idea from Bai et al. (2024), using an LLM to automatically evaluate the annotations of others, serving as a filter for invalid responses. This step is critical for automating the entire process and catching simple errors without the need for human intervention.

3
Background

This chapter presents an overview of the core concepts to understand this work better. It begins by explaining the Visual Question and Answering (VQA) task. Next, it presents the core concepts of Optical Character Recognition (OCR), Document Layout Analysis (DLA) and Table Structure Recognition (TSR) tasks. Lastly, it explains the concepts and processes related to the LLMs and the generic framework used to invoke different models.

3.1
Visual Question and Answering (VQA)

The VQA process can be broken down into two main tasks: the first is to extract and understand the visual information from an image, and the second is to comprehend and process the textual question posed by the user. These tasks are typically achieved using a combination of Convolutional Neural Network (CNN) for image analysis and Recurrent Neural Network (RNN) or Transformers for text processing (WU et al., 2017).

For the first task, VQA solutions employ two components, such as object detectors and scene classifiers. Object detectors help identify and locate objects within an image, while scene classifiers provide contextual information about the overall setting. For the second task, a language model interprets the question, which guides the system to focus on relevant parts of the image and synthesize an appropriate answer.

Figure 3.1 depicts the VQA workflow. The elephant represents the input image. The question is a textual input that queries specific information about the image. The VQA system processes both images and questions to produce an answer. The system combines features extracted from the image and the question to generate an accurate response.

VQA systems find applications in various fields, such as assistive technologies for visually impaired individuals, educational tools, image search engines, and interactive AI systems in video games and virtual environments. In addition to these applications, VQA has also been extended to the context of Document Analysis and Recognition (DAR), where systems are developed to automatically extract information presented on documents and addressed to human comprehension (MARINAI, 2008).

This is particularly useful in scenarios like document retrieval and automated document processing, as seen in the DocVQA challenge (MATHEW

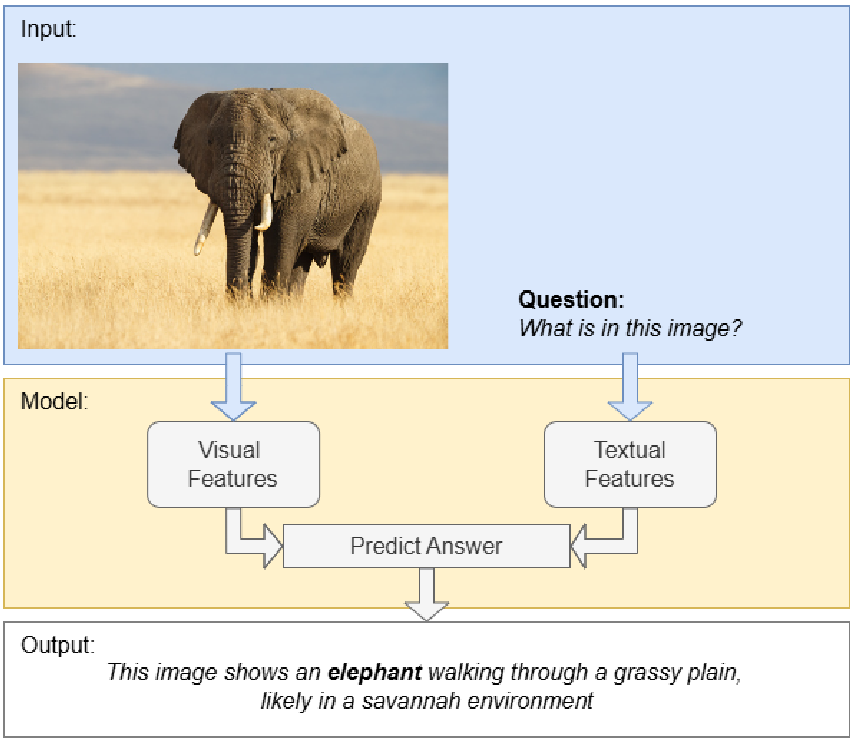

Input:
Question:
What is in this image?
Model:
Visual Features
Textual Features
Predict Answer
Output:
This image shows an **elephant** walking through a grassy plain, likely in a savannah environment

Figure 3.1: Visual Question and Answer Workflow.

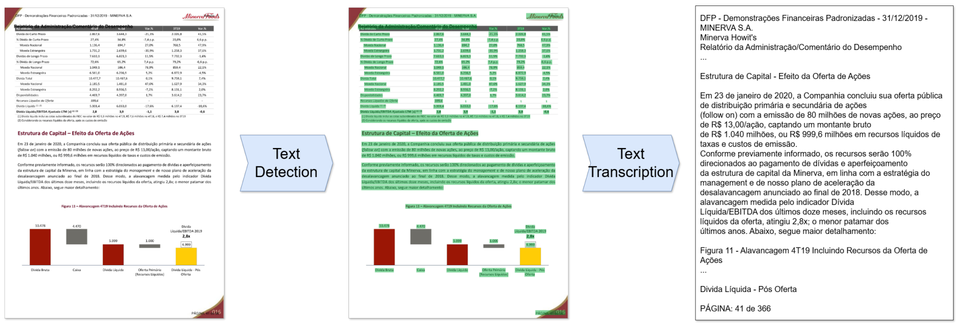
et al., 2021). These systems leverage similar methodologies to traditional VQA but are adapted to handle the specific challenges of understanding and answering questions based on text-rich documents.

3.2 Optical Character Recognition (OCR)

OCR is an essential procedure in the VQA process that converts documents such as scanned paper, PDF files, or images into editable and searchable data. Essentially, OCR systems extract features, such as characters' shapes and sequences, from a document and translate them into a machine-readable text format. Such systems typically employ a two-step solution: text detection followed by text transcription to effectively recognize and transcribe text in images. These steps often operate independently, using different models and strategies.

Text detection focuses on identifying the text within an image, which is represented as a three-dimensional tensor $C \times H \times W$, where C denotes the number of color channels, H the height, and W the width of the image. The challenge in text detection arises from the diverse shapes, orientations, and potential text distortions. Researchers commonly approach text detection either as an object detection task — where the model learns to generate bounding box coordinates around text — or as an instance segmentation task, where a mask is generated to differentiate text pixels from non-text pixels (SUBRAMANI et al., 2020).

Once the text is detected, the next step is text transcription, which involves converting the detected text within an image into a sequence of characters. This process begins with a cropped image of the text, usually at the character or word level, with dimensions $C \times H' \times W'$. The text transcription model then outputs a sequence of tokens from a predefined vocabulary V, which can be character- or word-based. Character-level transcription allows for flexibility in recognizing any word within the language, as the vocabulary size is limited to the set of possible characters. Conversely, word-level transcription can reduce minor errors but is restricted to the predefined vocabulary, limiting the ability to transcribe out-of-vocabulary words (SUBRAMANI et al., 2020).



DFP - Demonstrações Financeiras Padronizadas - 31/12/2019 -
MINERVA S.A.
Minerva Howit's
Relatório da Administração/Comentário do Desempenho
...

Estrutura de Capital - Efeito da Oferta de Ações

Em 23 de janeiro de 2020, a Companhia concluiu sua oferta pública
de distribuição primária e secundária de ações
(follow on) com a emissão de 80 milhões de novas ações, ao preço
de R$ 13,00/ação, captando um montante bruto
de R$ 1.040 milhões, ou R$ 999,6 milhões em recursos líquidos de
taxas e custos de emissão.
Conforme previamente informado, os recursos serão 100%
direcionados ao pagamento de dívidas e aperfeiçoamento
da estrutura de capital da Minerva, em linha com a estratégia do
management e de nosso plano de aceleração da
desalavancagem anunciado ao final de 2018. Desse modo, a
alavancagem medida pelo indicador Dívida
Líquida/EBITDA dos últimos doze meses, incluindo os recursos
líquidos da oferta, atingiu 2,8x; o menor patamar dos
últimos anos. Abaixo, segue maior detalhamento:

Figura 11 - Alavancagem 4T19 Incluindo Recursos da Oferta de
Ações
...

Divida Líquida - Pós Oferta

PÁGINA: 41 de 366

Figure 3.2: Optical Character Recognition Task Workflow.

As an example, Figure 3.2 illustrates the OCR task workflow. First, the document is processed through an object detection model, which generates bounding boxes around text regions. Subsequently, a transcription model is applied to convert the text within each bounding box into a readable format.

3.3 Document Layout Analysis (DLA)

DLA is an important task in various document understanding applications. It converts semi-structured information into structured data while extracting essential information from documents. This task is challenging due to the diverse layouts and formats found in documents. Often based on conventional rule-based approaches or machine learning techniques, existing methods struggle to generalize effectively because they rely on hand-crafted features that may not be robust to layout variations (CAO; MA; LI, 2021). This way, the objective of the DLA task is to identify and categorize different document components, such as text blocks, images, tables, and headings, to understand its structure and layout (ZHONG; TANG; YEPES, 2019).

The DLA task can be broken into two sub-tasks: the document segmentation into a sequence of elements and the classification and labeling of each element using different features and algorithms (CAO; MA; LI, 2021). These features can include visual characteristics like font size, spacing, and any other graphical cues, as well as textual features like keywords and phrases.

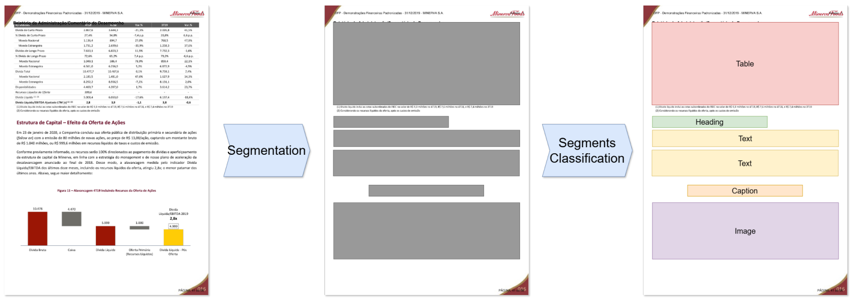


Figure 3.3: Document Layout Analysis Task Workflow.

Figure 3.3 illustrates the DLA workflow. Starting with a document image as input, which includes various elements like text blocks, images, and tables. The segmentation process identifies these elements, and the classification process labels them appropriately. The resulting structured document includes labeled components and segment bounding boxes, which facilitate further processing and analysis.

3.4 Table Structure Recognition

The TSR task involves representing a table in a machine-readable format, where its layout is encoded according to a predefined standard. This representation can be in either a physical or logical format. The logical format includes information about each cell's row and column span, while the physical format also includes the bounding box coordinates of the cells. TSR is a challenging problem due to the complex structures and high variability in table layouts (RAJA; MONDAL; JAWAHAR, 2020).

TSR can be understood in two main aspects: first, it involves detecting table regions and their boundaries within a document, and second, it focuses on recognizing the table's internal structure, such as rows, columns, and cells, as well as their hierarchical relationships (CHI; LIU; XU, 2019).

These processes are typically executed through a combination of image processing and machine learning techniques. TSR models are applicable in a variety of contexts, including the digitization of printed documents, automation of data entry tasks, enhancement of data accessibility, and improvement of document retrieval and management.

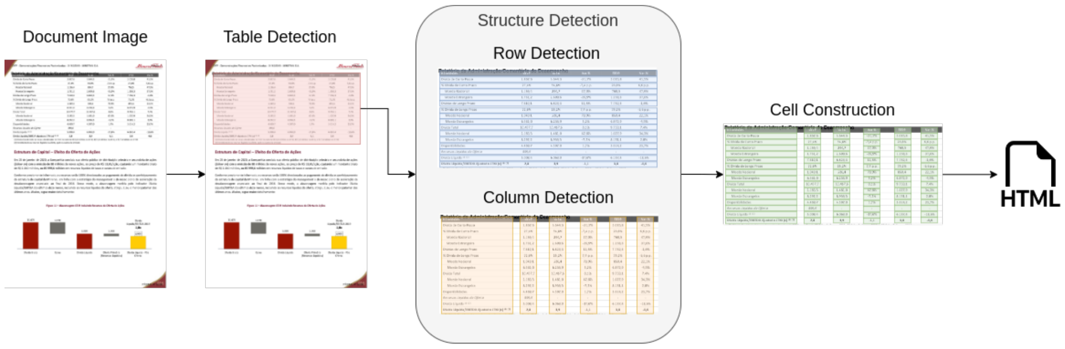


Figure 3.4: Table Structure Recognition Task Workflow.

Figure 3.4 illustrates a common TSR workflow. The workflow starts with a document image containing tables. The table detection process identifies the table's regions, and the structure detection process determines the rows and columns within each table. The cell construction process uses the rows and columns bounding boxes to determine the cell locations. Finally, post-processing is applied to generate the HTML representation of the table based on the cell locations.

Despite being presented in Figure 3.4, not all TSR models present post-processing for converting the table to HTML. Also, TSR models output can be a wide different of markup format, such as XML or LaTeX (GÖBEL et al., 2013; LI et al., 2019). The output format varies depending on the model implementation or according to task needs, given that one format can easily converted to the other.

3.5 Large Language Models

LLMs represent a significant advancement in NLP, as evidenced by their deep neural network architectures with a large number of parameters. These models are trained on massive datasets of unlabeled text, primarily utilizing self-supervised or semi-supervised learning techniques to predict the next word in a sequence based on the context provided (ABDULLAH; MADAIN; JARARWEH, 2022).

The Transformer model inspires the architecture of LLMs (VASWANI et al., 2017), which introduced the concept of an attention mechanism. This mechanism enables the model to assign different weights to various segments of the input text, thereby enhancing its ability to focus on the most relevant information during both the encoding and decoding processes. Unlike traditional linear sequence processing methods, the attention mechanism allows the model to evaluate the significance of different positions in the input, enabling a more nuanced representation of each position through a weighted aggregation of other input positions.

In addition to the attention mechanism, Transformer introduced the concept of a context window. This concept defines the maximum range of information that the model can directly process. A larger context window enables the model to consider a broader context, facilitating in-context learning (DONG et al., 2022) and more informed predictions. Conversely, a smaller context window limits the model's ability to capture long-range dependencies, potentially hindering performance on tasks requiring extensive contextual understanding (LI et al., 2024).

The number of parameters in an LLM is an important aspect of its design, directly correlating with the model's complexity and ability to process data. Models with a higher parameter count, such as those with 70 billion parameters, are capable of learning intricate relationships between words and phrases in the training data. However, this increased complexity also comes with greater computational demands for both training and deployment (DESAI, 2024).

Despite their capabilities, LLMs are not without limitations. One key challenge is that these models reflect the data they were trained on. This can lead to issues such as the temporal generalization problem, where the model struggles to handle facts that change over time, and the factual grounding problem, where it fails to capture specific facts accurately. Additionally, LLMs are prone to generating inaccurate information, commonly referred to as hallucinations or fabulations (ZIEGLER; BERRYMAN, 2023). To mitigate these issues, users can fine-tune the model with additional training examples or use few-shot learning techniques to guide the model toward generating more accurate and contextually appropriate outputs.

3.5.1 Prompt

A prompt is a natural language query provided by the user to guide a LLM in generating a specific response or performing a task. The LLM responds by predicting the most appropriate sequence of tokens—groups of letters or words—that align with the instructions in the prompt, utilizing the knowledge it has learned from its training data.

A prompt can include several key components, such as an instruction, which defines the specific task or command that the model is expected to perform; context, which provides additional information or background to help the model generate a more accurate response; input data, which represents the question or data for which the user is seeking an answer; and an output indicator, which specifies the desired format or type of the model's response (SARAVIA, 2022).

3.5.2 Prompt Engineering

Prompt engineering involves designing text inputs that an LLM model can accurately interpret and respond to. This process provides an effective way for users to interact with LLMs (BROWN et al., 2020). In addition to enhancing user interaction, prompt engineering is employed to improve the security and functionality of LLMs. It helps to mitigate hallucinations, integrate domain-specific knowledge, and incorporate external tools into the models.

Various techniques are employed in prompt engineering, including Zero-shot learning, Few-shot learning, and Chain-of-Thought prompting. Zero-shot learning refers to a model's capability to perform a task without prior specific training on that task (RADFORD et al., 2021). This approach is advantageous because it enables models to handle new tasks or scenarios without requiring extensive retraining, enhancing their adaptability and efficiency across a wide array of tasks. In the context of LLM prompts, Zero-shot prompting involves using a prompt that lacks examples or demonstrations. Essentially, the prompt directly instructs the model to carry out a task without providing additional guidance.

Although large language models exhibit impressive zero-shot capabilities, they often struggle with more complex tasks when relying solely on zero-shot prompts. To address this, the few-shot prompting technique can enhance contextual learning. By including examples within the prompt, the model is guided towards improved performance, with these examples serving as a conditioning mechanism for generating desired outputs (BROWN et al., 2020).

However, the few-shot prompting face difficulties with complex tasks such as arithmetic or commonsense reasoning, even when provided with a large amount examples. In this way, the Chain of Thought (CoT) technique involves adding multiple example solutions to the LLM's prompt, allowing the model to break down complex problems into intermediate steps. By providing a logical sequence of reasoning, CoT helps guide the model in producing more accurate and coherent responses (WEI et al., 2022).

3.6 OpenAI Models

OpenAI is a company dedicated to AI research and deployment, focusing on developing generative models through deep learning. This technology leverages large datasets to train AI systems for specific tasks. OpenAI's text models are language processing tools capable of text generation, classification, and summarization. Their research on generative models also encompasses image and audio processing (OPENAI, 2024a).

The company offers several generative models through the OpenAI API. Among those models, we have the Generative Pre-Trained Transformer (GPT) family, which has different capabilities and prices. Within the GPT family, there are two most relevant models — GPT-3.5 Turbo and GPT-4 Turbo. The GPT-3.5 Turbo model can understand and generate natural language or code for simple tasks. The GPT-4 Turbo is a large multimodal model that processes text or image inputs and generates text outputs. It can tackle complex problems with higher accuracy than its predecessors, owing to its expanded general knowledge and enhanced reasoning abilities (OPENAI, 2024b).

In May 2024, OpenAI launched the GPT-4o, their most advanced multimodal model. The model has the same GPT-4 reasoning abilities but is faster and has the best vision and multi-language performance across the other models from the company. In July 2024, OpenAI launched the faster and cheaper version of GPT-4o, the GPT-4o mini, being cheaper and more efficient than GPT-3.5 Turbo (OPENAI, 2024b).

Table 3.1: Characteristics of some GPT models available on OpenAI API, priced in dollars per million tokens. Data obtained in August 2024.

Model	Context Window	Knowledge Cut Off	Input Prices	Output Prices
GPT-3.5 Turbo	16,385 tokens	Sep 2021	00.50	01.50
GPT-4 Turbo	128,000 tokens	Dec 2023	10.00	30.00
GPT-4o	128,000 tokens	Oct 2023	05.00	15.00
GPT-4o mini	128,000 tokens	Oct 2023	00.15	00.60

In addition to the difference in price and performance, the models present differences in the number of tokens processed at input and the knowledge cut-off for training data. Table 3.1 summarizes some characteristics of the models available, such as the number of tokens in context windows, the training data range, and prices (U$) per million of tokens as input and output.

3.7 Anthropic Models

Anthropic is an AI safety and research company focused on developing artificial intelligence systems aligned with human values that can be controlled safely. The company emphasizes creating AI that benefits society while minimizing risks associated with advanced AI technologies (ANTHROPIC, 2024a).

The company offers the Claude family models, named after Claude Shannon, a foundational figure in information theory. The Claude models are designed to be general-purpose assistants with strong natural language understanding and generation capabilities. In the most recent version (version 3), they presented three models with different capabilities and prices.

The first is Claude 3 Haiku, the fastest and cheapest model for simple tasks that need in-instant responsiveness. The second one is Claude 3 Sonnet, a model with a balance of intelligence and speed, ideal for moderately complex tasks. The last one is Claude 3 Opus, the most expensive and powerful model ideal for highly complex tasks (ANTHROPIC, 2024b).

In June 2024, the Anthropic released Claude 3.5 Sonnet, a new optimized version of the Claude 3 Opus, bringing better reasoning capabilities, latency, vision performance, and natural language understanding (ANTHROPIC, 2024c).

Table 3.2: Characteristics of the Claude 3 models series available on Anthropic API, priced in dollars per million tokens. Data obtained in August 2024.

Model	Context Window	Knowledge Cut Off	Input Prices	Output Prices
Claude 3 Haiku	200,000 tokens	Aug 2023	00.25	01.25
Claude 3 Sonnet	200,000 tokens	Aug 2023	03.00	15.00
Claude 3 Opus	200,000 tokens	Aug 2023	15.00	75.00
Claude 3.5 Sonnet	200,000 tokens	Apr 2024	03.00	15.00

As mentioned previously, the models have different capabilities and prices. Table 3.2 summarizes these characteristics, showing the size of the context window, knowledge cut-off of training data, and the input and output price per million of tokens.

3.8 DeepMind Models

DeepMind is a leading AI research company known for pioneering artificial intelligence and machine learning work. Acquired by Google in 2015, DeepMind focuses on developing advanced AI systems that can learn and solve complex problems, often with applications in healthcare, energy efficiency, and gaming (DEEPMIND, 2024).

