**Atividade 2: Aplicação de Modelos de QA em Documentos com Hugging Face**

Disciplina: Processamento de Linguagem Natural  
Universidade Federal de Sergipe

**Equipe:  
José Batista  
Carlos Melo  
Roberdan Tamyr  
Arthur Matheus**

**Sumário**

[1. Introdução 3](#_heading=h.26jrv5is3q8f)

[2. Descrição da Atividade 3](#_heading=h.vjccr18hjlmh)

[3. Metodologia 3](#_heading=h.jhivr5es19in)

[3.1. Configuração do Ambiente 3](#_heading=h.462bbvubb5yc)

[Instalação de arquivos necessários 4](#_heading=h.9e3xnx3zlvj1)

[3.2. Extração e Processamento do Texto 4](#_heading=h.yg2nks3baxpp)

[Extração de texto do .docx 4](#_heading=h.5v77nst0uzow)

[Extração de texto do .pdf 4](#_heading=h.ep3rnc7khe2f)

[3.3. Experimentos Realizados 4](#_heading=h.z3hprbtrxz3c)

[3.4. Critérios de Avaliação 5](#_heading=h.7ne50d0mlix)

[4. Resultados e Análise 5](#_heading=h.6r5ezjpdo9yb)

[4.1. Experimento 1 5](#_heading=h.lsswqhmegryz)

[4.2. Experimento 2 7](#_heading=h.9k9cwo27uuc0)

[4.3. Experimento 3 8](#_heading=h.h8bbml4m4ewz)

[5. Conclusão 11](#_heading=h.vs39sc7x16cs)

[6. Referências 12](#_heading=h.bvb9sig3xhiq)

[7. Repositório 12](#_heading=h.rvj3f76abqn9)

[8. Notebook 12](#_heading=h.gnk0hqapjomo)

[9. Vídeo 12](#_heading=h.w5zx6ozcaf1f)

[10. Participação 12](#_heading=h.4nwwt5yrqzcm)

# 1. Introdução

Este documento detalha a implementação e avaliação de um sistema de Perguntas e Respostas (Question Answering - QA) que utiliza modelos de Linguagem Natural (LLMs) disponíveis na plataforma Hugging Face. A atividade tem como objetivo extrair informações de documentos nos formatos DOCX e PDF para responder a perguntas específicas, comparando a eficácia de diferentes modelos de PLN.  
 Este documento apresenta três experimentos distintos no contexto de Perguntas e Respostas (QA) aplicados a documentos em formato DOCX e PDF. O foco é explorar o impacto de diferentes abordagens de recuperação de contexto e uso de Modelos de Linguagem de Grande Porte (LLMs) sobre o desempenho final.

# 2. Descrição da Atividade

A atividade consiste na criação de um script em Python, no formato de um Notebook do Google Colab, para implementar um sistema de Perguntas e Respostas. A base de conhecimento para o sistema são dois documentos fornecidos: “DICIONARIO\_DE\_DADOS.docx” e “doencas\_respiratorias\_cronicas.pdf”.  
 O processo envolve a seleção de, no mínimo, três modelos de QA gratuitos da plataforma Hugging Face. Para cada um dos dois documentos, foram elaboradas três perguntas distintas, totalizando seis perguntas para o teste de cada modelo.  
 O principal objetivo é comparar o desempenho dos modelos, avaliando a precisão e o alinhamento das respostas em relação ao conteúdo dos textos de referência. A avaliação de desempenho é apresentada através de uma tabela comparativa detalhada e um gráfico visual, destacando a efetividade de cada modelo e os critérios utilizados para a comparação.

A atividade foi organizada em três experimentos:

* **Experimento 1(Baseline com QA)**: Utilização de três modelos de Question Answering do Hugging Face, aplicados diretamente sobre o texto integral dos documentos.
* **Experimento 2 (RAG com Embeddings):** Implementação de RAG com embeddings (paraphrase-multilingual-mpnet-base-v2) e FAISS.
* **Experimento 3 (RAG com Cross-Encoder + LLMs):** Extensão do RAG com reordenação via cross-encoder (ms-marco-MiniLM-L-6-v2).