Nowadays, DeepMind is a fusion of old DeepMind and Google Brain's, a research team within Google focused on artificial intelligence and deep learning. Established in 2011, it combines machine learning research with large-scale computing resources to advance the field of AI. Google Brain has contributed to key innovations in natural language processing, computer vision, and reinforcement learning, powering many of Google's AI-driven products and services (DEEPMIND, 2024).

The company has different deep learning models for distinct areas, such as computer vision, game agents, speech models, and generative models. Within the generative models, specifically language models, in December 2023, the company released the Gemini multimodal models family — the most capable and general model they ever built. This model family comes in three different sizes — Nano, Pro, and Ultra — designed for tasks with different degrees of complexity (TEAM et al., 2024a).

In February 2024, DeepMind launched the new version of the Gemini family, Gemini 1.5. The new version was built on Gemini 1.0 with significantly enhanced performance. It features a new Mixture-of-Experts (MoE) (SHAZEER et al., 2017) architecture, making it more efficient in training and operation. The model supports a vast context window of up to 1 million tokens, allowing it to process extensive data, such as long documents or videos, with improved accuracy. Gemini 1.5 Pro, the first version released, excels in complex reasoning, multimodal understanding, and in-context learning, setting a new benchmark in AI capabilities (TEAM et al., 2024b).

Table 3.3: Characteristics of the Gemini models family available on Vertex API, priced in dollars per million tokens, obtained in August 2024.

Model	Context Window	Knowledge Cut Off	Input Prices	Output Prices
Gemini 1.5 Flash	1,000,000 tokens	Nov 2023	0.075	00.30
Gemini 1.5 Pro	1,000,000 tokens	Nov 2023	3.509	10.50

Despite the various size options offered in the initial version, only two versions of Gemini 1.5 — Gemini 1.5 Pro and Gemini 1.5 Flash — are accessible through an API. These models are utilized via the Vertex AI API, which is a platform that facilitates the availability and management of AI models in Google's cloud. Table 3.3 provides a summary of the main characteristics of these two models, including the context window size and the input and output prices USD per million tokens.

3.9 Meta AI Models

Meta AI is the artificial intelligence research and development arm of Meta. It focuses on advancing AI technologies across various domains, including natural language processing, computer vision, and robotics. Meta AI aims to build AI systems that can understand and interact with the world like humans (AI@META, 2024a).

The Large Language Model Meta AI (LLaMA) is a series of open-source foundation language models developed by the company. The first version of the series was released with a range from 7 billion to 65 billion parameters, offering strong performance across various natural language processing tasks while requiring significantly fewer computational resources compared to larger models (TOUVRON et al., 2023).

This model collection has undergone several improvements and evolutions since its launch. In April 2024, the company released version 3 of the model collection, bringing optimizations to the architecture and better task execution performance, with 8 and 70 billion parameters (AI@META, 2024b). In July 2024, version 3.1 of the collection was released, with a larger context window and further improvements to the reasoning capacity. Among the models released in this version, we had the open source model with the largest number of parameters, LLaMA 3.1 405 billion parameters (DUBEY et al., 2024).

Model	Context Window	Knowledge Cut Off
LLaMA 3 8B	8,000 tokens	Mar 2023
LLaMA 3 70B	8,000 tokens	Dec 2023
LLaMA 3.1 8B	128,000 tokens	Dec 2023
LLaMA 3.1 70B	128,000 tokens	Dec 2023
LLaMA 3.1 405B	128,000 tokens	Dec 2023

Table 3.4: Characteristics of the LLaMA models collections.

Table 3.4 summarizes the relevant information about the recent collection versions, such as the model knowledge cut-off and the context window size. As these are open-source models, we do not have a specific service for consumption, therefore we do not have pricing information.

3.10 Mistral AI Models

Mistral AI is a cutting-edge artificial intelligence company focused on developing advanced language models. It aims to push the boundaries of

AI by creating models that are highly efficient, powerful, and capable of understanding and generating human-like text across various applications (AI, 2024b).

The company has several language models, ranging from open source to commercial, with different sizes, capabilities, and objectives. Among the main open source models, include Mistral 7B, a highly efficient 7-billion parameter model designed for strong performance in text generation and understanding (JIANG et al., 2023), and Mistral-8x22B, a mixture-of-experts model with 8 experts, each containing 22 billion parameters. The latter dynamically activates relevant experts for each task, offering powerful capabilities with optimized computational efficiency (AI, 2024a).

Model	Context Window	Knowledge Cut Off
Mistral 7B	8,000 tokens	Aug 2021
Mistral-8x22B	64,000 tokens	-

Table 3.5: Characteristics of the Mistral main open source models.

The Table 3.5, condenses the relevant information about the two models. As these are open-source models, the information available is only related to the context window size and the knowledge cut-off. Despite providing open weights models, free for local deployment, Mistral AI offers a cloud API to consume the models, billing per million of input and output tokens.

3.11 LangChain

LangChain is a framework for developing applications powered by language models. This framework supports the development of applications with a context-awareness approach by linking a language model to different sources of context, such as prompt instructions, few-shot examples, or additional content, the application can ground its responses appropriately (LangChain, 2024).

The main feature of LangChain is its capability to easily trigger an LLM with a given input. For example, Code 1 shows how to import the model API wrapper, configure the parameters, and invoke the model to get a response.

Code 1: Example of Using LangChain to Invoke the Gemini Model API

```
from langchain_google_genai import ChatGoogleGenerativeAI

llm_model_name = "gemini-1.5-flash-001"
llm = ChatGoogleGenerativeAI(
    model=llm_model_name,
    temperature=0,
    max_tokens=None,
    timeout=None,
    max_retries=2
)

message = [("human", "Tell me a history about the origin of documents.")]
response = llm.invoke(message)

print(response.content)
```

Although the provided example is basic, real-world applications often involve interactions with the LLM, where each step's intermediate results require logical processing before proceeding to the next step. This sequence of linked interactions is referred to as chains. Chains typically involve integrations with multiple components, such as model providers, data storage systems, and APIs. These modules can be combined to create more complex applications or used individually for simpler tasks.

LangChain is useful for developing applications like personal assistants, chatbots, and document-based question-answering systems. It can also summarize extensive documents, extract structured information from unstructured text, and query tabular data.

In this chapter, we introduce the key concepts necessary for a thorough understanding of the work. We begin by explaining the task to which our annotation process will be applied for the generation of the annotated dataset. Next, we cover concepts related to document image processing, such as OCR, DLA, and TSR, which enable the creation of machine-readable representations of each image. Finally, we present the tools and concepts relevant to the understanding and use of LLMs, highlighting the models from the major manufacturers utilized in this work. In the next chapter, we will present the proposed method, which builds upon all the concepts discussed so far.

4
Proposed Process

This chapter describes this work's core, a *Data Annotation Approach Using Large Language Models*. The process is composed of three stages: (i) Transcription, (ii) Question-Answer Generation, and (iii) Question-Answer Judgment. The transcription stage generates a textual representation of document files and feeds an LLM to generate QA tuples in the second stage. Finally, the same transcription and the generated QA tuples feed another LLM to select the valid tuples to compose the dataset. Figure 4.1 depicts this process, and each of its steps is described in the next paragraphs.

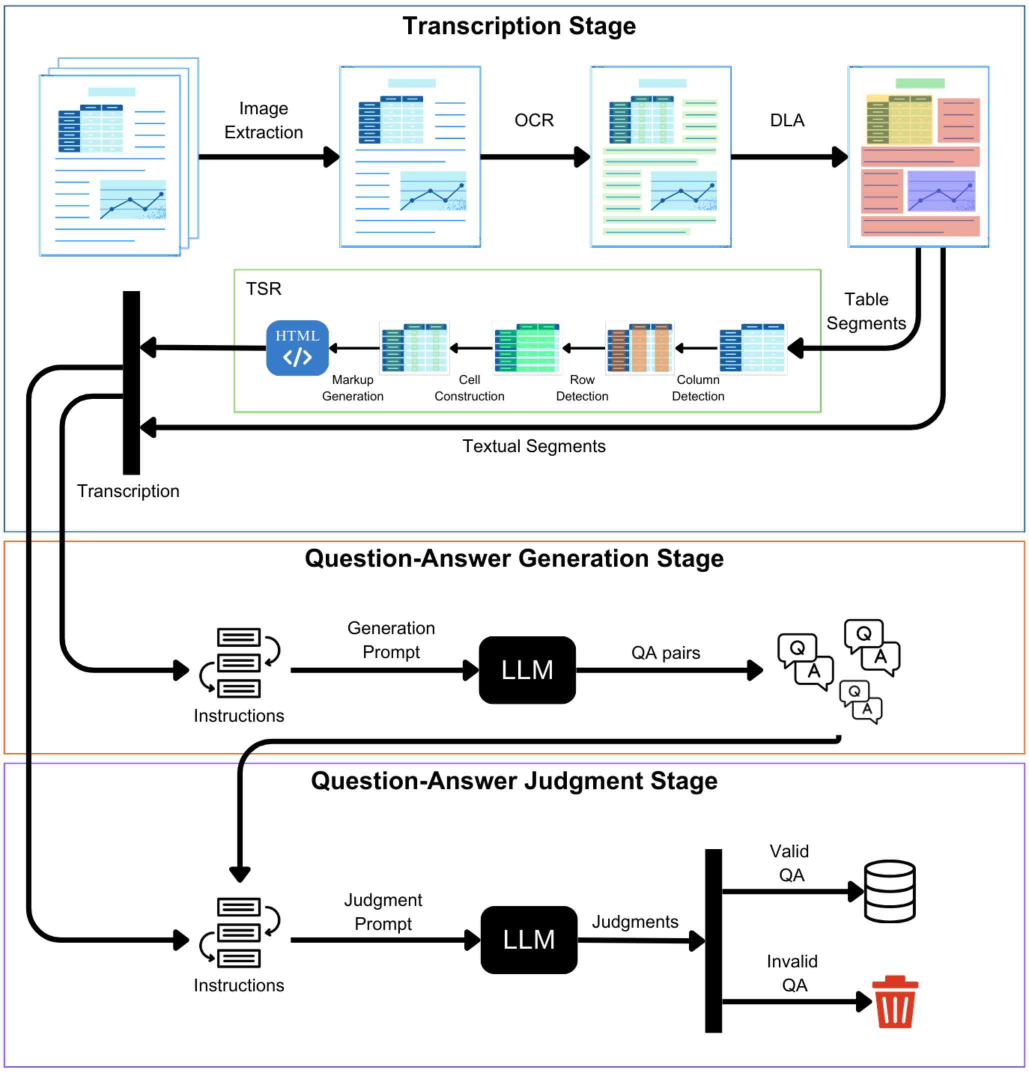


Figure 4.1: Proposed process Overview.

The proposed process starts with the Transcription Stage, where we perform an image extraction by receiving a document file as input and extracting the desired pages. These pages are used as a set of images for the next steps. After image extraction, we apply an OCR tool to detect the textual content of the document image and transcribe it.

The DLA step uses an object detection model to detect the structure of a document page. In general, object detection models can identify which objects from a known set are present in the image and provide information about their position. In our case, the objects are document segments, such as tables, images, section headers, etc. Therefore, the DLA model is responsible for returning the bounding boxes of the segments present in the image. We can isolate and extract the table images with these bounding boxes to feed the next step.

In the TSR step, a second object detection model is applied to identify and segment table structures. The model identifies its rows and columns bounding boxes for each table, constructing the concept of cells from the intersection of bounding boxes of both elements. Next, a second algorithm is used to convert these bounding boxes into the corresponding HTML code.

Concluding the Transcription Stage, we obtain the transcription of the document image provided as input. This transcription feeds the LLM in the Question-Answer Generation stage as a reference for proposing questions and constructing answers.

Finally, the objective of the Question-Answer Judgment stage is to ensure that the generated questions and answers are coherent and correct, filtering out possible hallucinations from the generating model and increasing the quality of the final dataset.

The methodology is organized as follows: first, we report the steps taken for data acquisition. Next, we present the process of selecting the OCR tool, DLA, and TSR models. We then discuss the selection of LLM and the construction of prompts for the Question-Answer Generation Stage. Finally, we explain the selection of LLM and the construction of prompts for the Question-Answer Judgment Stage.

4.1 Data Acquisition

Databases are fundamental for the development and validation of machine learning models. To validate the proposed process in this research, we collected public documents to compose our database.

Given the diversity of available public documents, we chose to focus on documents from the financial domain. In this domain, the CVM requires companies to publish documents related to financial statements, social impact factors, and the opinions of the fiscal council.

These documents are published at two distinct intervals — quarterly and annually — resulting in at least five different documents per year. Given the number of listed companies and the volume of available documents, we limited the scope of this work to the annual financial statements of the 100 publicly traded companies with the highest trading volume on the Brazilian stock exchange.

Thus, we collected 937 financial statements from these 100 companies, covering 2015 to 2023. Due to the large volume of pages, which do not contain any annotations or metadata regarding transcription or layout, only a random subset of document pages was used to conduct the experiments presented in the following sections.

4.2 Transcription

In this stage we will describe all the steps needed to extract the textual representation of the document that will serve as input for the following stages. We will start with OCR step, followed by DLA and finally TSR.

4.2.1 Optical Character Recognition (OCR)

In this step, we processed the documents collected in Section 4.1 using an OCR tool, which transcribed the text from each document page and provided bounding boxes with the coordinates of each detected word.

This step is crucial because the extracted text is used as input for the next steps of this methodology, specifically in Sections 4.2.3, 4.3 and 4.4. Therefore, detection failures or transcription errors could compromise the entire annotation process.

Due to its importance and the lack of annotated data to train and evaluate open-source OCR tools, we selected the commercial OCR tool Microsoft OCR V4.[1] This choice was based on the work of Santos, Silva e Reis (2023), who compared various OCR tools using a diverse set of images containing text in Brazilian Portuguese. Their tests showed that the Microsoft model outperformed all other evaluated models, delivering solid results even with different backgrounds and text rotation levels.

[1]https://azure.microsoft.com/en-us/products/ai-services/ai-document-intelligence

4.2.2 Document Layout Analysis (DLA)

In this step, we applied a DLA model on the extracted images of each document page to segment paragraphs, images, and tables. The output consisted of bounding boxes for the identified regions and their respective classes.

After segmentation, we stored the identified regions in a data structure that retained both the bounding boxes and the detected class. This structure also stored each region's positional information and the textual content detected by the OCR tool within the segmented areas.

To determine whether a text lies within a specific region, we used the intersection between the bounding boxes of the text and the segmented region. Let B_T represent the bounding box of the text and B_R the bounding box of the segmented region. The intersection between B_T and B_R is denoted as $I(B_T, B_R)$. To assess whether the text is contained within the segmented region, we calculated the intersection over union (IoU) coefficient, defined as:

$$\mathrm{IoU}(B_T, B_R) = \frac{I(B_T, B_R)}{A(B_T) + A(B_R) - I(B_T, B_R)}$$

in which $A(B_T)$ and $A(B_R)$ represent the areas of the bounding boxes B_T and B_R, respectively. If $\mathrm{IoU}(B_T, B_R) \geq 0.9$, we considered the text part of the corresponding segmented region. In such cases, the text was linked to that segment and not assigned elsewhere.

This segmentation and data structure construction process could be performed by any DLA model. Given the variety of available DLA models, we selected models from the literature and conducted experiments to choose the best one, which we describe in more detail in Section 5.1.

4.2.3 Table Structure Recognition (TSR)

The TSR step is responsible for generating a viable textual representation to be used as input prompts for the LLM, as outlined in Sections 4.3 and 4.4. This recognition is applied to regions identified and classified as tables by the layout analysis stage (Section 4.2.2).

The recognition process uses a detection model to identify the table's cell structure, including horizontally and vertically merged cells, and returns their respective bounding boxes. The detection results are then transformed into HTML code, which is populated with textual tokens using the IoU method, similar to the approach described in the DLA stage (Section 4.2.2).

This table structure recognition process can be performed by any TSR model. Given the variety of TSR models available, we selected models from the literature and conducted experiments to determine the best option. These experiments are further detailed in Section 5.2.

4.3 Question-Answer Generation

In this stage, the selected LLM receives as input, the generation instructions, document transcription with positional markers and tasked with generating three question-answer tuples using only the information available in the transcription.

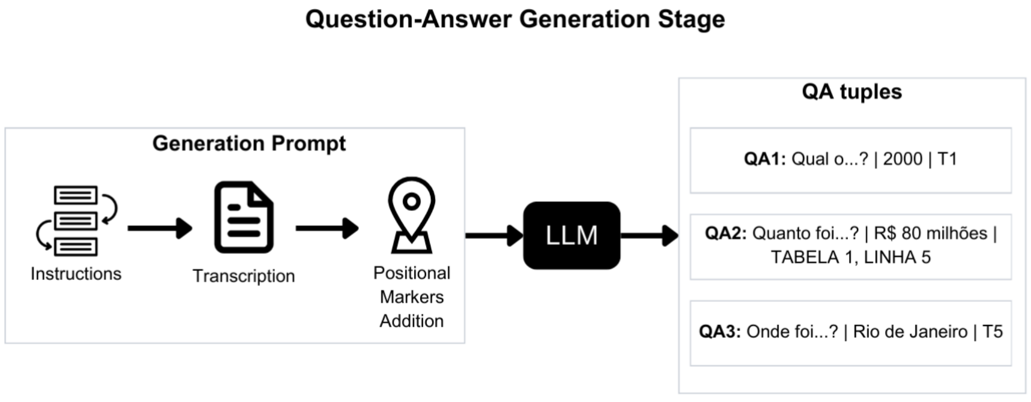

Question-Answer Generation Stage
Generation Prompt
Instructions
Transcription
Positional Markers Addition
LLM
QA tuples
QA1: Qual o...? | 2000 | T1
QA2: Quanto foi...? | R$ 80 milhões | TABELA 1, LINHA 5
QA3: Onde foi...? | Rio de Janeiro | T5

Figure 4.2: Overview of the steps taken to generate the questions and answers.

Figure 4.2 outlines the steps required to generate each QA tuple. The first step uses a system prompt, providing detailed instructions on how to execute the task. These instructions include the coherence standards the model should follow, rules for using relative pronouns, the number of desired tuples, the focus area, and the key elements to include in the output. The prompt also provides positive and negative examples to guide the generation process.

The second prompt contains the transcribed document, resulting from the Transcription stage, described in Sections 4.2.1, 4.2.2 and 4.2.3. Positional markers are inserted into the transcription to standardize the output and reduce the randomness inherent to LLM models.

In this way, the expected output of this stage is a textual format divided into three parts:

QUESTION | ANSWER | TEXT REGION

The first and second parts correspond to the QA tuple generated by the model. The third part is the positional marker, following the established format based on the text region used for question and answer generation. The prompt used for QA tuples generation is detailed in Appendix section 8.1.

4.4 Question-Answer Judgment

In this stage, the chosen LLM was fed the document transcription, and the three QA tuples were generated during the Question-Answer Generation Stage to evaluate the coherence of the questions and the correctness of the answers.

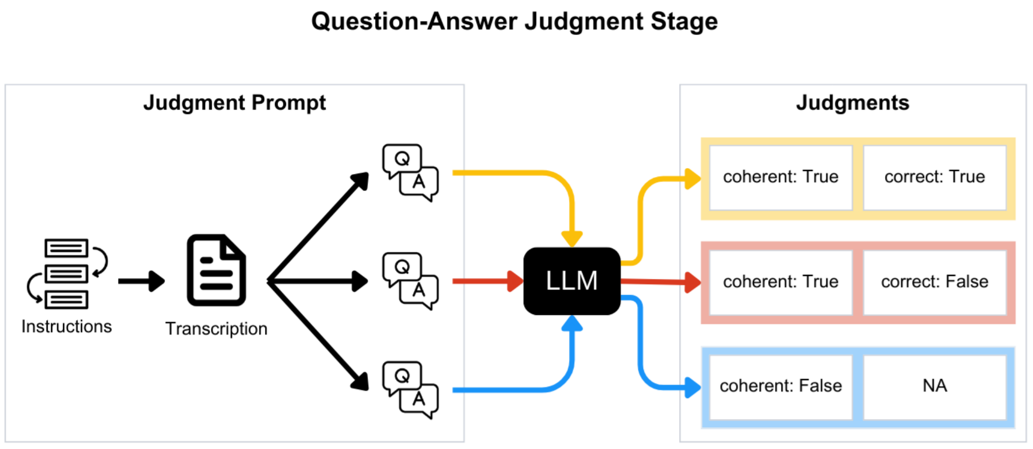


Figure 4.3: Overview of the steps taken to judge the questions and answers.

Figure 4.3 outlines the steps required to judge each QA tuple. The first part contains the necessary instructions for the task. It begins with an overview of the task, setting the standards the model should follow. Next, the specific criteria for evaluating the questions and answers are presented. Unlike the generation stage, the document transcription is provided without positional markers.

The following parts are used for the direct evaluation of the QA tuples. The process is divided into two calls to the LLM. The first call evaluates the coherence of the question, considered coherent if the questions comply with the correct use of the grammatical rules of the Portuguese language and have only one answer. The second call evaluates the answer to see if it answers the question.

Since each QA tuple is evaluated individually, the process described is carried out in 3 iterations per document page. The expected output for each iteration is binary, confirming or denying the coherence of the question and the validity of the provided answer. The prompt used for QA tuples judgment is detailed in Appendix Section 8.2.

5
Experiments

In this chapter, we describe the experiments conducted, following the flow of the proposed process. We begin with the experimentation of the transcription stage, where we evaluated available Document Layout Analysis (DLA) models from the literature using a dataset created during this study. Next, we assessed open-source Table Structure Recognition (TSR) models, testing them on four different datasets to measure their generalization capabilities.

After the transcription stage, we proceeded to evaluate various LLMs for the question and answer generation stage. Finally, we assessed more robust LLMs for the QA tuple judgment phase.

5.1
Document Layout Analysis (DLA)

This section describes the experiments conducted to determine the DLA model used in this work. Additionally, we outline the process of creating the dataset and the metric employed for model evaluation. Finally, we present the results for each evaluated model and define the best model we found for the proposed process.

5.1.1
Dataset

To evaluate the DLA models, we constructed an annotated dataset based on the DocLayNet dataset (PFITZMANN et al., 2022a), which we selected due to its similarity to the domain of the documents used in our work. Additionally, DocLayNet offers a wide range of document layouts with varying levels of complexity.

For the validation dataset, we randomly selected 990 pages from the financial statements of different companies. We prioritized the central pages of these statements, which typically contain text, tables, and images.

Each selected page was annotated individually by two annotators, following the same criteria used in the construction of DocLayNet (PFITZMANN et al., 2022b). Both annotators reviewed each other's annotations to improve consistency.

Table 5.1 presents the number of elements for each annotated class. The most common class is *Text*, as these documents are highly narrative, containing many paragraphs of explanations. The second most frequent class

Table 5.1: Annotated instances per class in constructed evaluation dataset for DLA step.

Class	Instances
Text	4993
Page-header	3478
Section-header	2008
Page-footer	1539
List-item	1121
Table	986
Picture	439
Caption	239
Footnote	29
Title	5
Formula	0

is *Page-header*, since every page includes information such as the company's name, publication date, and report version. The third, fourth, and fifth most frequent classes are *Section-header*, *Page-footer*, and *List-item*, which reflect the structured way companies present information by segmenting content into sections and using lists. The *Page-footer* class typically includes logos or page numbers.

The less frequent classes are *Tables*, *Pictures*, and *Captions*. Despite the low occurrence of *Tables*, this class is crucial for our work. The *Picture* class is more complex, as it involves detecting all visual elements such as charts, infographics, and illustrations. *Captions* are closely related to *Pictures*, as they provide titles or descriptions for images in the documents.

Finally, the *Footnote* and *Title* classes are nearly absent, justified by the common structure of financial documents, using few footnotes and presenting titles only on the first pages. The *Formula* class had no occurrences, as this category is more common in technical or scientific materials.

5.1.2 Metrics

To evaluate the models, we used the Mean Average Precision (mAP), a standard metric for assessing object detection and segmentation models. For each class, the Average Precision (AP) was calculated, representing the area under the precision-recall curve. The mAP is the average of the APs across all classes, offering a comprehensive view of the model's performance, crucial in benchmarks like COCO and PASCAL VOC (EVERINGHAM et al., 2010).

5.1.3 Models Selection

The model selection was based on two main criteria. First, we searched for DLA models trained on the DocLayNet dataset. Second, we required the model to be open-source, allowing access to its weights for evaluation and modification.

Four models were selected for evaluation, each with unique features for document layout segmentation. The first was SwinDocSegmenter, a model based on the Swin Transformer, using deformable attention and contrastive denoising strategies for precise segmentation (BANERJEE et al., 2023). The second model, RoDLA, combines the DINO architecture (ZHANG et al., 2022) with attention channels, designed for robust layout analysis across various document types (CHEN et al., 2024). The remaining models, Malaysia-AI (Malaysia-AI, 2024) and Maik Thiele (Maik Thiele, 2024), are adaptations of YoloV8 for layout segmentation, focusing on agility (JOCHER; CHAURASIA; QIU, 2023).

Table 5.2: mAP performance for each class and model on constructed dataset. The best performance is formatted in boldface.

Class	RoDLA	Malaysia-AI	SwinDocSegmenter	Maik Thiele
Caption	**0.302**	0.015	0.067	0.127
List Item	**0.860**	0.741	0.676	0.642
Picture	**0.749**	0.413	0.569	0.524
Text	**0.979**	0.910	0.776	0.901
Footnote	**0.735**	0.486	0.345	0.254
Page Footer	**0.911**	0.446	0.621	0.435
Section-header	**0.776**	0.490	0.391	0.346
Title	0.007	**0.015**	0.003	0.003
Formula	-	-	-	-
Page Header	**0.870**	0.476	0.485	0.447
Table	**0.975**	0.955	0.899	0.906
All	**0.716**	0.495	0.483	0.459

All models were evaluated using the constructed dataset, measuring both mAP and execution time. Table 5.2 shows the mAP performance for each class. RoDLA achieved the highest overall performance, likely due to its design for handling diverse document types and perturbations. While optimized for speed, Malaysia-AI performed well in most classes, particularly in detecting tables and text. Despite using the same architecture as Maik Thiele, Malaysia-AI's superior performance resulted from different data augmentation techniques adopted during the model training. SwinDocSegmenter, although computationally demanding, showed lower performance than Malaysia-AI, which can run efficiently on CPU.

Table 5.3: Size and average time cost for each model to segment 990 images on the constructed dataset.

Models	Parameters	Avg Time Cost
RoDLA	338.9M	1.27 ± 0.01
SwinDocSegmenter	213.3M	0.87 ± 0.01
Malaysia-AI	68.1M	0.35 ± 0.03
Maik Thiele	68.1M	0.33 ± 0.03

Table 5.3 details the execution time required to process 990 images on a Tesla T4 GPU with 15 GB VRAM. RoDLA, despite its superior segmentation performance, had the slowest throughput, likely due to its larger model size and feature extraction process. Malaysia-AI and Maik Thiele demonstrated the fastest throughput, benefiting from the low latency of the YoloV8 architecture (JOCHER; CHAURASIA; QIU, 2023).

In conclusion, RoDLA is the best choice for DLA task, given the importance of accurate layout segmentation in this research. Despite its higher execution time, RoDLA's superior ability to identify various document regions ensures more precise segmentation, making it the ideal model for this task.

5.2 Table Structure Recognition (TSR)

In this section, we present the experiments conducted to select the TSR model used in the proposed process. Additionally, we outline the datasets and metrics used for evaluation. Finally, we present the results and the selected model.

5.2.1 Datasets

Building a specific dataset for model evaluation would be unfeasible due to the high time cost associated with data annotation and refinement. This cost arises from the need to complete three distinct tasks: 1) marking the bounding boxes of columns and rows; 2) transcribing the table text; and 3) creating the HTML structure.

Thus, four distinct datasets were selected for TSR task evaluation. The first, PubTabNet, contains about 500,000 table images along with their HTML equivalents for the structural recognition of tables (ZHONG; SHAFIEIBA-VANI; YEPES, 2020). FinTabNet, which specializes in financial documents, offers annotated data for table extraction and conversion to HTML, focusing on the unique challenges of this document type (ZHENG et al., 2021). Finally, we modified the ICDAR 2013 Table Competition dataset, used as the benchmark, was modified for this work by converting its ground truths from XML to HTML to match the format of the other selected datasets (GÖBEL et al., 2013).

5.2.2 Metric

We used a variant of the Tree-Edit-Distance-based Similarity (TEDS) proposed by Zhong, ShafieiBavani e Yepes (2020) to measure the table extraction model performance. The original TEDS calculates the distance between the ground truth table tree and the predicted one. In this process, the metrics consider the alignment between the markup and textual content of the table.

The similarity between the two tables is evaluated using tree-edit distance, as proposed by Pawlik and Augsten (PAWLIK; AUGSTEN, 2016). Insertion and deletion operations have a cost of 1. The substitution cost is 1 if at least one node is not a table data node. For table data nodes, the cost is 1 if their column or row spans differ; otherwise, it is the normalized Levenshtein similarity (LEVENSHTEIN et al., 1966) between their contents. The TEDS computed between two trees is

$$TEDS(T_a, T_b) = 1 - \frac{EditDist(T_a, T_b)}{max(|T_a|, |T_b|)}$$

where EditDist denotes tree-edit distance, and |T| is the number of nodes in T. The table recognition performance is assessed by calculating the mean tree-edit distance similarity score between the recognition result and ground truth for each sample (ZHONG; SHAFIEIBAVANI; YEPES, 2020).

To ignore errors that come from text transcription, Qiao et al. (2021) proposed the TEDS-Struct, a modification in the use of the TEDS metric, which evaluates only the final HTML structure of the table, ignoring the accuracy of the textual content. Therefore, for this work we adopt TEDS-Struct to evaluate TSR models performance.

5.2.3 Model Selection

To select the TSR models evaluated in this work, we searched for open-source models that could perform table-to-HTML conversion from table images. Four models were selected for evaluation: 1) Table Transformer (TATR); 2) MLT-TabNet; 3) Table Master and 4) Local and Global Pyramid Mask Alignment (LGPMA).

The TATR, a transformer-based model developed for table detection and structural recognition. It comes in two versions: All and Fin. The All version is trained on over 1 million tables, featuring diverse styles and formats, while the Fin version is specifically trained on financial documents using a modified version of the FinTabNet dataset (SMOCK; PESALA; ABRAHAM, 2022).

MLT-TabNet combines computer vision techniques with a Transformer-based encoder-decoder architecture for cell detection, text transcription, and table structure extraction (LY; TAKASU, 2023). TableMaster adopts an end-to-end approach for table extraction and HTML conversion, integrating image processing with natural language processing (YE et al., 2021). Lastly, LGPMA stands out for its ability to segment complex tables in diverse documents using a spatial attention-based approach (QIAO et al., 2021).

The models were evaluated using the four datasets mentioned earlier, allowing us to assess their generalization capabilities. This approach provided a comprehensive analysis of each model's ability to handle various document types and table structures.

Table 5.4: TEDS-Struct performance by model on each dataset. The best performance is formatted in boldface.

Model	FinTabNet	PubTabNet	ICDAR 2013
TATR All	0.92	0.93	**0.94**
TATR Fin	0.92	0.88	0.91
MLT-TabNet	**0.97**	0.90	0.91
TableMaster	0.80	**0.97**	0.90
LGPMA	0.41	0.96	0.91

Table 5.4 shows the results for each tested model. While no model outperforms all others across all datasets, TATR All and MLT-TabNet demonstrated the best generalization abilities. Among these two top-performing models, TATR All showed the best performance on the ICDAR 2013, our benchmark dataset.

The results indicate that models trained on the FinTabNet dataset have a greater capacity for generalization. This is likely due to the dataset's complexity and variety of examples, featuring different degrees of customization. This contrasts with PubTabNet, which only includes tables from scientific publications.

The second factor evaluated was the execution time. Table 5.5 shows the time required to process 258 tables from the benchmark dataset on a Titan V GPU with 12 GB VRAM. TATR outperformed the other models in speed, extracting the structure faster because it does not transcribe the text within each table cell. However, it uses the output of the OCR performed previously to fill in the text of each cell quickly.

Table 5.5: Average time cost for each model to extract the structure of 258 tables.

Model	Avg Time Cost
TATR All	0.12 ± 0.03 s
TATR Fin	0.12 ± 0.01 s
LGPMA	0.26 ± 0.02 s
MLT-TabNet	4.70 ± 0.03 s
TableMaster	8.30 ± 0.04 s

Models that handle both structural recognition and text transcription, such as MLT-TabNet and TableMaster, were the slowest. Their slower speed is due to the number of tasks they execute, including detection, transcription, and HTML conversion.

Given its extraction performance and high execution time, TATR All is the best model for this work. It demonstrated strong generalization abilities, performing well across all datasets tested. Moreover, it was the fastest model with low latency. Besides the temporal tests executed on GPU, TATR is the only model capable of maintaining similar execution time on CPU, further demonstrating its superior performance.

5.3 Question-Answer Generation

This section outlines the criteria for selecting the LLMs used in the Question-Answer Generation stage. We also describe the datasets utilized during the experiments and present the results for each model.

5.3.1 Model Selection

We selected models based on cost-benefit for the Question-Answer Generation stage, considering the token usage required to complete the task. This stage generates the highest number of output tokens, making it the primary source of costs. On average, the document transcriptions and task instructions need on average $3,471(\pm 1080)$ input tokens, while the question, answer, and positional marker generate on average $170(\pm 36)$ output tokens.

Figure 5.1 illustrates the quality and cost in USD per million tokens for the main available models. The prices reflect a blended rate, assuming a 3:1 ratio of input to output tokens. Quality is presented as an average result across the MMLU (HENDRYCKS et al., 2020), GPQA (REIN et al., 2023), MATH (HENDRYCKS et al., 2021), HumanEval (CHEN et al., 2021), and MGSM (SHI et al., 2022) datasets.

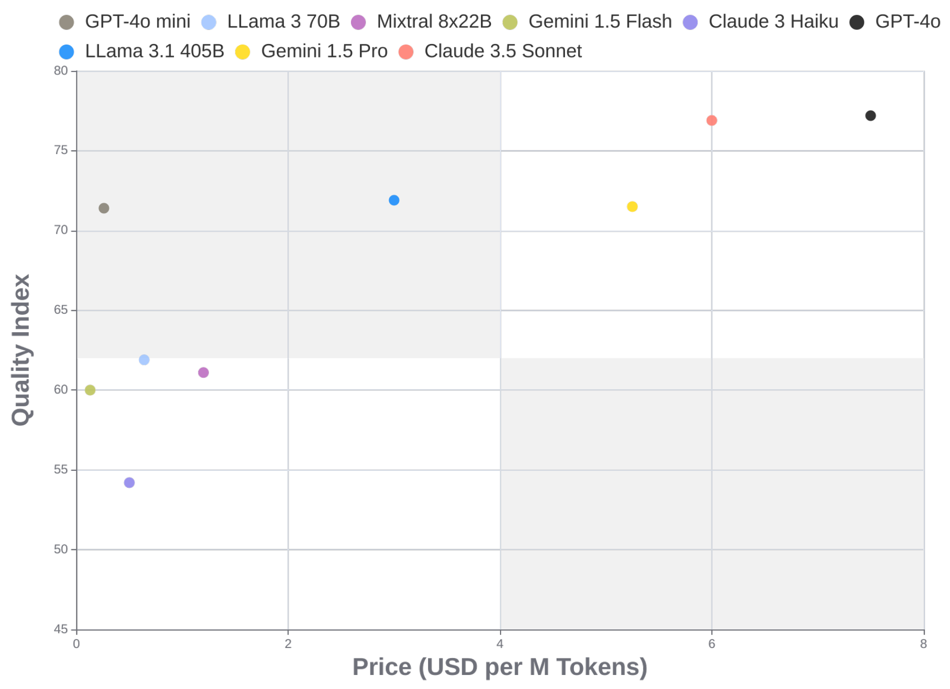


Figure 5.1: Quality vs Price for the main LLM models available via a pay-per-use API. The price of each model was collected in August 2024. Adapted from Analysis (2024).

The models are categorized into four quadrants. The first quadrant includes models with high quality and high cost per token, suitable for complex tasks requiring advanced capabilities. The second quadrant features models with high quality and low cost per token, ideal for simpler tasks and offering a competitive cost-benefit ratio. The third quadrant comprises models with reasonable quality and low cost, generally suitable for basic tasks such as conversation.

Therefore, based on the average amount of tokens necessary for the task and price versus quality exhibited in Figure 5.1, we chose the models from the second and third quadrants: GPT-4o mini, LLaMA 3 70B, Mixtral 8x22B, Gemini 1.5 Flash, and Claude 3 Haiku. However, we excluded LLaMA 3.1 405B for this stage, because is the better and expensive open-source model that will be used in the second stage.

5.3.2 Datasets

To conduct the experiments on question generation using LLM, we created two datasets: validation and test, each with distinct sizes and objectives.

The validation set, built to be used in the development of prompt and parameters tuning, consists of 20 pages randomly selected from the collected documents, described in Section 4.1. The selected pages are not continuous, and the information on a page does not necessarily start or end on the same page.

Table 5.6: Validation and Test dataset page layout samples distribution.

Class	Validation	Test
Text	0	10
Table	7	94
Mixed (Text + Table)	13	196

Table 5.6 shows the distribution of layouts in the validation set and test set. Most of the validation dataset consists of pages with varying sizes and arrangements of text and tables. These pages contain a variety of tables interspersed with paragraphs that either complement or elaborate on the information presented. In some cases, the textual content covers topics unrelated to the table. Pages that contain only tables have different structures, either showing multiple tables summarizing information from various sections or a single complex table occupying the entire page, which presents a higher level of complexity for reading and interpretation.

The test set follows a similar structure to the validation set and was built to evaluate the proposed process's generalization capacity. It comprises 300 pages, including mixed pages with text and tables, pages with only tables, and pages with only text. The distribution is consistent with the validation set, with the primary difference being the presence of pages containing only continuous text.

5.3.3 Experimental Setup

In the experiments conducted to assess the generation quality for each model, we used Langchain to standardize the API calls and output metadata across all models. For consistency, we set the temperature to zero for all models, ensuring more deterministic outcomes. All other parameters were kept at their default settings for each respective model.

5.3.4 Question Variation

The first experiment assessed the distribution of questions based on their initial 3-grams. The goal was to evaluate the models' ability to generate diverse questions.

Figure 5.2 shows the distribution of questions generated for the validation set. Each level represents the most frequent words, following the natural order

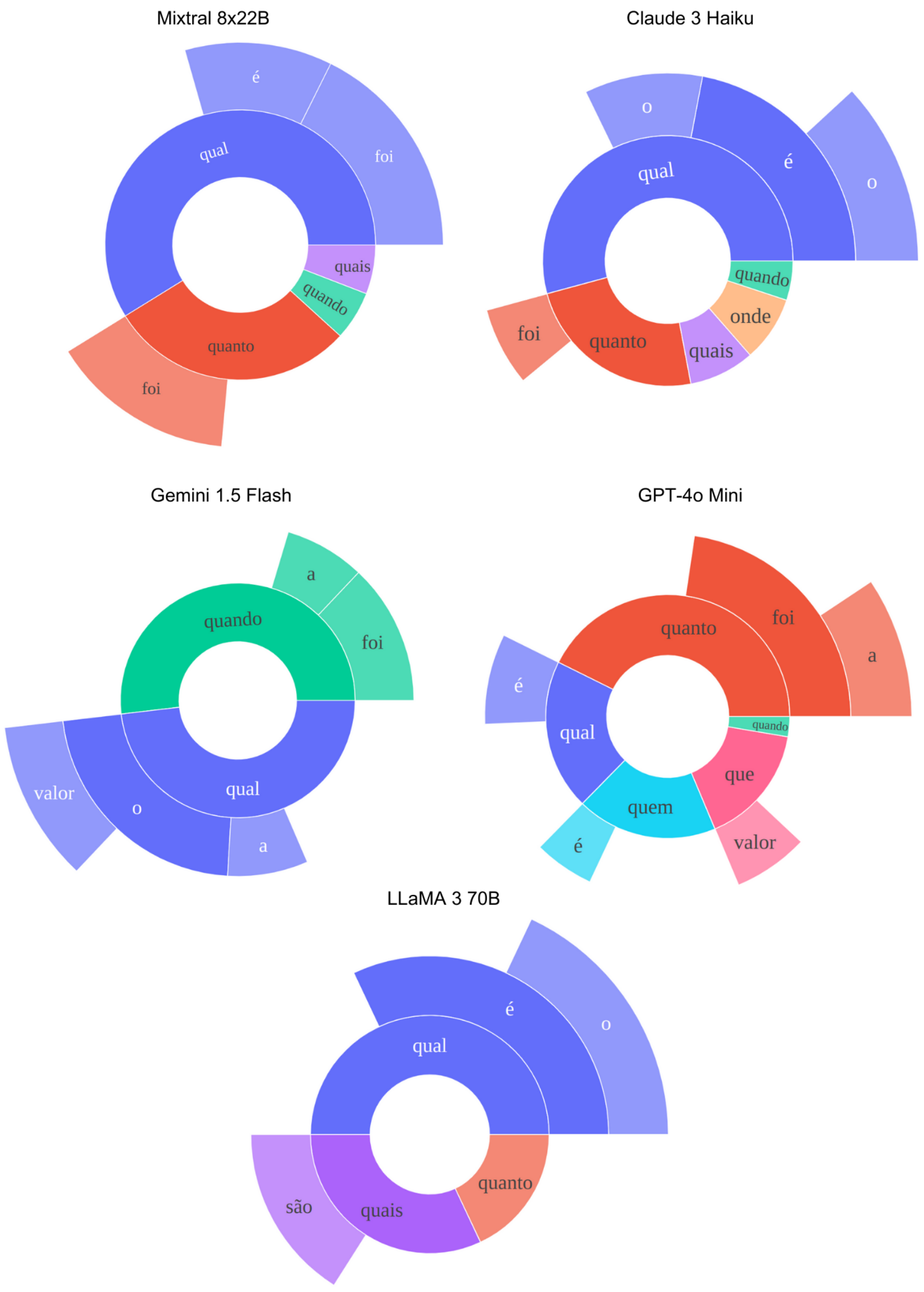


Figure 5.2: Distribution of questions by their starting 3-grams in the validation dataset.

of human reading. Thus, the first level corresponds to the first word of the question, the second level to the second word, and so on. For instance, in the word distribution generated by Mixtral 8x22B, the most frequent questions begin with “Qual foi” (“What was”) and “Qual é o” (“What is the”). In the second level, empty regions indicate words that are not frequent enough to be represented in the chart.