# 3. Metodologia

A metodologia foi estruturada em 4 etapas principais: configuração do ambiente, extração e processamento do texto dos documentos, realização dos experimentos e, por fim, a avaliação dos resultados.

## 3.1. Configuração do Ambiente

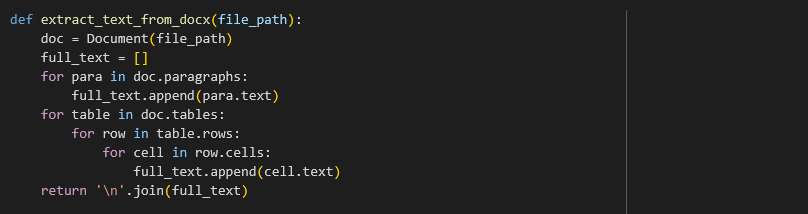
O ambiente de desenvolvimento foi configurado no Google Colab, garantindo a reprodutibilidade do projeto. As bibliotecas essenciais para a manipulação de arquivos, processamento de linguagem natural e visualização de dados foram instaladas, assim como bibliotecas Hugging Face Transformers, FAISS e Sentence-Transformers.

### Instalação de arquivos necessários

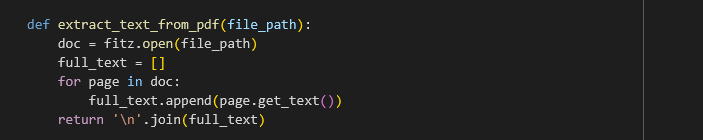
## **3.2. Extração** e Processamento do Texto

Para que os modelos pudessem processar as informações, o texto foi extraído dos arquivos DOCX e PDF. Foram criadas funções específicas para cada tipo de arquivo, utilizando as bibliotecas python-docx e PyMuPDF.

### Extração de texto do .docx



### Extração de texto do .pdf



Como os modelos de QA possuem uma limitação no tamanho do contexto que podem processar de uma só vez, os textos extraídos foram divididos em blocos menores (chunks) para garantir que nenhuma informação fosse perdida durante a análise.

## 3.3. Experimentos Realizados

* **Experimento 1 (Baseline com QA):** Utilização de três modelos de *Question Answering* do Hugging Face, aplicados diretamente sobre o texto integral dos documentos. Este experimento corresponde à **primeira versão da tarefa**, realizada antes das sugestões fornecidas pelo professor em aula. Uma descrição mais detalhada dessa versão inicial pode ser consultada no seguinte link: [Arthur\_Lima\_atividade2\_v1.docx](https://docs.google.com/document/d/1q2rOGUq5_8dpt74ry1nd_KeMMwLv05G6/edit?usp=sharing&ouid=109641451200619281802&rtpof=true&sd=true).
* **Experimento 2 (RAG com Embeddings):** Após as recomendações do professor, implementamos a técnica de Recuperação e Geração (RAG), utilizando o modelo de embeddings *paraphrase-multilingual-mpnet-base-v2* para representação vetorial dos trechos do texto. Foi criado um banco de dados vetorial com FAISS e, para cada pergunta, os cinco trechos mais semelhantes foram recuperados e utilizados como contexto para os modelos de linguagem.
* **Experimento 3 (RAG com Cross-Encoder + LLMs):** Como extensão da estratégia anterior, recuperamos os 10 trechos mais semelhantes e aplicamos um *cross-encoder* (*ms-marco-MiniLM-L-6-v2*) para reordená-los por relevância. Os três melhores foram então utilizados como contexto para os LLMs (*google/gemma-2b-it*, *meta-llama/Llama-3.2-1B-Instruct* e *meta-llama/Llama-3.2-3B-Instruct*).