Most models managed to use at least three different interrogative pronouns, whereas the Gemini 1.5 Flash model showed less variety, relying heavily on a more conservative approach to question phrasing.

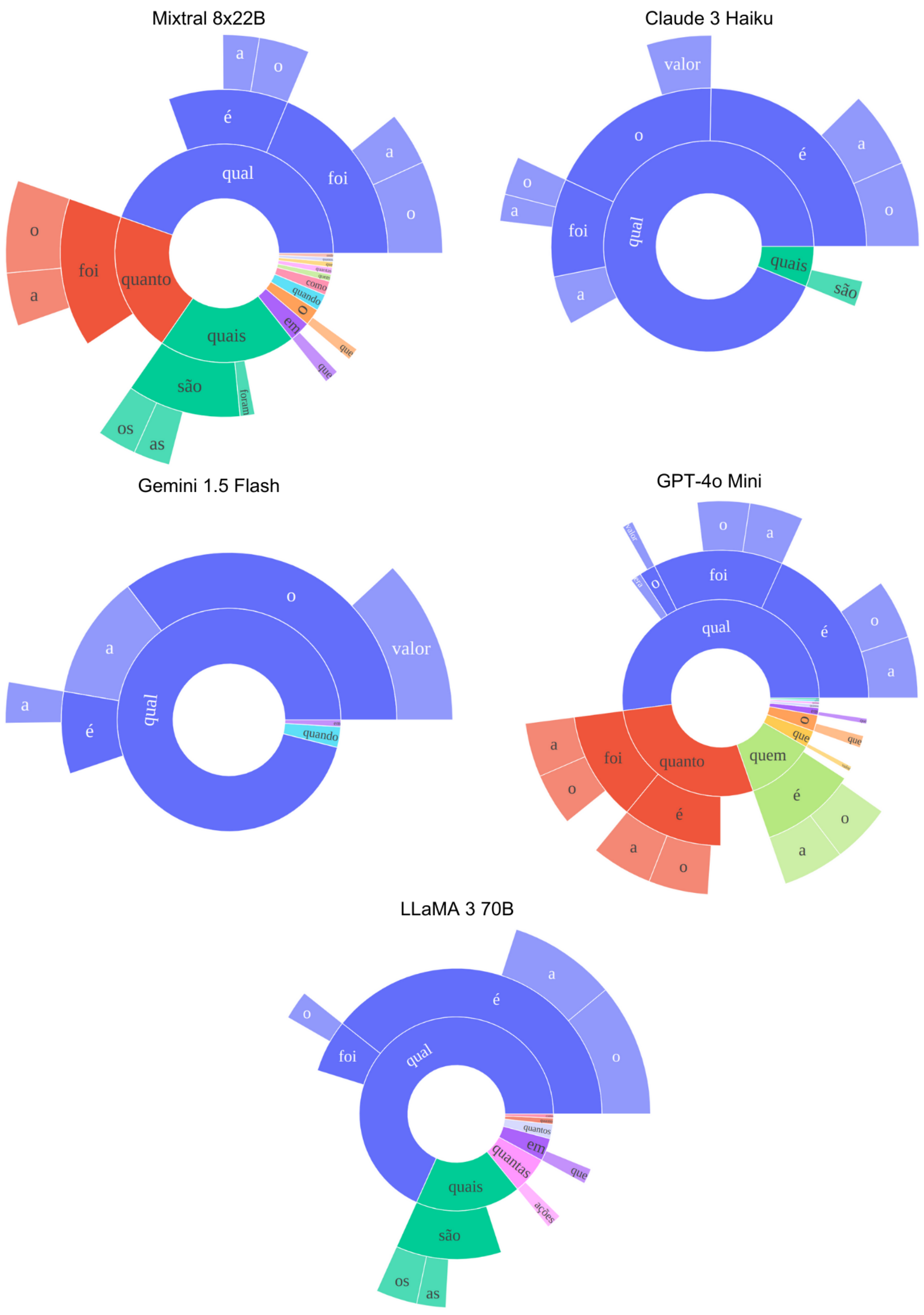


Figure 5.3: Distribution of questions by their starting 3-grams in the test dataset.

In the larger test set, the models exhibited similar behavior. Figure 5.3 shows the question distribution for this dataset. Most models employed at least five variations of interrogative pronouns. The conservative behavior of Gemini 1.5 Flash persisted, and unlike in the validation set, Claude 3 Haiku showed limited variation.

Most models favored using *"Qual"* ("What") as the interrogative pronoun, likely due to its adaptability in various contexts. This pattern aligns with human-generated questions, as demonstrated by Mathew et al. (2021). *"Quanto"* ("how much" or "how many") was the second most frequently used pronoun, linked to the prevalence of tables in the selected pages. Lastly, *"Quando"* ("when") and *"Quem"* ("Who"), though not explored equally by all models, were employed by GPT-4o mini and Mixtral-8x22B.

Based on the variation in both datasets, GPT-4o mini and Mixtral-8x22B showed the most promise for question generation. Considering the cost-performance trade-off, Mixtral-8x22B is the optimal choice, being open-source and offering strong performance at a lower cost.

Table 5.7: Number of questions generated for each grounded regions.

Model	Validation Dataset		Test Dataset	
	Text	Table	Text	Table
LLaMA 3 70B	12	48	182	688
Mixtral 8x22B	04	56	122	760
Gemini 1.5 Flash	12	48	125	757
GPT-4o mini	06	54	109	773
Claude 3 Haiku	10	50	089	793

Beyond the initial 3-grams of questions, we evaluated the distribution of the generated questions comparing table and text regions. Table 5.7 shows the number of questions generated for each grounded region by model in both datasets. We can note that all models focus on table regions, respecting the instruction described in the generation prompt.

5.3.5 Positional Marker Returning

All models were instructed to return to the location of the text where the answer to the posed question could be found. The prompt provided examples of the required formatting, which varied depending on the target region of the question, as detailed in appendix section 8.1. This formatting is crucial for facilitating human validation.

We evaluated the models based on their ability to follow the formatting instructions given in the prompt. This evaluation focused on the text in the returned region, disregarding any numerical characters within that area. Below is a list of valid combinations for the return region:

- For text regions:
  - Tn
  - Tn1, ..., Tn
  - Tn1 - Tn2
  - Tn1 and Tn2
  - Tn1 to Tn2
- For table regions:
  - TABLE n, ROW x1
  - TABLE n, ROW x1 and x2
  - TABLE n, ROW x1 to x2
  - TABLE n, ROW x1 - x2

Table 5.8: Valid regions per QA tuple in validation and test datasets.

Model	Validation		Test	
	Valid	Invalid	Valid	Invalid
Mixtral-8x22b	60	00	894	06
GPT-4o mini	57	03	899	01
Gemini 1.5 Flash	57	03	894	06
LLaMA 3 70B	46	14	884	16
Claude 3 Haiku	14	46	818	82

Table 5.8 shows the number of valid regions for each model in both datasets, considering the total number of QA tuples generated. In the validation set, Mixtral-8x22b performed the best, adhering closely to the instructions and formatting provided. Following it, GPT-4o mini and Gemini 1.5 Flash demonstrated similar performance, with only a few formatting errors. At the bottom, LLaMA 3 70B and Claude 3 Haiku exhibited the most formatting mistakes, with Claude 3 Haiku standing out as having the highest error rate, indicating significant issues in meeting the required output format.

In the test set, unlike the validation set, GPT-4o mini performed the best, followed by Mixtral-8x22b and Gemini 1.5 Flash, both showing similar performance with few formatting errors. Once again, LLaMA 3 70B and Claude 3 Haiku were the models with the highest rate of formatting errors.

Although there were slight variations in the number of invalid outputs, the general performance of the models remained consistent. The top three models — Mixtral-8x22b, GPT-4o mini, and Gemini 1.5 Flash — continued to show minimal formatting errors. Meanwhile, LLaMA 3 70B and Claude 3 Haiku maintained their lower performance concerning formatting. For examples of these formatting errors, read the Appendix section 8.3.

5.3.6 Human Evaluation

To evaluate the quality of the questions and answers generated by the models, we conducted a human evaluation with 28 voluntary annotators who were not domain experts.

These annotators were instructed to assess each QA tuple using a web tool developed specifically for this research. They evaluated the coherence of the questions, ensuring they were understandable, adhered to Portuguese language norms, and were free of ambiguity. In addition, if the question was judged as coherent, they validated whether the answers were correct based on the information provided in the documents. The validation process was binary: the annotators responded “yes” or “no” for both criteria. Thus, a question was considered valid if it was coherent and the answer was correct.

The evaluation was conducted in batches of 40 pages (every page contained 3 tuples of QA), with annotators reviewing each page individually. Within each batch, 10 pages were evaluated by two additional annotators to measure inter-annotator agreement. Due to the number of annotators and the total number of available files, some annotators were responsible for more than one batch. To maintain consistency and allow the comparison among annotators, each batch was distributed, ensuring an overlap: each batch was evaluated by at least two distinct annotators.

The average percent inter-annotator agreement is 0.84 (± 0.29) for question coherence and 0.84 (± 0.28) for response accuracy. Appendix 8.4 provides detailed percent inter-annotator agreement data for coherence and accuracy. Based on this high level of agreement and the number of annotators, we retained all annotations and used them to validate the models.

Prior to this evaluation task, a longer one was performed but with one annotator only. This single annotator evaluated a set of 30 pages. This prior task allowed us to plan the following evaluation task, defining the strategy employed. As a result, much of the annotation performed by this single annotator was out of the batches. However, the percent inter-annotator agreement on average between that single annotator and others was 0.84 (± 0.27).

Thus, after validation, we divided the QA tuples into two groups: the golden dataset, consisting of tuples evaluated by three annotators, and the

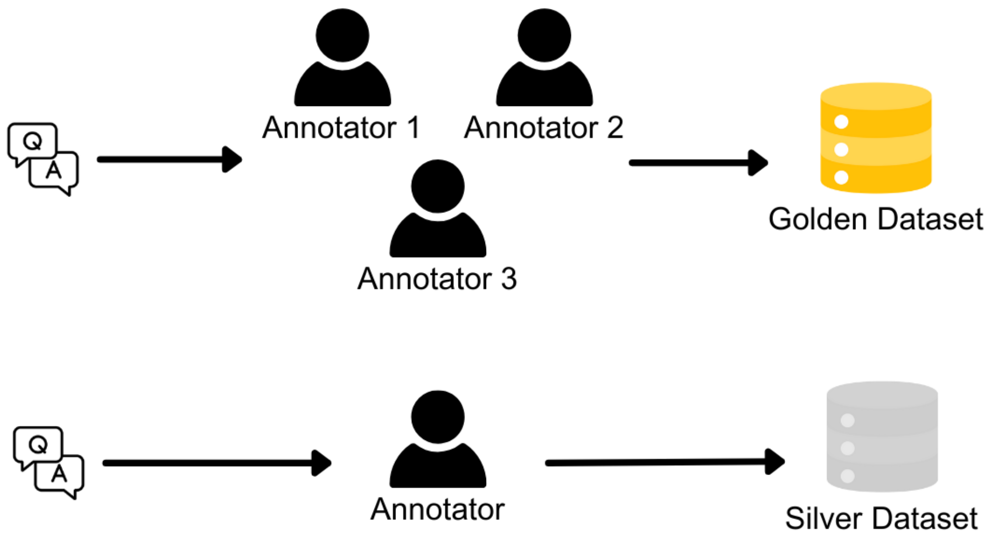


Figure 5.4: Golden and Silver QA datasets construction flow.

silver dataset, containing tuples evaluated by only one annotator, as shown in Figure 5.4. The golden dataset is more reliable, as the tuples were evaluated by three annotators, and the question coherence and answer correctness are given by majority votes.

Table 5.9: Question coherence, answer accuracy, and QA tuples valid proportion for each LLM in the silver and golden datasets.

Model	Question Coherence		Answers Accuracy		QA Tuples Valid	
	Silver	Golden	Silver	Golden	Silver	Golden
LLaMA 3 70B	0.87	0.89	0.96	**0.97**	0.83	**0.87**
Mixtral-8x22B	**0.91**	0.89	0.97	0.95	**0.88**	0.85
Gemini 1.5 Flash	0.88	**0.93**	0.96	0.91	0.84	0.84
GPT-4o mini	0.87	0.83	0.96	**0.97**	0.84	0.81
Claude 3 Haiku	0.88	0.85	**0.98**	0.94	0.86	0.79

Using these datasets, we evaluated the model's performance in terms of the proportion of coherent questions and correct answers to the total of evaluated QA tuples. Table 5.9 presents the performance of models on both datasets. The results show that LLaMA 3 70B has 0.87 of the tuples generated in the Silver Dataset that are coherent. Of these coherent questions, 0.96 have correct answers. Thus, for every model, the final collection of tuples in both datasets was composed of those tuples for which the question was coherent and the answer accurate.

Based on the results, the models present a low coherence variation in the silver dataset, in which Mixtral-8x22B presents the best performance.

However, the same behavior does not occur in the golden dataset, where for the GPT-4o mini and the Claude 3 Haiku, the coherence was well below Gemini 1.5 Flash, the best model. Another model that deserves to be highlighted is LLaMA 3 70B, which presented a good coherence proportion in both datasets with a low variation between them. In relation to answer accuracy on the silver dataset, the models presented high performance and low variation. The performance stayed high on the golden dataset, but the Gemini 1.5 Flash made more mistakes in the silver dataset, dropping its performance.

Based on the proportion, the Mixtral-8x22B and LLaMA 3 70B proved to be the best models for the Silver and Golden datasets, respectively. However, the LLaMA 3 70B performance drops when the model generated more QA tuples. The opposite is true for Mixtral-8x22B, in which generating more QA tuples the model's performance remained stable. The GPT-4o mini and Gemini 1.5 Flash showed similar behavior to Mixtral-8x22B, maintaining their respective performance with more QA tuples generated. Finally, the Claude 3 Haiku is the model with the highest variation between the two datasets, indicating a randomness in its generation process based on the quantity and variation of pages used to generate questions.

Table 5.10: Number of questions generated for each region type by LLM.

Model	Text		Table	
	Silver	Golden	Silver	Golden
LLaMA 3 70B	170	12	616	72
Mixtral-8x22b	112	10	686	74
Gemini 1.5 Flash	118	06	674	78
GPT-4o mini	093	16	705	68
Claude 3 Haiku	077	12	721	72

Table 5.11: Number of incoherent questions generated for each region type by LLM.

Model	Text		Table	
	Silver	Golden	Silver	Golden
LLaMA 3 70B	23	1	80	08
Mixtral-8x22b	12	0	61	09
Gemini 1.5 Flash	17	0	78	06
GPT-4o mini	06	2	96	12
Claude 3 Haiku	04	0	96	13

Table 5.10 presents the number of QA tuples generated for each region type by LLM. Each model generated a specific number of QA tuples for the different region types. The table region received the most focus, as instructed in

Table 5.12: Number of incorrect answers given for each region type by LLM.

Model	Text		Table	
	Silver	Golden	Silver	Golden
LLaMA 3 70B	05	0	21	2
Mixtral-8x22b	03	0	19	4
Gemini 1.5 Flash	10	0	17	7
GPT-4o mini	06	1	22	1
Claude 3 Haiku	05	0	06	4

the generation prompt, due to both its complexity and the significant challenge it poses for the proposed annotation process.

To better understand the values of proportion of valid QA tuples obtained, we check the main mistakes made by each model. Table 5.11 shows the number of incoherent questions generated according to the region in which it is grounded. The main problem lies in interpreting the tables to generate a coherent and unambiguous question. Table 5.12 shows the number of incorrect answers given to coherent questions. Just like the coherence of the question, the regions of the table concentrate most of the wrong questions. This problem is not only related to the table interpretability of each model; errors in recognizing the structure have a direct impact on the model's capability, as it is the only representation of the table visualized by LLMs. So, part of the problem might be inherited from the Table Structure Recognition.

5.3.7 Ablation Test

To understand the impact of each instruction provided in the prompt, we conducted an ablation test during the question generation stage. The first test aimed to evaluate the effect of removing explanations on the use of interrogative pronouns and the resulting variation in their usage. For this test, we removed the explanations from the prompt and generated QA tuples using the validation dataset. To better understand the explanations regarding the use of interrogative pronouns, refer to the Appendix Section 8.1.

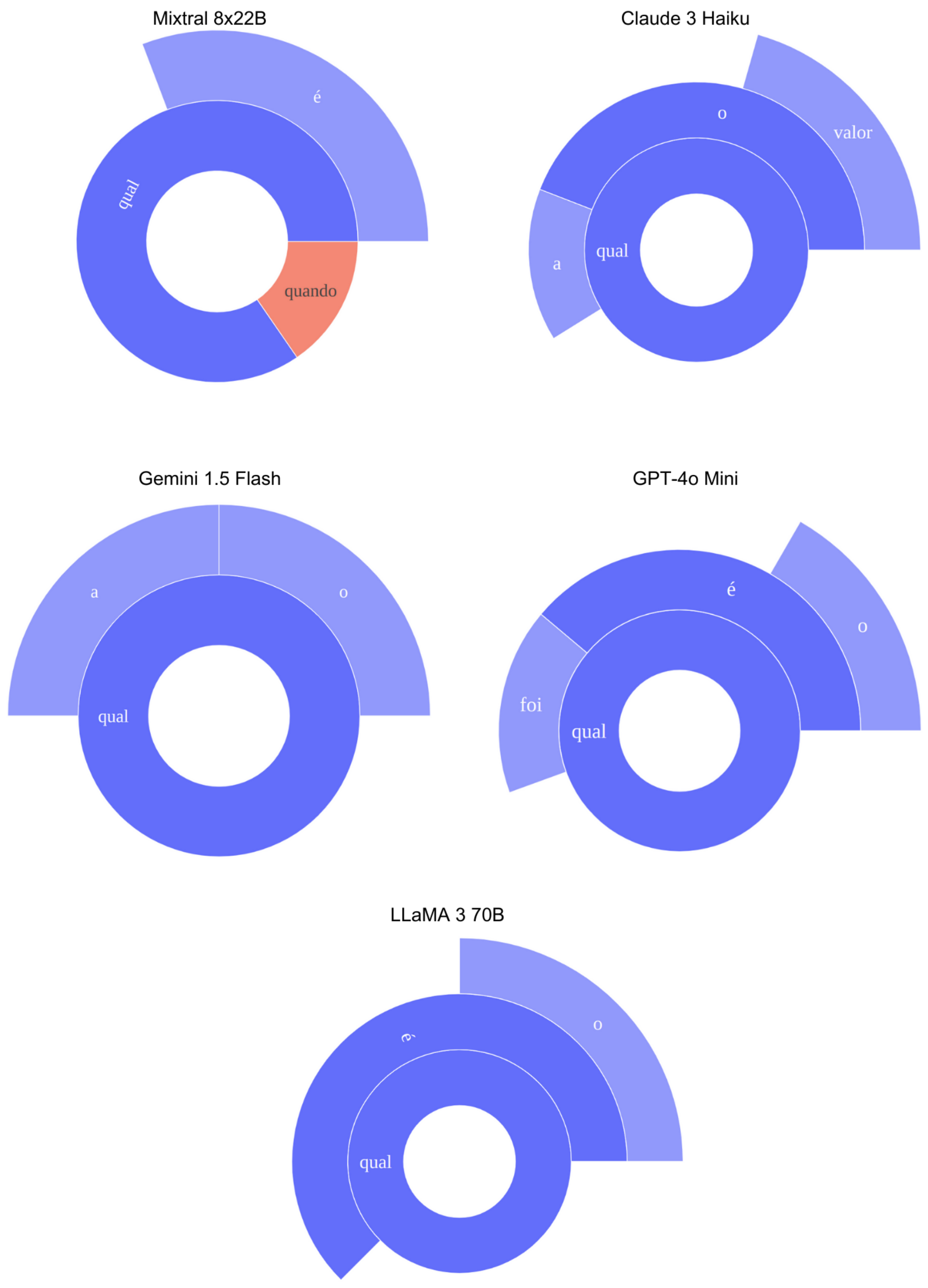


Figure 5.5: Distribution of questions by their starting 4-grams in ablation test using validation dataset.

Figure 5.5 displays the results of this experiment. We observed that, without the explanations, the models overwhelmingly favored the use of *"Qual"* (What). This indicates that the instruction is crucial for ensuring a diverse set of interrogative pronouns in the generated questions.

The second test focused on assessing the impact of explanations regarding

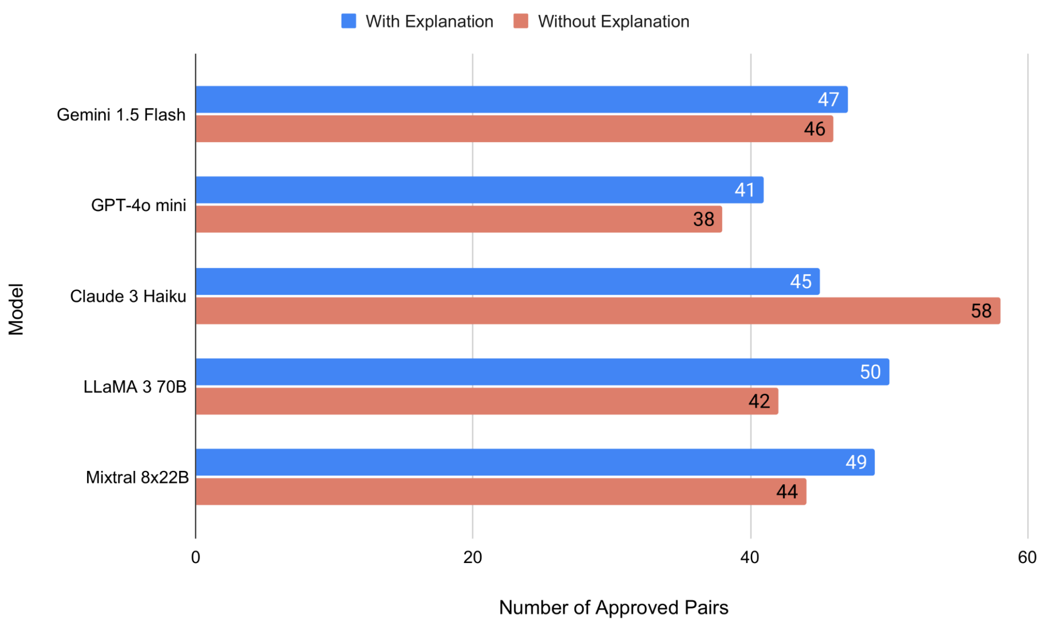


Figure 5.6: Impact of coherence explanation in ablation test, using the validation dataset.

coherence in Portuguese. This test was performed by removing all coherence-related instructions, described in the Appendix Section 8.1, from the prompt and comparing the percentage of coherent questions generated, both with and without these explanations.

Figure 5.6 illustrates each model's number of approved QA tuples. The results indicate that providing coherence-related instructions generally improves model performance, increasing the proportion of coherent questions. However, an exception was found with the Claude 3 Haiku model, which showed an increase in coherence even when no explanation was provided.

5.3.8 Generation Costs

As all models are consumed by a pay-per-use API, we evaluate the total cost of generation process. To calculate the value, we use the entire test dataset. Table 5.13 shows the amount of input and output tokens spent. Moreover, the table contains the value spent in USD to generate the questions.

The most economical model is the GPT-4o mini, which, due to its coding strategy, represented the input data with a smaller number of tokens. However, the model generated a large number of tokens in its output, but not enough to make it more expensive. Next up is Gemini 1.5 Flash, which has a good input and output encoder. However, the input encoder is not as good as the GPT-4o mini, requiring more tokens to represent the input, increasing the final cost.

The most expensive models are the LLaMA 3 70B and the Mixtral-8x22B.

Table 5.13: Number of input and output tokens and total spent to generate 3 questions for 300 pages for each LLM. Final cost calculated based on model prices in August 2024.

Model	Input Tokens	Output Tokens	Final Cost
GPT-4o mini	905,317	53,189	0.16
Gemini 1.5 Flash	911,450	45,205	0.27
Claude 3 Haiku	1,087,023	53,189	0.34
LLaMA 3 70B	979,342	40,912	0.63
Mixtral 8x22B	1,206,217	51,125	1.64

The LLaMA has a good input and output encoder, but the cost of consuming the API used makes the process more expensive. On the other hand, Mixtral-8x22B consumes more input and output tokens and is the model with the highest cost per token among the generation models.

5.4 Question-Answer Judgment

This section presents the criteria used to select the LLMs for the Question-Answer Judgment stage. We also describe the datasets employed for the development and evaluation of this stage and report the results for each model.

5.4.1 Model Selection

For the QA judgment stage, we selected models primarily based on quality. This stage is critical to our process, as it serves as a filter to automatically discard invalid questions. To ensure accurate assessment of the generated QA tuples, we needed models with strong performance in general tasks and the ability to effectively comprehend document transcriptions. This ensures reliable evaluations and proper usage of the generated tuples.

At this stage, the main cost comes from input tokens rather than output tokens, as the instruction prompt is resent for each QA tuple evaluation. On average, the input size was 2261.33 ± 819.75 tokens, while the output averaged 2.67 ± 1.72 tokens. We selected models from the first quadrant based on the models shown in Figure 5.1, which illustrates the relationship between model quality and cost. In addition to these three models – GPT-4o, Claude 3.5 Sonnet and Gemini 1.5 Pro – we included LLaMA 3.1 405B, one of the most advanced open-source models currently available, for comparison between commercial and open-source models.

5.4.2 Datasets

To evaluate this stage, we used two datasets. The first was the validation set created for prompt development in the QA generation stage, described in Subsection 5.3.2. This dataset helped refine the prompt used in this stage and was submitted to human evaluation by a non-domain expert annotator.

The second dataset was the golden dataset, created after a human evaluation during the QA generation stage, as detailed in Subsection 5.3.6. We used this dataset to assess the generalization of judgment capabilities of the proposed approach by comparing the performance of an LLM and a human annotator.

5.4.3 Experimental Setup

To evaluate each model, we standardized the API calls and output metadata using Langchain. For consistency, we set the temperature to zero for all models to ensure deterministic outcomes. All other parameters were left at their default settings for each respective model.

5.4.4 Evaluating the LLM Judge

The LLM judgment of QA tuples followed the same criteria as the human evaluation. Each model assessed the coherence of the questions, ensuring they were understandable, adhered to the norms of the Portuguese language, and were free of ambiguity. Additionally, the models validated whether the answers were accurate based on the document transcriptions. The judgment process was binary, with the LLM responding "yes" or "no" to both criteria. A question was marked as valid if it was coherent and the answer was accurate.

After running all models on the validation dataset, we compared the F1 scores of each LLM model and their performance against human judgments for final dataset utilization. Figure 5.7 shows the results for each model on the validation dataset. All models performed well, with F1 scores above 0.8. The best performance came from LLaMA 3.1 405B, which achieved an F1 score of 0.97, followed by Gemini 1.5 Pro with 0.95. The lowest performer was GPT-4o, scoring 0.89.

Figure 5.8 shows the confusion matrix comparing human votes to each LLM. The lower F1 scores for GPT-4o and Claude 3.5 were caused by a large number of false negatives, though these models did not generate false positives. In contrast, Gemini 1.5 Pro and LLaMA 3.1 405B, despite their better F1

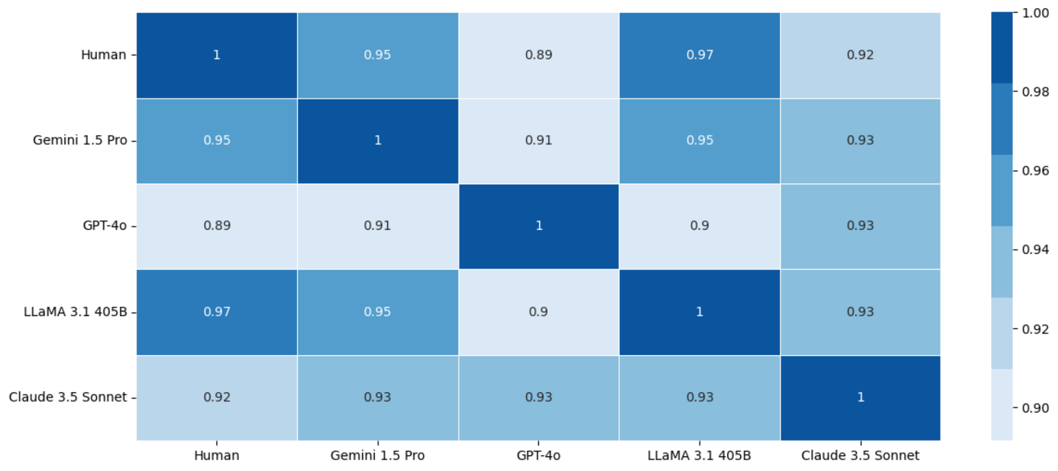


Figure 5.7: F1 score between LLMs and humans for the validity of a QA tuple on the validation dataset.

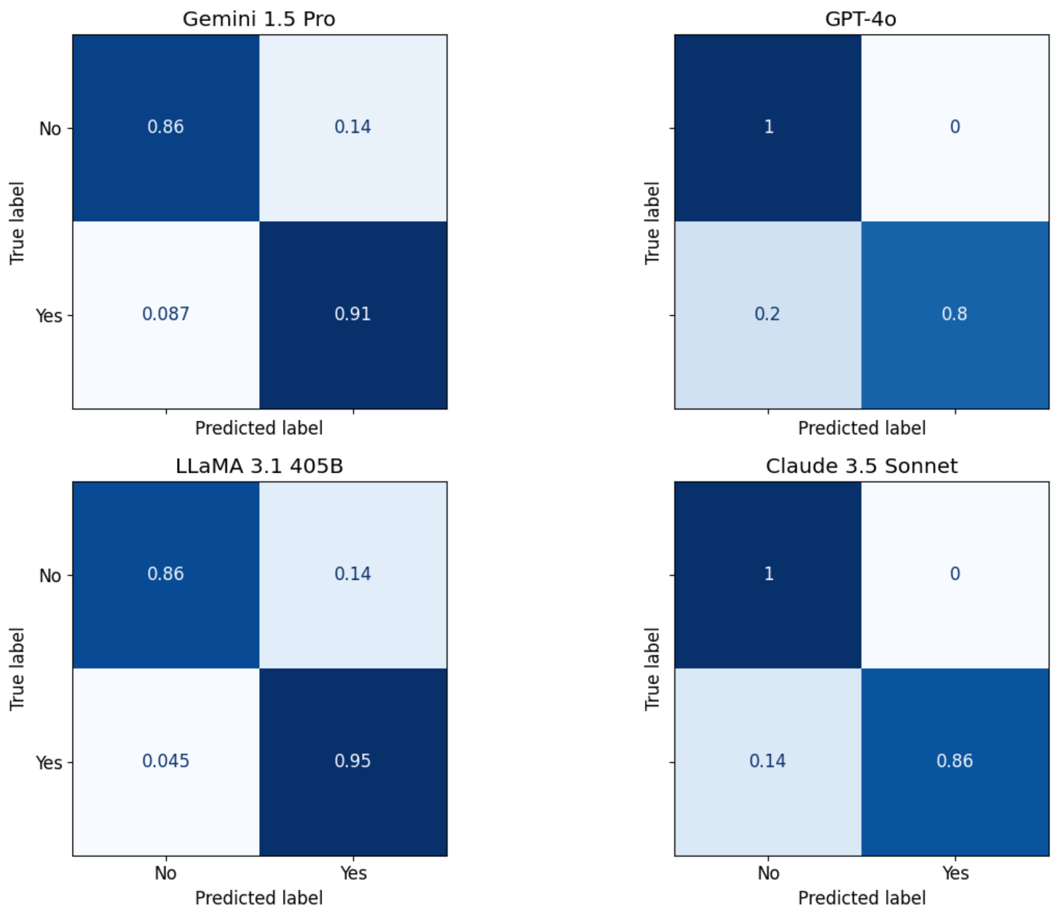
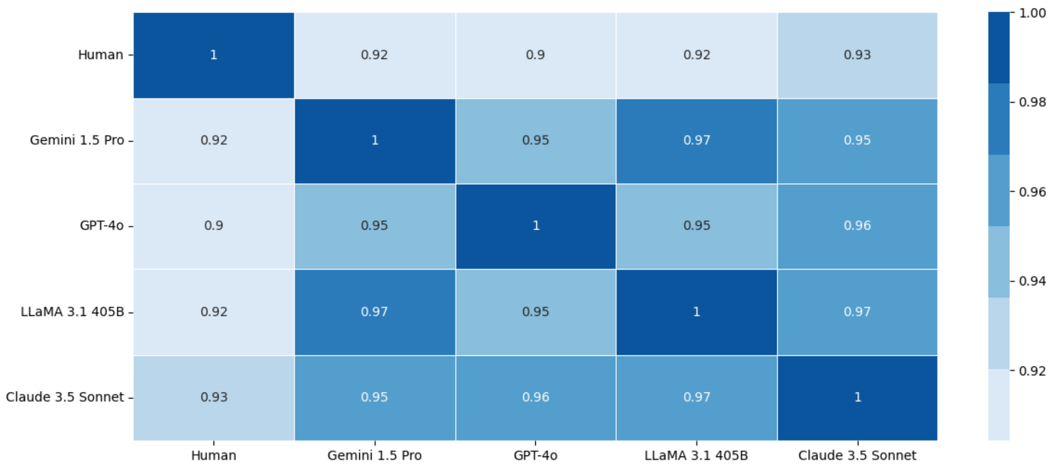


Figure 5.8: Confusion matrix between LLM and human for the validity of a QA tuple on the validation dataset. The human annotation is considered the true label.

scores, did produce false positives—problematic since this stage aims to prevent invalid questions from contaminating the final dataset.

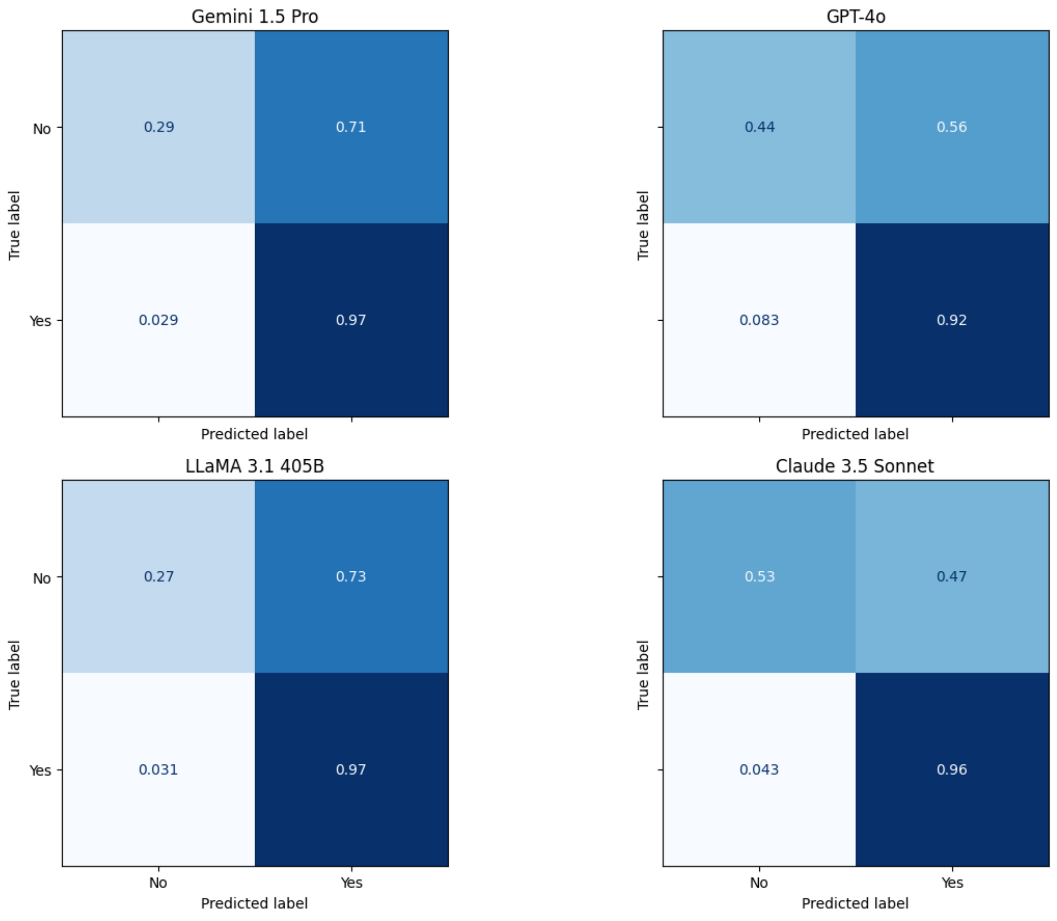
On the golden dataset, the performance of LLaMA 3.1 405B and Gemini



	Human	Gemini 1.5 Pro	GPT-4o	LLaMA 3.1 405B	Claude 3.5 Sonnet
Human	1	0.92	0.9	0.92	0.93
Gemini 1.5 Pro	0.92	1	0.95	0.97	0.95
GPT-4o	0.9	0.95	1	0.95	0.96
LLaMA 3.1 405B	0.92	0.97	0.95	1	0.97
Claude 3.5 Sonnet	0.93	0.95	0.96	0.97	1

Figure 5.9: F1 score between LLMs and humans for the validity of a QA tuple on the golden dataset.

1.5 Pro dropped slightly, with both achieving an F1 score of 0.92, as shown in Figure 5.9. However, Claude 3.5 Sonnet and GPT-4o showed stable performance, each increasing by 0.01, making Claude 3.5 Sonnet the best performer.



Model	True No → Pred No	True No → Pred Yes	True Yes → Pred No	True Yes → Pred Yes
Gemini 1.5 Pro	0.29	0.71	0.029	0.97
GPT-4o	0.44	0.56	0.083	0.92
LLaMA 3.1 405B	0.27	0.73	0.031	0.97
Claude 3.5 Sonnet	0.53	0.47	0.043	0.96

Figure 5.10: Confusion matrix between LLM and human for the validity of a QA tuple on the golden dataset. The human annotation is considered the true label.

The confusion matrix for the golden dataset, shown in Figure 5.10, reveals some divergences from the validation set. False negatives decreased for models like GPT-4o and Claude 3.5 Sonnet, while false positives increased across all models. This increase is related to the larger proportion of negative samples in the golden set, making it more diverse and challenging to detect invalid tuples.

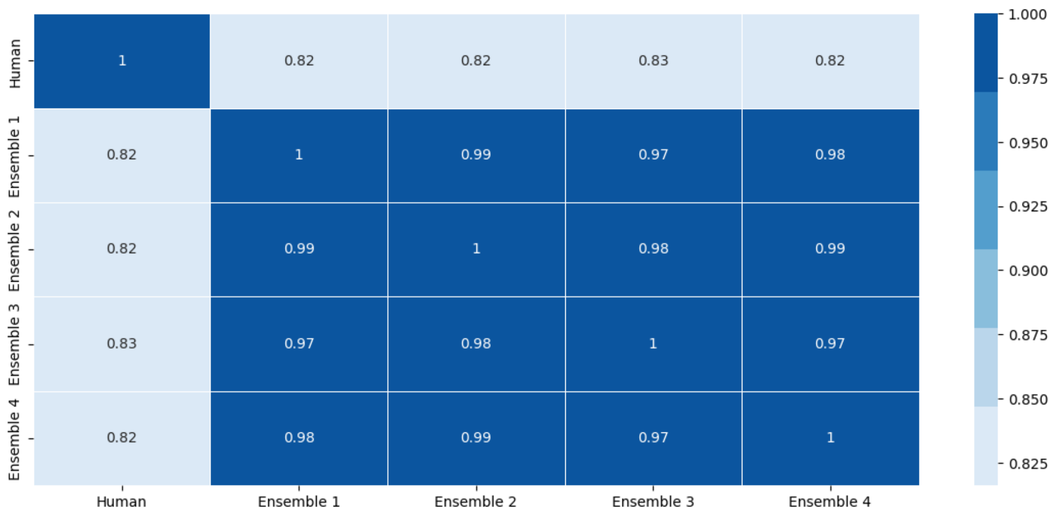


Figure 5.11: F1 score between LLMs and humans for the validity of a QA tuple on the golden dataset. Where Ensemble 1: Gemini 1.5 Pro + GPT-4o + LLaMA 3.1 405B; Ensemble 2: Gemini 1.5 Pro + GPT-4o + Claude 3.5 Sonnet; Ensemble 3: Gemini 1.5 Pro + LLaMA 3.1 405B + Claude 3.5 Sonnet; Ensemble 4: GPT-4o + LLaMA 3.1 405B + Claude 3.5 Sonnet.

To improve judgment performance, we combined the models into an ensemble. In this approach, a QA tuple was considered valid only if all models confirmed both the coherence of the question and the accuracy of the answer. Figure 5.11 shows the F1 obtained by each ensemble, where we can see that the combination of models did not improve the metric, showing a similar performance for all combinations.

The confusion matrix for this ensemble, shown in Figure 5.12, indicates a slight reduction in false negatives. However, the same judgment errors were common across models, suggesting a need for further investigation into the source of the problem.