## 3.4. Critérios de Avaliação

A avaliação da efetividade dos modelos foi baseada em três critérios:

1. Avaliação Manual: As respostas geradas foram classificadas manualmente em três categorias para uma análise qualitativa rápida:  
   ✔️ (Correta): A resposta do modelo é idêntica ou semanticamente equivalente à resposta esperada.  
   ➖ (Parcialmente Correta): A resposta contém parte da informação correta, mas está incompleta ou imprecisa.  
   ❌ (Incorreta): A resposta está errada ou não tem relação com a pergunta.
2. Similaridade Semântica (Cosseno): Para uma avaliação quantitativa, a similaridade de cosseno entre os embeddings da resposta do modelo e da resposta esperada foi calculada. Utilizou-se o modelo ***sentence-transformers/all-MiniLM-L6-v2*** para gerar os embeddings. Essa métrica varia de -1 a 1, onde valores mais próximos de 1 indicam maior similaridade semântica.
3. Confiança do Modelo: A pontuação de confiança (score) retornada pelo próprio modelo de QA foi registrada. Este valor indica o quão confiante o modelo está em sua própria resposta.

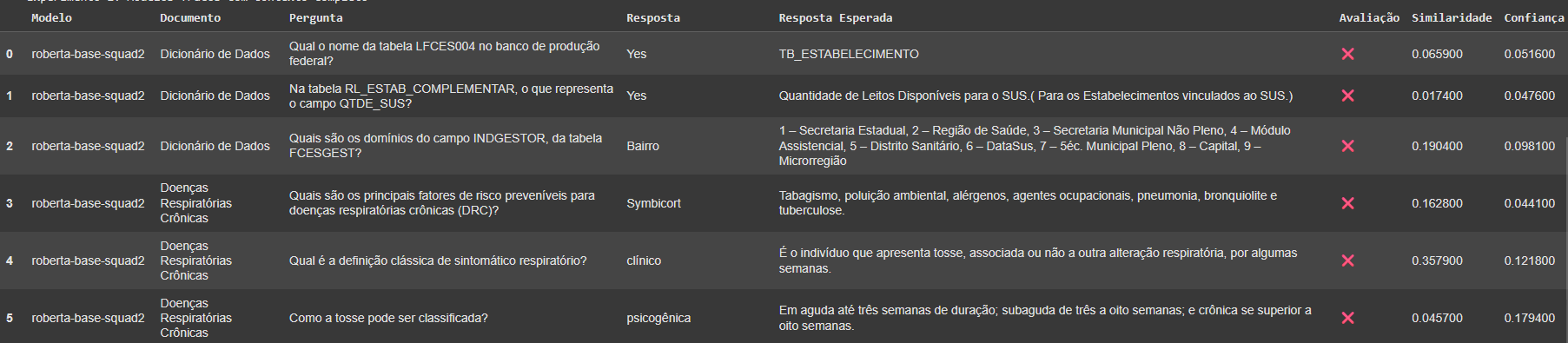
# 4. Resultados e Análise

Os resultados da avaliação foram compilados em tabelas detalhadas para cada modelo e em um gráfico comparativo para facilitar a análise.

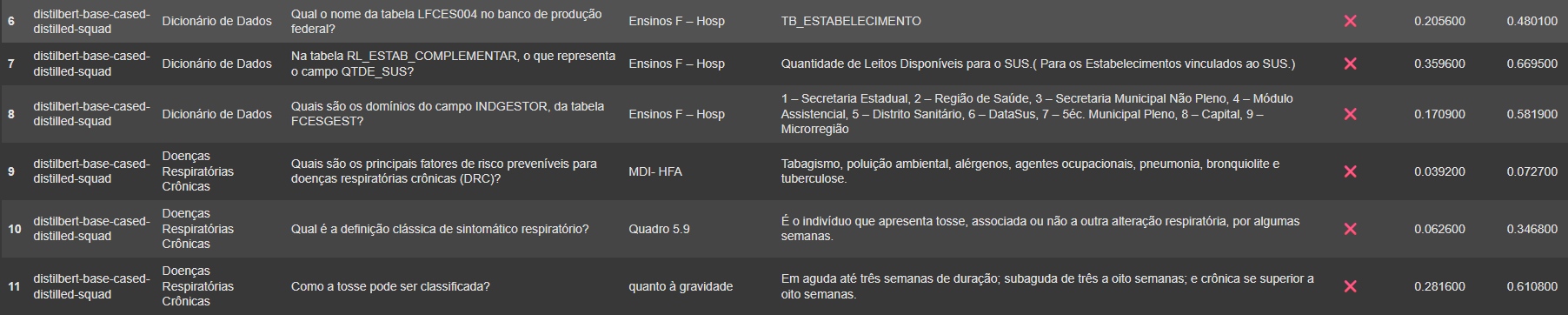
## 4.1. Experimento 1

As tabelas a seguir exibem os resultados detalhados para cada modelo, incluindo a pergunta, a resposta gerada, a resposta esperada e as métricas de avaliação.

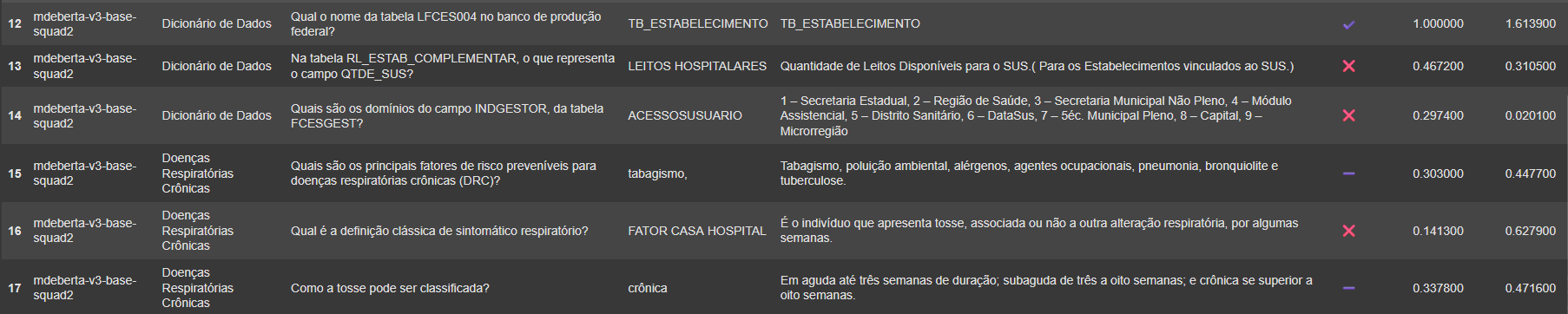
Resultados para: ***roberta-base-squad2***