5.4.5 Judgment Costs

Since all models were accessed via pay-per-use APIs, we also evaluated the total cost of the judgment process using the golden dataset. Table 5.14 shows the number of tokens consumed and the cost in USD per QA tuple. LLaMA 3.1 405B was the most cost-effective model, its final cost was much

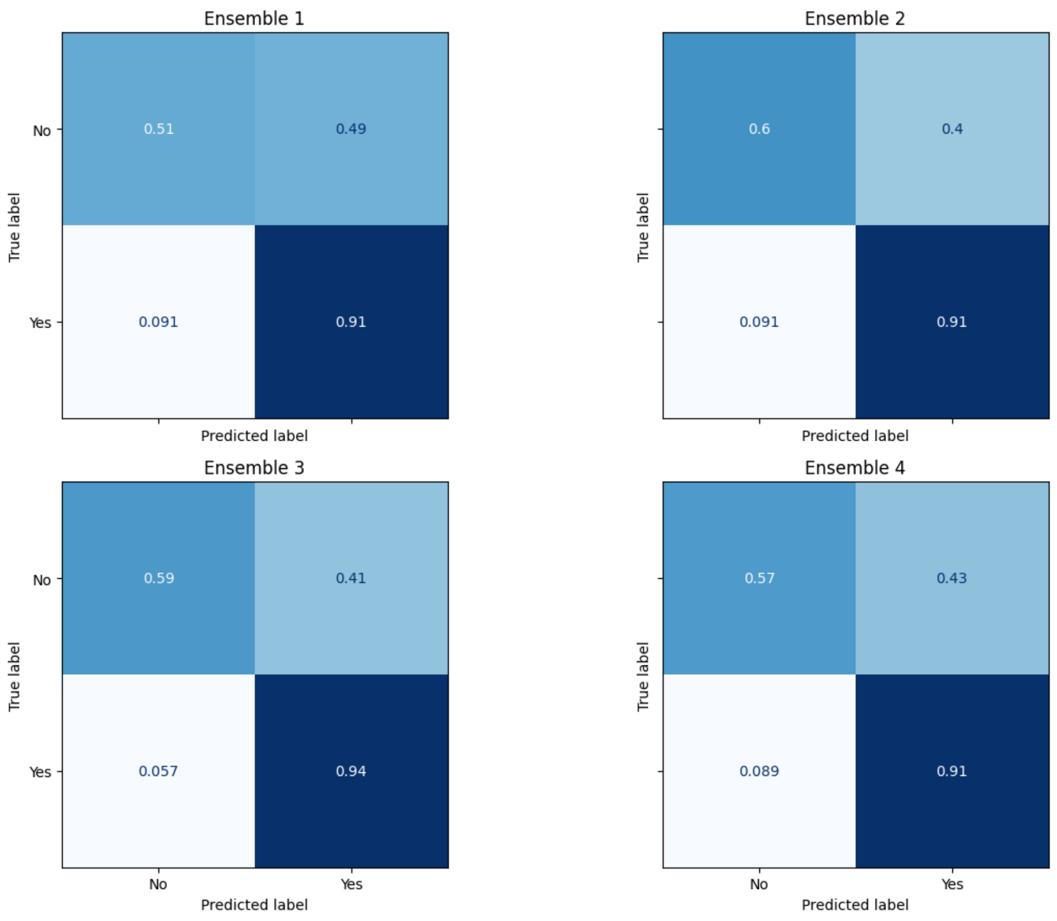


Figure 5.12: Confusion matrix between Ensemble and human for the validity of a QA tuple on the golden dataset. The human annotation is considered the true label.

lower than other commercial models. Gemini 1.5 Pro and Claude 3.5 Sonnet had similar costs, \$9.24 and \$9.51, respectively, though they differed greatly in the number of tokens required for input and output.

Table 5.14: Number of input and output tokens and final cost (USD) for each LLM to judge 420 QA tuples on the golden dataset. Final cost calculated based on model prices in August 2024.

Model	Input Tokens	Output Tokens	Final Cost
LLaMA 3.1 405B	2,946,672	3,867	08.85
Gemini 1.5 Pro	2,637,177	1,350	09.24
Claude 3.5 Sonnet	3,136,644	6,612	09.51
GPT-4o	2,676,624	1,725	13.41

The most expensive model was GPT-4o which, despite not consuming many input and output tokens, presents a high cost for each of them, making it the most expensive to use. Although not shown in the table, the ensemble was even more costly, with the combined model cost totaling \$ 27.6 in the best case.

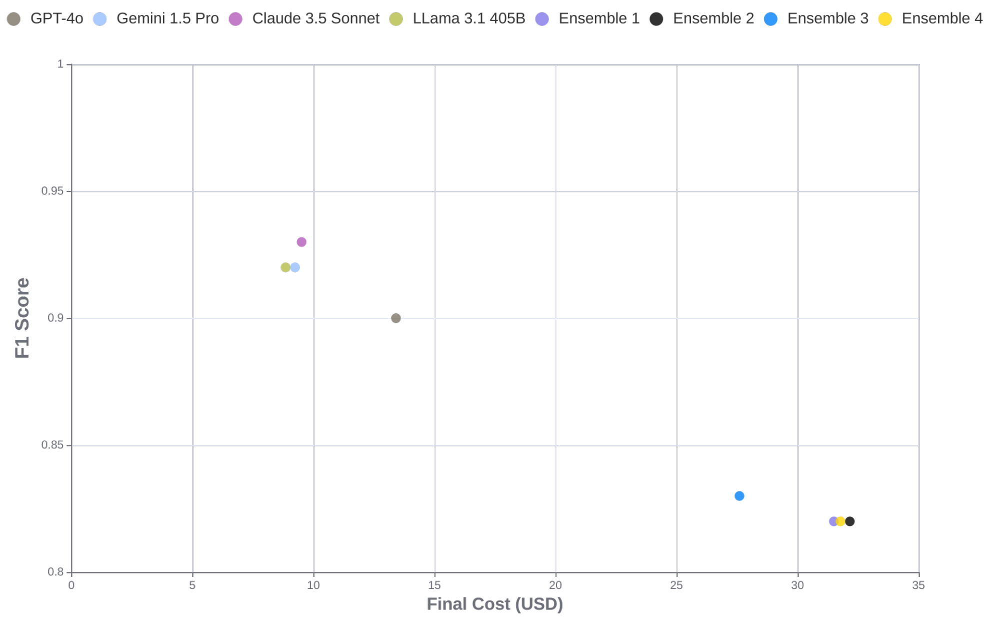


Figure 5.13: Final Cost (USD) x F1 score for each LLM and combinations to judge 420 QA tuples on the golden dataset.

Figure 5.13 compares the cost-performance proportion for each model and the ensemble. This graph shows that combining models leads to high costs for relatively low performance. Additionally, GPT-4o stands out for its high cost, given its comparatively low performance. Finally, we found that LLaMA 3.1 405B offers the best cost-effectiveness, delivering excellent judgment performance at a highly competitive price.

6
Conclusion

This chapter presents our final considerations about our work presented in this dissertation. In Section 1 at the Introduction, we posed the main research question (*How can we use Large Language Models in data annotation for Document Visual Question Answering task?*) we aimed to answer. As an attempt to respond, we developed a process for data annotation using LLM for the DocVQA task, as presented in Chapter 4.

To answer the first part of **RQ1** (*How can we combine computer vision models and Large Language Models to generate questions and answers from documents?*), we developed a process composed of three steps, where we used different CV models to extract the textual representation of documents. In the first step, we applied an OCR to recognize the words on the page, obtaining your coordinates and text. In the second step, we applied a DLA model to analyze the document layout, making it possible to map the information structures contained in the document. In the third step, we used a TSR model in the tables regions detected in the previous step to generate the markup representation of them. All three steps are described in Section 4.2.

For the second part of **RQ1**, we evaluated the performance of five cost-effective LLMs in generating questions and answers based on the transcription of the documents and following the generation rules defined in the work. The rules were defined in such a way as to ensure that the questions were well-formed and made human evaluation possible.

To answer the **RQ2** (*How can we evaluate the quality of the generated questions and answers?*), we carried out two separate steps. The first step is a human evaluation of QA pairs generated by each LLM to assess the capability of following the instructions established for generation and to measure the ratio between the number of valid and invalid pairs. To automate the validation of the question pairs generated, we evaluated the performance of four robust LLMs in judging the validity of these pairs, using the result of the human evaluation as a benchmark.

Finally, our findings show that LLMs are able to generate QA pairs with a good ratio between the validity and invalidity of each pair. Our best results are obtained by Mixtral-8x22B and GPT-4o mini, the models that showed the most significant variability in the questions generated, with a low rate of invalid QA pairs and almost perfectly following the instructions given. In terms of cost-effectiveness, the GPT-4o mini stands out even more, with the lowest price for generating QA pairs.

The best model for the automatic judgment of QA pairs is Claude 3.5 Sonnet, with an F1 score of 0.93 compared to human evaluations. Although the score was obtained, the stage needs more improvements to reduce the number of false negatives generated.

As an attempt at improvement, we tested the use of ensemble models, which had a lower F1 score than the individual models, 0.83 F1 score in the best case, and almost three times more expensive than the best model. Despite reducing the number of false positives slightly, the final cost of combining models does not compensate for the low delivery of results.

In the following section, we state some future works and enhancements.

6.1 Future Works

As discussed earlier, the process occasionally fails in the validity judgment of QA pairs. To address this, we plan to introduce a review score for each generated pair in future work. This score would guide whether a question requires human review, aiming to reduce false positives and negatives. Beyond enhancing the robustness of the dataset, this score would help avoid resource loss caused by incorrect generation or judgment.

Since our work focused solely on determining question validity, we intend to expand this analysis by conducting a qualitative assessment of the questions. This future study will help us evaluate which questions are most relevant and well-formed for the target domain.

Given that the annotation process relies entirely on document transcription, we also aim to analyze the impact of transcription errors on the question-generation process. This analysis will help us understand how models handle these failures and propose strategies to mitigate their impact.

Another future direction involves adding a step in the transcription stage to handle figures within documents. We will focus on graphic regions, for which we will create textual representations that LLMs can interpret. This step is necessary since not all models are multimodal.

We plan to explore sample balancing strategies to address variability issues in the QA pairs generated by some models. These strategies will be incorporated into the prompt design to encourage the generation of a wider variety of questions.

Currently, our tests are limited to generating QA pairs from a single page at a time. In the future, we plan to experiment with processing multiple pages simultaneously, considering the token limits of each LLM model.

We also aim to optimize the cost of generation and evaluation by creating workflows that direct specific types of pages to the most suitable model. By identifying each model's strengths — such as handling tables or complex document structures — we can minimize invalid annotations. As part of this improvement, we will explore the use of Knowledge Graphs (CHEN; JIA; XIANG, 2020) to create more efficient document representations, thereby reducing the amount of information passed to the LLMs.

Our ultimate goal is to create a fully annotated dataset using the proposed process and establish baselines with leading VQA models. This step will allow us to identify our process's strengths and weaknesses and guide future enhancements.

After defining baselines, we will select an optimization model. This optimization will involve adjustments to the architecture and workflow to improve the efficiency of response generation based on input documents.

Finally, we intend to propose a tool to annotate the data from the TSR task, since we have not found any tool that helps to annotate the three steps needed to correctly represent the image of the table as HTML. With this tool we can create an annotated dataset using the worked domain, to better validate the selected models.

7 Bibliography

ABDULLAH, O.; MADAIN, R.; JARARWEH, Y. Large language models and their impact on nlp. In: **Proceedings of the 2022 International Conference on Natural Language Processing (NLP)**. [S.l.: s.n.], 2022. p. 12–24.

AGRAWAL, M. et al. Large language models are few-shot clinical information extractors. In: GOLDBERG, Y.; KOZAREVA, Z.; ZHANG, Y. (Ed.). **Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing**. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022. p. 1998–2022. Disponível em: <https://aclanthology.org/2022.emnlp-main.130>.

AGUDA, T. et al. Large language models as financial data annotators: A study on effectiveness and efficiency. **arXiv preprint arXiv:2403.18152**, 2024.

AI, M. **Cheaper, Better, Faster, Stronger**. 2024. Disponível em: <https://mistral.ai/news/mixtral-8x22b/>.

AI, M. **Mistral AI team**. 2024. Disponível em: <https://mistral.ai/company/>.

AI@META. **About AI at Meta**. 2024. <https://ai.meta.com/about/>.

AI@META. **Llama 3 Model Card**. 2024. Disponível em: <https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md>.

ANALYSIS, A. **Independent Analysis of AI models and API providers**. 2024. <https://artificialanalysis.ai/>.

ANTHROPIC. **Anthropic: AI Safety and Research**. 2024. <https://www.anthropic.com>.

ANTHROPIC. **Anthropic Models**. 2024. <https://docs.anthropic.com/en/docs/about-claude/models>.

ANTHROPIC. **What Is Claude 3.5 Sonnet?** 2024. <https://beginswithai.com/claude-3-5-sonnet/#:~:text=A%20brief%20Timeline%20of%20Claude,Released%20on%20June%2020%2C%202024>.

BAI, Y. et al. Benchmarking foundation models with language-model-as-an-examiner. **Advances in Neural Information Processing Systems**, v. 36, 2024.

BANERJEE, A. et al. Swindocsegmenter: An end-to-end unified domain adaptive transformer for document instance segmentation. In: SPRINGER. **International Conference on Document Analysis and Recognition**. [S.l.], 2023. p. 307–325.

BROWN, T. et al. Language models are few-shot learners. **Advances in neural information processing systems**, v. 33, p. 1877–1901, 2020.

CAO, J.; MA, Y.; LI, J. Docbank: A benchmark dataset for document layout analysis. In: **Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [S.l.: s.n.], 2021. p. 978–993.

CHEN, M. et al. Evaluating large language models trained on code. 2021.

CHEN, X.; JIA, S.; XIANG, Y. A review: Knowledge reasoning over knowledge graph. **Expert systems with applications**, Elsevier, v. 141, p. 112948, 2020.

CHEN, Y. et al. Rodla: Benchmarking the robustness of document layout analysis models. **arXiv preprint arXiv:2403.14442**, 2024.

CHI, Z.; LIU, L.; XU, Y. Table structure recognition with conditional attention networks. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2019. p. 715–724.

DEEPMIND, G. **About Google DeepMind**. 2024. <https://deepmind.google/about/>.

DESAI, P. Understanding and leveraging large language models in nlp. In: **Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [S.l.: s.n.], 2024. p. 543–556.

DING, Y. et al. Vqa: A new dataset for real-world vqa on pdf documents. In: SPRINGER. **Joint European Conference on Machine Learning and Knowledge Discovery in Databases**. [S.l.], 2023. p. 585–601.

DONG, Q. et al. A survey on in-context learning. **arXiv preprint arXiv:2301.00234**, 2022.

DUBEY, A. et al. **The Llama 3 Herd of Models**. 2024. Disponível em: <https://arxiv.org/abs/2407.21783>.

EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. **International journal of computer vision**, Springer, v. 88, n. 2, p. 303–338, 2010.

GÖBEL, M. et al. Icdar 2013 table competition. In: IEEE. **2013 12th International Conference on Document Analysis and Recognition**. [S.l.], 2013. p. 1449–1453.

HENDRYCKS, D. et al. Measuring massive multitask language understanding. **arXiv preprint arXiv:2009.03300**, 2020.

HENDRYCKS, D. et al. Measuring mathematical problem solving with the math dataset. **NeurIPS**, 2021.

JIANG, A. Q. et al. Mistral 7b. **arXiv preprint arXiv:2310.06825**, 2023.

JOCHER, G.; CHAURASIA, A.; QIU, J. **Ultralytics YOLOv8**. 2023. Disponível em: <https://github.com/ultralytics/ultralytics>.

KITTUR, A.; CHI, E. H.; SUH, B. Crowdsourcing user studies with mechanical turk. In: **Proceedings of the SIGCHI conference on human factors in computing systems**. [S.l.: s.n.], 2008. p. 453–456.

LangChain. **LangChain Documentation**. 2024. <https://www.langchain.com>. Accessed: 2024-06-10.

LEVENSHTEIN, V. I. et al. Binary codes capable of correcting deletions, insertions, and reversals. In: SOVIET UNION. **Soviet physics doklady**. [S.l.], 1966. v. 10, n. 8, p. 707–710.

LI, M. et al. **TableBank: A Benchmark Dataset for Table Detection and Recognition**. 2019.

LI, R. et al. Extending context window in large language models with segmented base adjustment for rotary position embeddings. **Applied Sciences**, MDPI, v. 14, n. 7, p. 3076, 2024.

LY, N. T.; TAKASU, A. An end-to-end multi-task learning model for image-based table recognition. SciTePress, p. 626–634, 2023.

Maik Thiele. **documentlayoutsegmentation_YOLOv8_ondoclaynet (Revision 25486d5)**. Hugging Face, 2024. Disponível em: <https://huggingface.co/DILHTWD/documentlayoutsegmentation_YOLOv8_ondoclaynet>.

Malaysia-AI. **YOLOv8X-DocLayNet-Full-1024-42**. Hugging Face, 2024. Disponível em: <https://huggingface.co/malaysia-ai/YOLOv8X-DocLayNet-Full-1024-42>.

MARINAI, S. Introduction to document analysis and recognition. In: **Machine learning in document analysis and recognition**. [S.l.]: Springer, 2008. p. 1–20.

MATHEW, M. et al. Infographicvqa. In: **Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision**. [S.l.: s.n.], 2022. p. 1697–1706.

MATHEW, M. et al. Docvqa: A dataset for vqa on document images. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2021. p. 2200–2209.

NGUYEN, P. et al. Tabiqa: Table questions answering on business document images. **arXiv preprint arXiv:2303.14935**, 2023.

OPENAI. **About OpenAI**. 2024. <https://openai.com/about>.

OPENAI. **OpenAI API Documentation**. 2024. <https://platform.openai.com/docs>.

PAWLIK, M.; AUGSTEN, N. Tree edit distance: Robust and memory-efficient. **Information Systems**, Elsevier, v. 56, p. 157–173, 2016.

PFITZMANN, B. et al. Doclaynet: a large human-annotated dataset for document-layout segmentation. In: **Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 2022. p. 3743–3751.

PFITZMANN, B. et al. **IBM DocLayNet Labeling Guide**. Säumerstrasse 4, 8803 Rüschlikon, Switzerland, 2022. © IBM Corporation 2021, 2022. Disponível em: <https://github.com/DS4SD/DocLayNet/blob/main/assets/DocLayNet_Labeling_Guide_Public.pdf>.

QIAO, L. et al. Lgpma: complicated table structure recognition with local and global pyramid mask alignment. In: SPRINGER. **International conference on document analysis and recognition**. [S.l.], 2021. p. 99–114.

RADFORD, A. et al. Learning transferable visual models from natural language supervision. In: PMLR. **International conference on machine learning**. [S.l.], 2021. p. 8748–8763.

RAJA, S.; MONDAL, A.; JAWAHAR, C. Table structure recognition using top-down and bottom-up cues. In: SPRINGER. **Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16**. [S.l.], 2020. p. 70–86.

REIN, D. et al. Gpqa: A graduate-level google-proof q&a benchmark. **arXiv preprint arXiv:2311.12022**, 2023.

SANTOS, Y.; SILVA, M.; REIS, J. C. S. Evaluation of optical character recognition (ocr) systems dealing with misinformation in portuguese. In: **2023 36th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.: s.n.], 2023. p. 223–228.

SARAVIA, E. Prompt Engineering Guide. **https://github.com/dair-ai/Prompt-Engineering-Guide**, 12 2022.

SHAZEER, N. et al. **Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer**. 2017. Disponível em: <https://arxiv.org/abs/1701.06538>.

SHI, F. et al. Language models are multilingual chain-of-thought reasoners. **URL https://arxiv. org/abs/2210.03057**, 2022.

SMOCK, B.; PESALA, R.; ABRAHAM, R. Pubtables-1m: Towards comprehensive table extraction from unstructured documents. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 4634–4642.

SUBRAMANI, N. et al. A survey of deep learning approaches for ocr and document understanding. **arXiv preprint arXiv:2011.13534**, 2020.

TEAM, G. et al. **Gemini: A Family of Highly Capable Multimodal Models**. 2024. Disponível em: <https://arxiv.org/abs/2312.11805>.

TEAM, G. et al. **Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context**. 2024. Disponível em: <https://arxiv.org/abs/2403.05530>.

TOUVRON, H. et al. Llama: Open and efficient foundation language models. **arXiv preprint arXiv:2302.13971**, 2023.

VASWANI, A. et al. Attention is all you need. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2017. p. 5998–6008.

WANG, S. et al. Want to reduce labeling cost? GPT-3 can help. In: **Findings of the Association for Computational Linguistics: EMNLP 2021**. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021. p. 4195–4205. Disponível em: <https://aclanthology.org/2021.findings-emnlp.354>.

WEI, J. et al. Chain-of-thought prompting elicits reasoning in large language models. **Advances in neural information processing systems**, v. 35, p. 24824–24837, 2022.

WU, Q. et al. Visual question answering: A survey of methods and datasets. **Computer Vision and Image Understanding**, v. 163, p. 21–40, 2017. ISSN 1077-3142. Language in Vision. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1077314217300772>.

YE, J. et al. Pingan-vcgroup's solution for icdar 2021 competition on scientific literature parsing task b: table recognition to html. **arXiv preprint arXiv:2105.01848**, 2021.

ZHANG, H. et al. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. **arXiv preprint arXiv:2203.03605**, 2022.

ZHANG, R. et al. LLMaAA: Making large language models as active annotators. In: **Findings of the Association for Computational Linguistics: EMNLP 2023**. Singapore: Association for Computational Linguistics, 2023. p. 13088–13103. Disponível em: <https://aclanthology.org/2023.findings-emnlp.872>.

ZHENG, X. et al. Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In: **Proceedings of the IEEE/CVF winter conference on applications of computer vision**. [S.l.: s.n.], 2021. p. 697–706.

ZHONG, X.; SHAFIEIBAVANI, E.; YEPES, A. J. Image-based table recognition: data, model, and evaluation. In: SPRINGER. **European conference on computer vision**. [S.l.], 2020. p. 564–580.

ZHONG, X.; TANG, J.; YEPES, A. J. Publaynet: largest dataset ever for document layout analysis. In: IEEE. **2019 International conference on document analysis and recognition (ICDAR)**. [S.l.], 2019. p. 1015–1022.

ZIEGLER, A.; BERRYMAN, M. Hallucinations in large language models: Causes and mitigation strategies. In: **Proceedings of the 2023 Conference on Machine Learning and Natural Language Processing**. [S.l.: s.n.], 2023. p. 321–334.

8 Appendix

8.1 Question-Answer Generation Prompt

In this section of the appendix, the prompt constructed for QA generation is detailed. Beyond the prompt, we describe each positional marker utilized in our approach.

Code 2: System prompt of QA generation stage

```
Você é um sistema especialista no domínio {dominio} capaz de analisar e compreender textos com tabelas. Você deve gerar perguntas coerentes, baseando-se nos seguintes níveis de coerência:

    1. Coerência sintática: Remete à estruturação linguística, como seleção das palavras conforme o contexto, ordenamento lógico das frases e coesão textual. Auxilia no uso adequado dos conectivos e evita a construção de pensamentos ambíguos (incertos).
    2. Coerência semântica: Está relacionada com a composição lógica das ideias, ou seja, elaboração de argumentos ordenados, harmônicos e sem nenhuma contradição. Como a semântica é a parte da linguística que estuda o significado das palavras, a incoerência desfaz as relações de sentido entre as palavras que compõem as partes do texto.
    3. Coerência temática: Todas as frases precisam ser relevantes para o tema discutido. O autor deve privilegiar apenas os argumentos que possibilitam o desenvolvimento de ideias pertinentes e que permitem a compreensão do público leitor.
    4. Coerência pragmática: Refere-se aos textos orais ou escritos que levam em conta a linguagem falada e seus efeitos na comunicação entre os interlocutores. Quando fazemos uma pergunta para outra pessoa, por exemplo, é esperado uma resposta, que pode ser de cunho afirmativo ou negativo. Caso essa regra seja quebrada, acontece o que é definido pela linguística como incoerência pragmática.
    5. Coerência estilística: Diz respeito ao estilo linguístico aplicado no texto, que deve ser mantido do início ao fim. A mistura entre as variedades, como o uso da linguagem coloquial e formal ao mesmo tempo, não interfere na interpretabilidade do conteúdo, mas deve ser evitada porque pode levar facilmente a incoerência textual.
    6. Coerência genérica: É ligada a escolha do gênero textual, que deve dialogar com os acontecimentos discursivos. Por exemplo, se o objetivo de uma pessoa é vender um produto, certamente ela vai adotar uma linguagem persuasiva em seu
```

```
          texto, pois essa é a caraterística desse tipo de gênero. Já
          se a intenção for contar uma história, uma das opções é
          escrevê-la em formato de conto.

Além disso, você deve seguir as seguintes regras para utilização dos
     pronomes e advérbios interrogativos nas perguntas:

    1. "Que?": Refere-se principalmente a coisas, podendo perguntar:
         "que coisa?" ou "que espécie de coisa ou pessoa?".
    2. "Qual?": Refere-se a coisas ou a pessoas. Pode transmitir uma
         ideia de seleção, ou seja, de identificação de um ou vários
         elementos dentro de um grupo.
    3. "Quanto?": Refere-se a coisas ou a pessoas. Transmite uma
        ideia de quantificação ou busca por um valor.
    4. "Quem?": Refere-se principalmente a pessoas ou coisas
        personificadas.
    5. "Quando?": Refere-se a alguma "circunstância" indicando tempo
        .
    6. "Onde?": Refere-se a alguma "circunstância" indicando lugar.
    7. "Como?": Refere-se a alguma "circunstância" indicando o modo.

Você vai receber um texto de referência e deve executar os seguintes
     passos:
    1. FAÇA {qtd_questions} PARES DE PERGUNTAS E RESPOSTAS
        diferentes, utilizando o texto de referência. Foque nas
        regiões de tabelas, que serão informadas em código HTML e
        estarão dentro de um Code Block.
    2. Garanta que a pergunta feita tenha todos os elementos
        necessários para que apresente uma única resposta e de forma
         direta. Não faça perguntas que apresentem duas ou mais
        respostas possíveis dentro da estrutura do texto.
    3. NÃO FAÇA PERGUNTAS que precise de ALGUMA OPERAÇÃO MATEMÁTICA
        para ser resolvida ou que precise de mais de uma linha da
        tabela para ser respondida. Não faça perguntas que fiquem
        sem respostas, todas perguntas feitas devem ter respostas.
    4. As respostas devem ser uma cópia direta do próprio texto
        informado.
    5. Todo par de pergunta e resposta deve apresentar a REGIÃO DO
        TEXTO que se encontra a resposta. Para as respostas
        localizadas em parágrafos, forneça o indicador de texto que
        se encontra no começo do parágrafo utilizado. Para as
        respostas localizadas em tabelas, informe o NÚMERO DA TABELA
         e o NÚMERO DA LINHA utilizado. Você encontra o NÚMERO DA
        LINHA da tabela na PRIMEIRA CÉLULA DE CADA LINHA, utilize
        essa informação para compor a região do texto.
    6. Para as respostas com valores numéricos, apenas o respectivo
        valor deve ser informado na resposta, preservando símbolos
        monetários, indicadores de grandeza ou símbolos matemático.
        NÃO ADICIONE ESSES SÍMBOLOS BASEADO NO CONTEXTO, APENAS SE
        EXISTIREM PREVIAMENTE NO TEXTO.

```

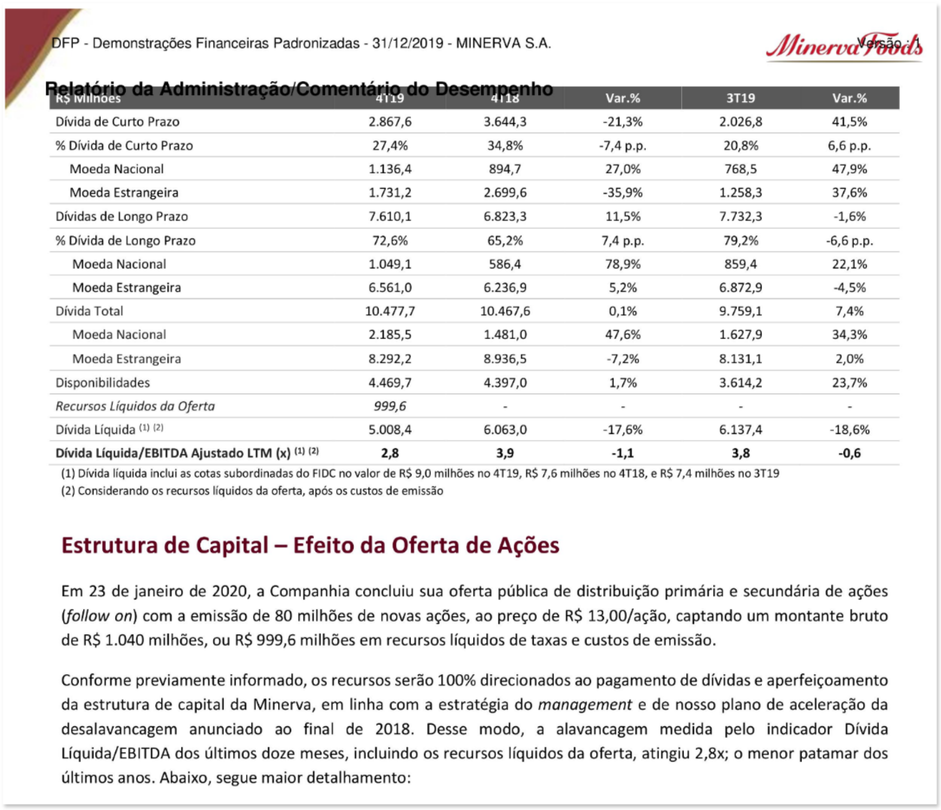
```
NÃO FAÇA PERGUNTAS que a RESPOSTA seja uma REGIÃO DO TEXTO, exemplo:
    1. Pergunta: Onde é apresentada a composição por idade de
       vencimento de contas a receber? Resposta: TABELA2
    2. Pergunta: Onde é apresentada o valor de lucros das empresas?
       Resposta: TABELA1
    3. Pergunta: Em que local é apresentado o resumo do faturamento?
        Resposta: T10

VOCÊ DEVE FORMATAR a saída da seguinte forma:
    {question_examples}
```

Code 2 describes the system prompt used for generation stage. The prompt starts with a contextualization of the domain and the task to be realized. Next, we have a description of the different coherence levels existent on Brazilian Portuguese. Then, a description of how to use adverbs and interrogative pronouns is provided to the model. The last two parts of the prompt is dedicated specifically for direct instructions from the generation process, such as the number of questions generated, focus region, output format and negative examples to avoid mistakes on generation.

In addition to the system prompt, the stage utilizes a user prompt containing the document transcription with the positional markers. Figure 8.1 illustrates the process of adding these positional markers. The first region shows a document fragment that undergoes transcription. The second region reflects the application of transcription processes outlined in the proposed approach. Finally, the last region presents the transcribed document with positional markers added at the beginning of each paragraph.

These markers vary according to the output of the DLA stage (Section 4.2.2). For regions classified as text, we added "T{paragraph number}" at the beginning of each paragraph. For regions classified as tables, we added "TABLE {table number}" before the HTML structure. Additionally, table transcriptions were modified to include row numbers, as shown in Figure 8.2.

1) Input



DFP - Demonstrações Financeiras Padronizadas - 31/12/2019 - MINERVA S.A. Versão : 1

Relatório da Administração/Comentário do Desempenho

R$ Milhões	4T19	4T18	Var.%	3T19	Var.%
Dívida de Curto Prazo	2.867,6	3.644,3	-21,3%	2.026,8	41,5%
% Dívida de Curto Prazo	27,4%	34,8%	-7,4 p.p.	20,8%	6,6 p.p.
Moeda Nacional	1.136,4	894,7	27,0%	768,5	47,9%
Moeda Estrangeira	1.731,2	2.699,6	-35,9%	1.258,3	37,6%
Dívidas de Longo Prazo	7.610,1	6.823,3	11,5%	7.732,3	-1,6%
% Dívida de Longo Prazo	72,6%	65,2%	7,4 p.p.	79,2%	-6,6 p.p.
Moeda Nacional	1.049,1	586,4	78,9%	859,4	22,1%
Moeda Estrangeira	6.561,0	6.236,9	5,2%	6.872,9	-4,5%
Dívida Total	10.477,7	10.467,6	0,1%	9.759,1	7,4%
Moeda Nacional	2.185,5	1.481,0	47,6%	1.627,9	34,3%
Moeda Estrangeira	8.292,2	8.936,5	-7,2%	8.131,1	2,0%
Disponibilidades	4.469,7	4.397,0	1,7%	3.614,2	23,7%
Recursos Líquidos da Oferta	*999,6*	-	-	-	-
Dívida Líquida [1] [2]	5.008,4	6.063,0	-17,6%	6.137,4	-18,6%
Dívida Líquida/EBITDA Ajustado LTM (x) [1] [2]	**2,8**	**3,9**	**-1,1**	**3,8**	**-0,6**

(1) Dívida líquida inclui as cotas subordinadas do FIDC no valor de R$ 9,0 milhões no 4T19, R$ 7,6 milhões no 4T18, e R$ 7,4 milhões no 3T19
(2) Considerando os recursos líquidos da oferta, após os custos de emissão

Estrutura de Capital – Efeito da Oferta de Ações

Em 23 de janeiro de 2020, a Companhia concluiu sua oferta pública de distribuição primária e secundária de ações (*follow on*) com a emissão de 80 milhões de novas ações, ao preço de R$ 13,00/ação, captando um montante bruto de R$ 1.040 milhões, ou R$ 999,6 milhões em recursos líquidos de taxas e custos de emissão.

Conforme previamente informado, os recursos serão 100% direcionados ao pagamento de dívidas e aperfeiçoamento da estrutura de capital da Minerva, em linha com a estratégia do *management* e de nosso plano de aceleração da desalavancagem anunciado ao final de 2018. Desse modo, a alavancagem medida pelo indicador Dívida Líquida/EBITDA dos últimos doze meses, incluindo os recursos líquidos da oferta, atingiu 2,8x; o menor patamar dos últimos anos. Abaixo, segue maior detalhamento:

2) Text Transcription + Positional Marker

3) Output

T1: DFP - Demonstrações Financeiras Padronizadas - 31/12/2019 - MINERVA S.A.
T2: Minerva Howit's
T3: Relatório da Administração/Comentário do Desempenho

TABELA1:
```html
<table><thead><tr><th id=line-number> 1 </th><th> Relatório da Administração/Comentário do Desempenho...
</table>
```

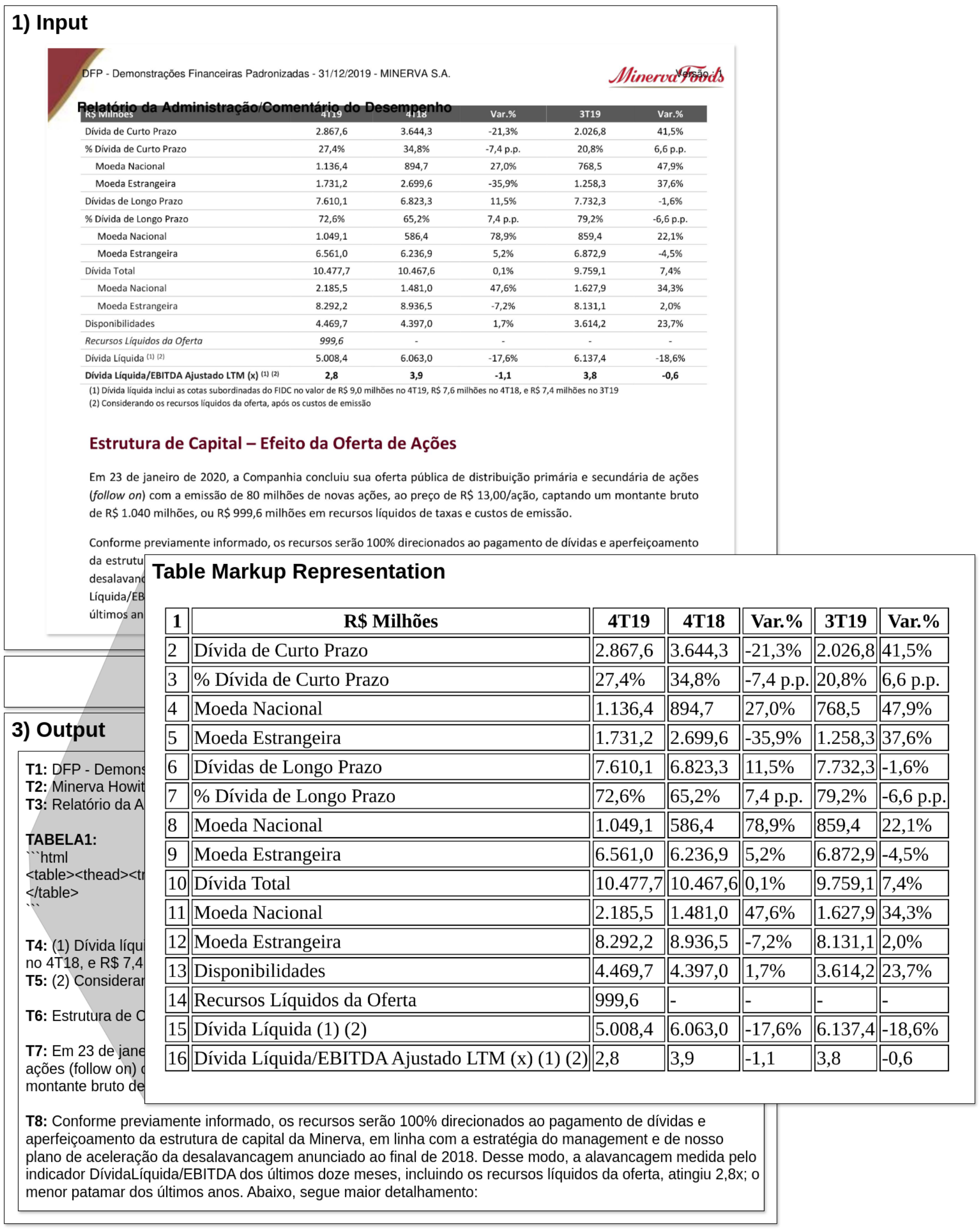
T4: (1) Dívida líquida inclui as cotas subordinadas do FIDC no valor de R$ 9,0 milhões no 4T19, R$ 7,6 milhões no 4T18, e R$ 7,4 milhões no 3T19
T5: (2) Considerando os recursos líquidos da oferta, após os custos de emissão

T6: Estrutura de Capital - Efeito da Oferta de Ações

T7: Em 23 de janeiro de 2020, a Companhia concluiu sua oferta pública de distribuição primária e secundária de ações (follow on) com a emissão de 80 milhões de novas ações, ao preço de R$ 13,00/ação, captando um montante bruto de R$ 1.040 milhões, ou R$ 999,6 milhões em recursos líquidos de taxas e custos de emissão.

T8: Conforme previamente informado, os recursos serão 100% direcionados ao pagamento de dívidas e aperfeiçoamento da estrutura de capital da Minerva, em linha com a estratégia do management e de nosso plano de aceleração da desalavancagem anunciado ao final de 2018. Desse modo, a alavancagem medida pelo indicador DívidaLíquida/EBITDA dos últimos doze meses, incluindo os recursos líquidos da oferta, atingiu 2,8x; o menor patamar dos últimos anos. Abaixo, segue maior detalhamento:

Figure 8.1: Process of adding the positional markers to the transcription of the text to generate the question-answer pairs



1) Input

DFP - Demonstrações Financeiras Padronizadas - 31/12/2019 - MINERVA S.A. Versão : 1

Relatório da Administração/Comentário do Desempenho

R$ Milhões	4T19	4T18	Var.%	3T19	Var.%
Dívida de Curto Prazo	2.867,6	3.644,3	-21,3%	2.026,8	41,5%
% Dívida de Curto Prazo	27,4%	34,8%	-7,4 p.p.	20,8%	6,6 p.p.
Moeda Nacional	1.136,4	894,7	27,0%	768,5	47,9%
Moeda Estrangeira	1.731,2	2.699,6	-35,9%	1.258,3	37,6%
Dívidas de Longo Prazo	7.610,1	6.823,3	11,5%	7.732,3	-1,6%
% Dívida de Longo Prazo	72,6%	65,2%	7,4 p.p.	79,2%	-6,6 p.p.
Moeda Nacional	1.049,1	586,4	78,9%	859,4	22,1%
Moeda Estrangeira	6.561,0	6.236,9	5,2%	6.872,9	-4,5%
Dívida Total	10.477,7	10.467,6	0,1%	9.759,1	7,4%
Moeda Nacional	2.185,5	1.481,0	47,6%	1.627,9	34,3%
Moeda Estrangeira	8.292,2	8.936,5	-7,2%	8.131,1	2,0%
Disponibilidades	4.469,7	4.397,0	1,7%	3.614,2	23,7%
Recursos Líquidos da Oferta	*999,6*	-	-	-	-
Dívida Líquida (1) (2)	5.008,4	6.063,0	-17,6%	6.137,4	-18,6%
Dívida Líquida/EBITDA Ajustado LTM (x) (1) (2)	**2,8**	**3,9**	**-1,1**	**3,8**	**-0,6**

(1) Dívida líquida inclui as cotas subordinadas do FIDC no valor de R$ 9,0 milhões no 4T19, R$ 7,6 milhões no 4T18, e R$ 7,4 milhões no 3T19
(2) Considerando os recursos líquidos da oferta, após os custos de emissão

Estrutura de Capital – Efeito da Oferta de Ações

Em 23 de janeiro de 2020, a Companhia concluiu sua oferta pública de distribuição primária e secundária de ações (*follow on*) com a emissão de 80 milhões de novas ações, ao preço de R$ 13,00/ação, captando um montante bruto de R$ 1.040 milhões, ou R$ 999,6 milhões em recursos líquidos de taxas e custos de emissão.

Conforme previamente informado, os recursos serão 100% direcionados ao pagamento de dívidas e aperfeiçoamento
da estrutu
desalavanc
Líquida/EB
últimos an

Table Markup Representation

1	R$ Milhões	4T19	4T18	Var.%	3T19	Var.%
2	Dívida de Curto Prazo	2.867,6	3.644,3	-21,3%	2.026,8	41,5%
3	% Dívida de Curto Prazo	27,4%	34,8%	-7,4 p.p.	20,8%	6,6 p.p.
4	Moeda Nacional	1.136,4	894,7	27,0%	768,5	47,9%
5	Moeda Estrangeira	1.731,2	2.699,6	-35,9%	1.258,3	37,6%
6	Dívidas de Longo Prazo	7.610,1	6.823,3	11,5%	7.732,3	-1,6%
7	% Dívida de Longo Prazo	72,6%	65,2%	7,4 p.p.	79,2%	-6,6 p.p.
8	Moeda Nacional	1.049,1	586,4	78,9%	859,4	22,1%
9	Moeda Estrangeira	6.561,0	6.236,9	5,2%	6.872,9	-4,5%
10	Dívida Total	10.477,7	10.467,6	0,1%	9.759,1	7,4%
11	Moeda Nacional	2.185,5	1.481,0	47,6%	1.627,9	34,3%
12	Moeda Estrangeira	8.292,2	8.936,5	-7,2%	8.131,1	2,0%
13	Disponibilidades	4.469,7	4.397,0	1,7%	3.614,2	23,7%
14	Recursos Líquidos da Oferta	999,6	-	-	-	-
15	Dívida Líquida (1) (2)	5.008,4	6.063,0	-17,6%	6.137,4	-18,6%
16	Dívida Líquida/EBITDA Ajustado LTM (x) (1) (2)	2,8	3,9	-1,1	3,8	-0,6

3) Output

T1: DFP - Demons
T2: Minerva Howit
T3: Relatório da A

TABELA1:
```html
<table><thead><tr
</table>
```

T4: (1) Dívida líqui
no 4T18, e R$ 7,4
T5: (2) Considerar

T6: Estrutura de C

T7: Em 23 de jane
ações (follow on) c
montante bruto de

T8: Conforme previamente informado, os recursos serão 100% direcionados ao pagamento de dívidas e aperfeiçoamento da estrutura de capital da Minerva, em linha com a estratégia do management e de nosso plano de aceleração da desalavancagem anunciado ao final de 2018. Desse modo, a alavancagem medida pelo indicador DívidaLíquida/EBITDA dos últimos doze meses, incluindo os recursos líquidos da oferta, atingiu 2,8x; o menor patamar dos últimos anos. Abaixo, segue maior detalhamento:

Figure 8.2: Process of adding the line numbers to table structure in question and answer generation step

8.2 Question-Answer Judgment Prompt

In this section of the appendix, the prompt constructed for QA judgment is detailed. As this is a three-stage method, we will have three text blocks, representing each prompt used for the process.

Code 3: System prompt of QA judgment stage

```
Você é um sistema de avaliação de perguntas e respostas muito
    critérioso. Você avalia detalhadamente as perguntas e respostas,
     garantindo que ambos respeitam a gramática normativa da língua
    portuguesa.
Você só confirma que uma pergunta é coerente quando tem absoluta
    certeza da sua resposta. Você só confirma que a resposta está
    completamente correta quando tem absoluta certeza da sua
    resposta, sempre
levando em consideração o texto de referência informado.

Sua avaliação deve seguir os critérios abaixo:
    a. PASSOS PARA AVALIAR SE A PERGUNTA É COERENTE:
        1. A pergunta permite a compreensão textual do leitor,
            apresentando uma combinação de palavras com conteúdo
            claro, lógico e direto;
        2. A pergunta segue as regras da gramática normativa da
            lingua portuguesa;
        3. A pergunta não apresenta ambiguidade possibilitando a
            indentificação clara da sua resposta;
        4. A pergunta está diretamente relacionada ao texto de
            referência;
        5. A pergunta pode ser respondida diretamente pelo texto de
            referência;
        6. Quando a sua formulação possibilita apenas uma única
            resposta, ESPECIALMENTE EM TABELAS.

    b. PASSOS PARA AVALIAR SE UMA RESPOSTA ESTÁ COMPLETAMENTE
        CORRETA:
        1. Não considerar as ausências ou presenças das unidades de
            medidas
        2. Não considerar as ausências ou presenças das unidades
            monetárias
        3. Não considerar as ausências ou presenças dos simbolos
            matemáticos de uma resposta
        4. Não considerar as ausências ou presenças de parenteses ou
             colchetes
        5. Perguntas que apresentam o advérbio interrogativo "Quando
            ", devem apresentar uma resposta indicando valor ou
            termo temporal;
        6. Perguntas que apresentam o advérbio interrogativo "Quem",
             devem apresentar uma resposta indicando uma pessoa ou
            coisas personificadas;
        7. Perguntas que apresentam o advérbio interrogativo "Onde",
             devem apresentar uma resposta indicando uma cidade,
            estado, país ou endereço;
        8. A resposta deve ser coerente e baseada no texto de
            referência;
        9. Se a PERGUNTA NÃO É COERENTE a RESPOSTA NÃO É CORRETA.

Esse é o seu texto de referência:
{texto_referencia}
```

```

Você só deve RESPONDER SIM ou NÃO.
```

The first stage is the model preparation, which we gived it all the instructions it needs to carry out the judgement correctly, as presenting in Listing 3. In this prompt, we define the rules for evaluating questions and answers, as well as defining certain behaviors that the model must follow, such as only considering an answer correct if the question is coherent.

Code 4: User prompts of QA judgment stage

```
question_prompt: A pergunta: "{pergunta}"; é semânticamente coerente
    em português?

answer_prompt: Com base no texto de referência, a resposta: "{
    resposta_gen_model}"; é semânticamente coerente e responde a
    pergunta: "{pergunta}"?
```

Subsequently, there are two other prompts which give the question to be evaluated and its answer, respectively. They are separated into two because the model performs one evaluation at a time. Listing 4 describes these two prompts, where the first is completed with the question to be evaluated and the second requires both the question and the answer.

8.3 Positional Marker Returning Errors

This section outlines the main errors in positional markers for each model used in the Question and Answer Generation Stage. Table 8.1 summarizes these errors. For the Mixtral-8x22B model, the primary issues stem from not returning the full positional marker and incorrectly combining output formats for both text and table regions. A similar problem with mixing formats occurs with LLama3 70B and GPT-4o mini.

The Gemini 1.5 Flash model's key issue is the inclusion of column information, which was not requested in the prompt instructions. Additionally, it occasionally returns empty strings. Claude 3 Haiku, however, produced the highest number of errors, including incomplete information, unnecessary column details, and inconsistent formatting between text and table regions.

8.4 Inter-annotator Agreement

This Appendix Section outlines percent inter-annotator agreement between each annotator. All the three evaluated conditions are showed in this section, the Question Coherence Figure 8.3, Answer Accuracy Figure 8.4 and Utilization 8.5.

Model	Errors
Mixtral-8x22b	TABELA T (TABELA , LINHA) EMPTY STRING
GPT-4o mini	T, LINHA
Gemini 1.5 Flash	TABELA - LINE-NUMBER TABELA - LINHA - COLUNA EMPTY STRING
LLama3 70B	T, LINHA T, TABELA , LINHA
Claude 3 Haiku	TABELA TABELA , LINHA , COLUNA PARÁGRAFO TABELA , ÚLTIMA LINHA

Table 8.1: Main errors made by each model in positional marker format on generation stage

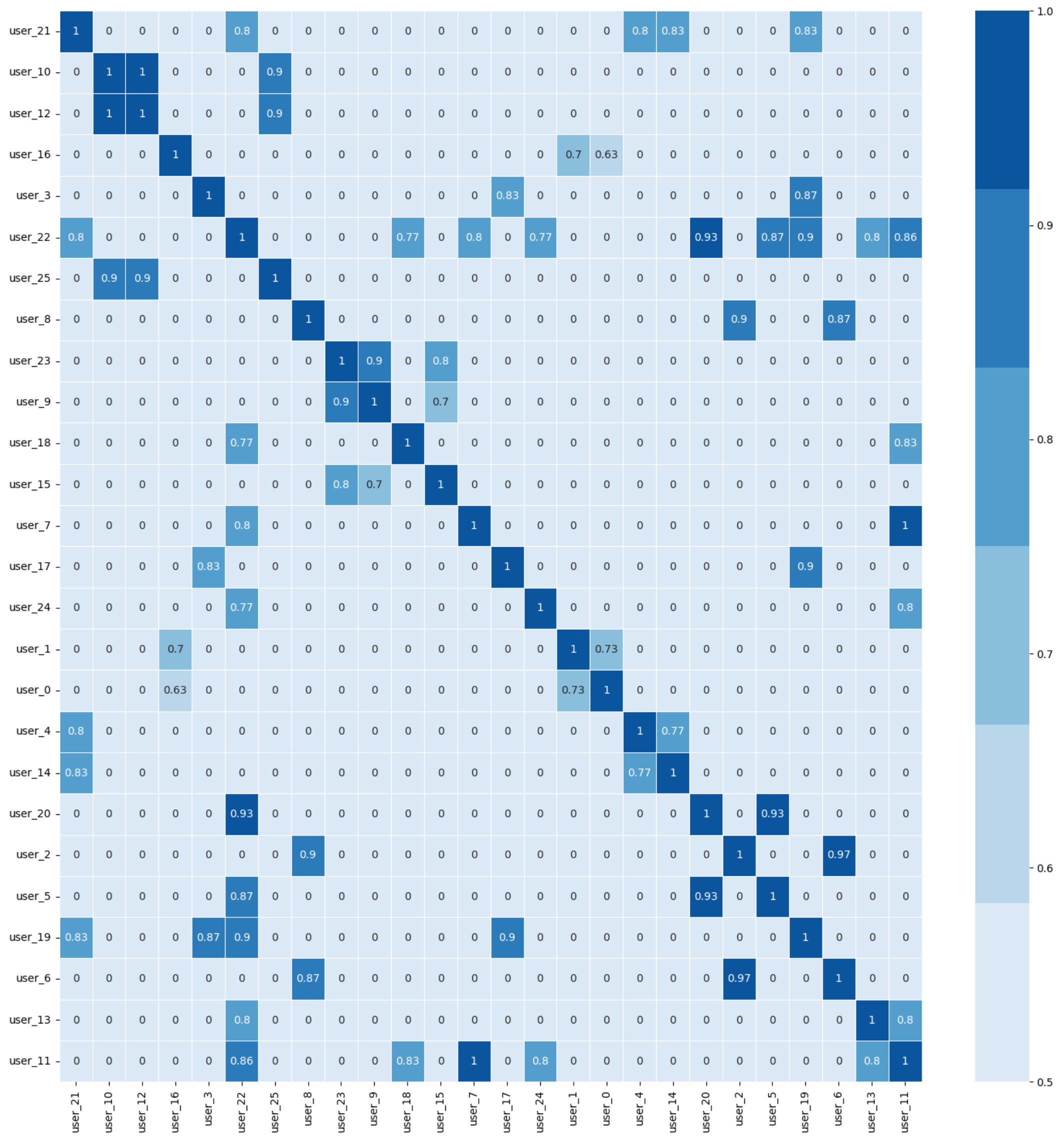
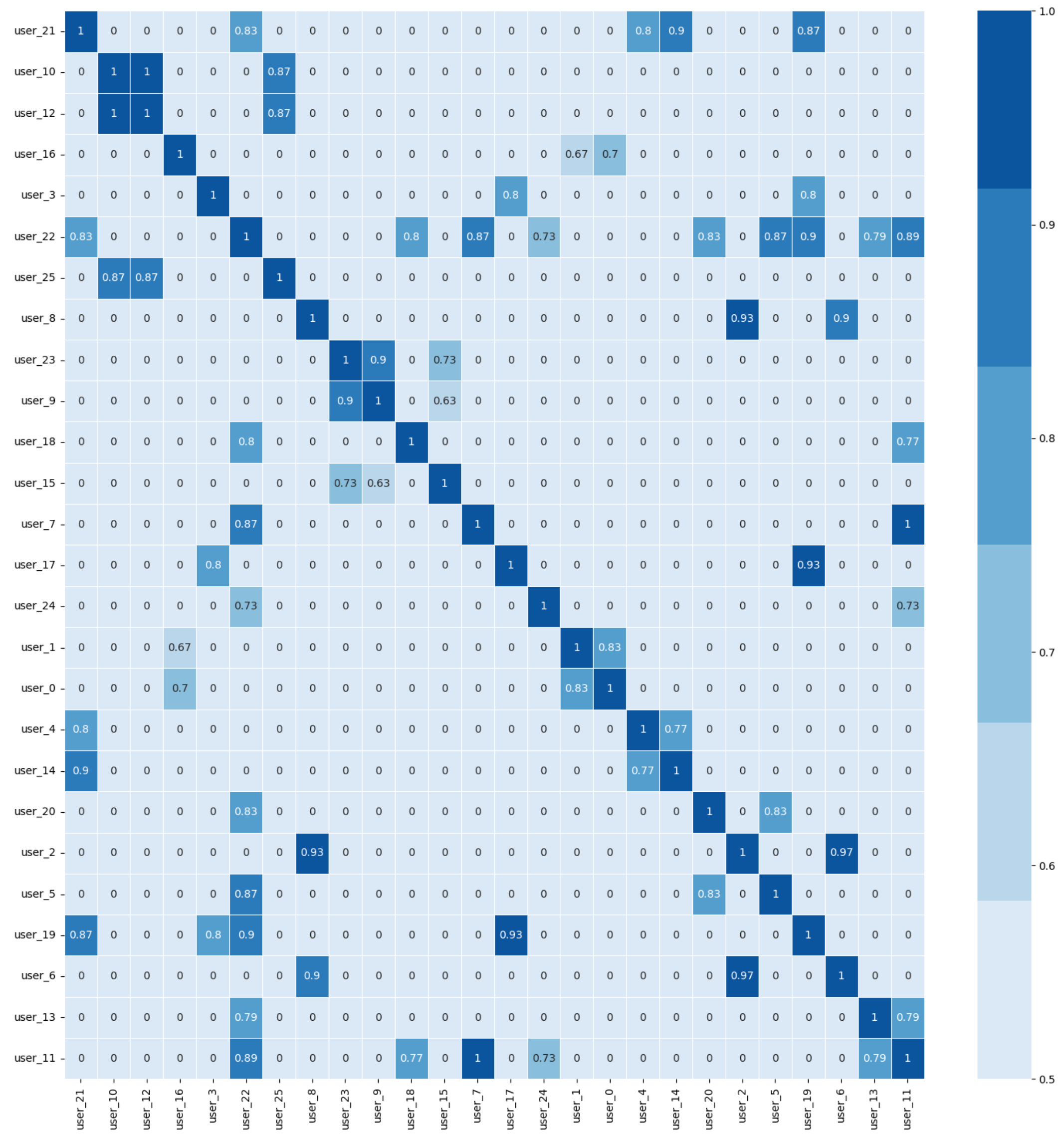


Figure 8.3: Inter-annotator agreement for question coherence of QA pairs in golden dataset



	user_21	user_10	user_12	user_16	user_3	user_22	user_25	user_8	user_23	user_9	user_18	user_15	user_7	user_17	user_24	user_1	user_0	user_4	user_14	user_20	user_2	user_5	user_19	user_6	user_13	user_11
user_21	1	0	0	0	0	0.83	0	0	0	0	0	0	0	0	0	0	0	0.8	0.9	0	0	0	0.87	0	0	0
user_10	0	1	1	0	0	0	0.87	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_12	0	1	1	0	0	0	0.87	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_16	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0.67	0.7	0	0	0	0	0	0	0	0	0
user_3	0	0	0	0	1	0	0	0	0	0	0	0	0	0.8	0	0	0	0	0	0	0	0	0.8	0	0	0
user_22	0.83	0	0	0	0	1	0	0	0	0	0.8	0	0.87	0	0.73	0	0	0	0	0.83	0	0.87	0.9	0	0.79	0.89
user_25	0	0.87	0.87	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0.93	0	0	0.9	0	0
user_23	0	0	0	0	0	0	0	0	1	0.9	0	0.73	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_9	0	0	0	0	0	0	0	0	0.9	1	0	0.63	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_18	0	0	0	0	0	0.8	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.77
user_15	0	0	0	0	0	0	0	0	0.73	0.63	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
user_7	0	0	0	0	0	0.87	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
user_17	0	0	0	0	0.8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0.93	0	0	0
user_24	0	0	0	0	0	0.73	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0.73
user_1	0	0	0	0.67	0	0	0	0	0	0	0	0	0	0	0	1	0.83	0	0	0	0	0	0	0	0	0
user_0	0	0	0	0.7	0	0	0	0	0	0	0	0	0	0	0	0.83	1	0	0	0	0	0	0	0	0	0
user_4	0.8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.77	0	0	0	0	0	0	0
user_14	0.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.77	1	0	0	0	0	0	0	0
user_20	0	0	0	0	0	0.83	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.83	0	0	0	0
user_2	0	0	0	0	0	0	0	0.93	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0.97	0	0
user_5	0	0	0	0	0	0.87	0	0	0	0	0	0	0	0	0	0	0	0	0	0.83	0	1	0	0	0	0
user_19	0.87	0	0	0	0.8	0.9	0	0	0	0	0	0	0	0.93	0	0	0	0	0	0	0	0	1	0	0	0
user_6	0	0	0	0	0	0	0	0.9	0	0	0	0	0	0	0	0	0	0	0	0	0.97	0	0	1	0	0
user_13	0	0	0	0	0	0.79	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.79
user_11	0	0	0	0	0	0.89	0	0	0	0	0.77	0	1	0	0.73	0	0	0	0	0	0	0	0	0	0.79	1

Figure 8.4: Inter-annotator agreement for answer accuracy of QA pairs in golden dataset

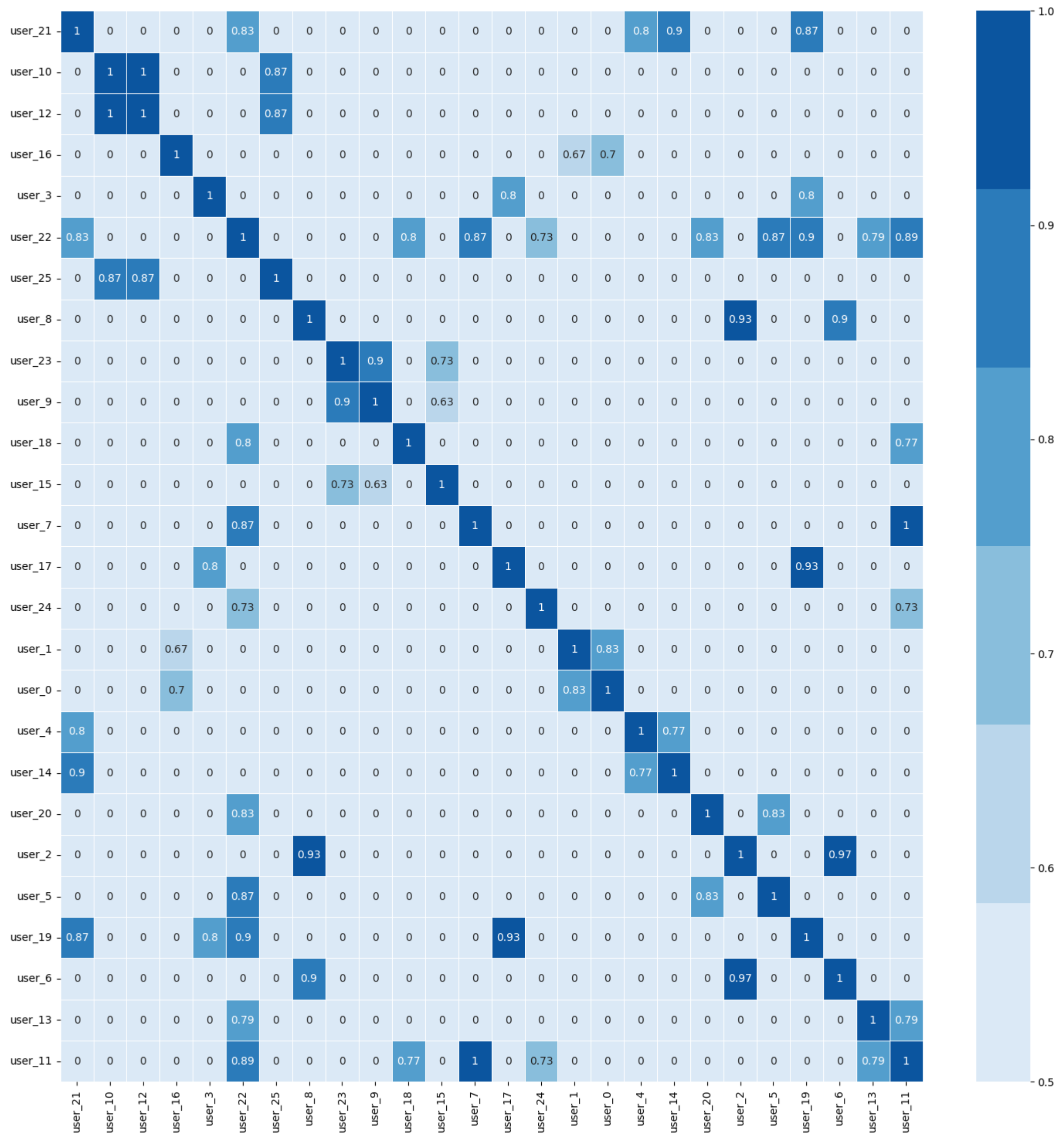


Figure 8.5: Inter-annotator agreement for utilization of QA pairs in golden dataset