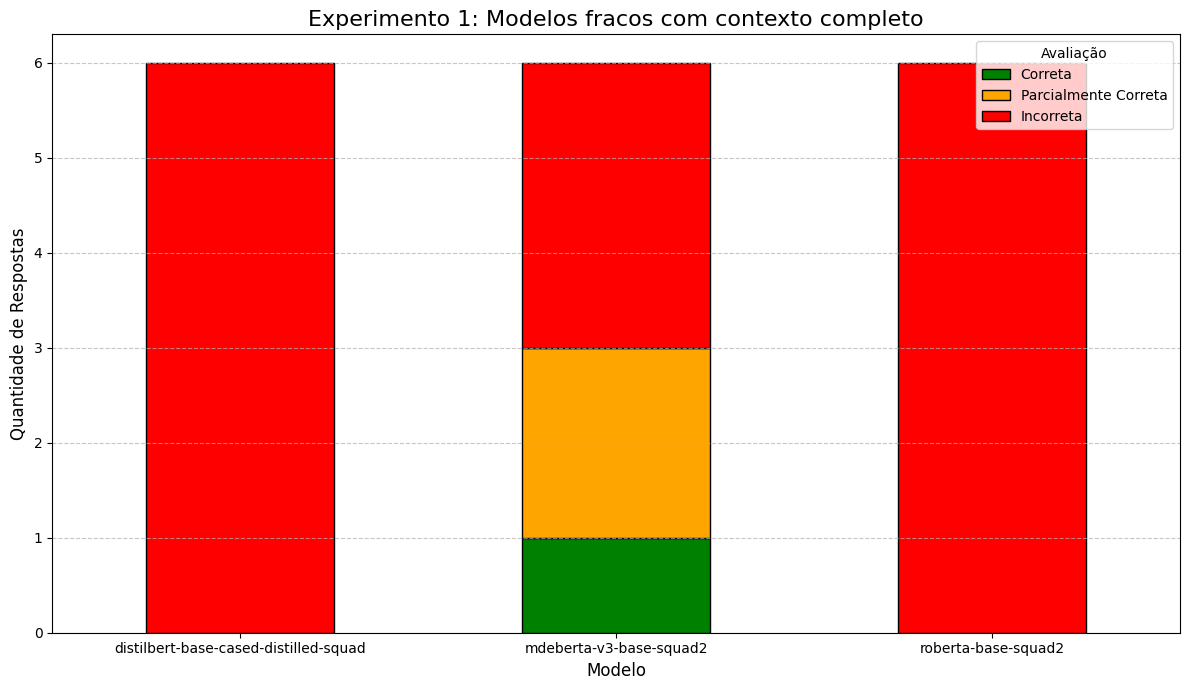
Resultados para: ***distilbert-base-cased-distilled-squad***



Resultados para: ***mdeberta-v3-base-squad2***



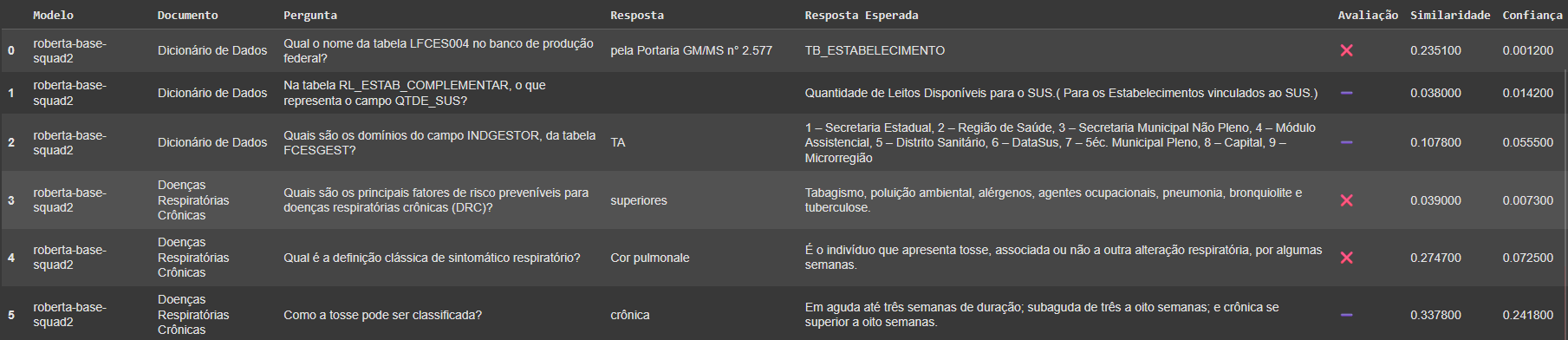
O gráfico de barras empilhadas abaixo resume a avaliação manual, mostrando a quantidade de respostas corretas, parcialmente corretas e incorretas para cada modelo.



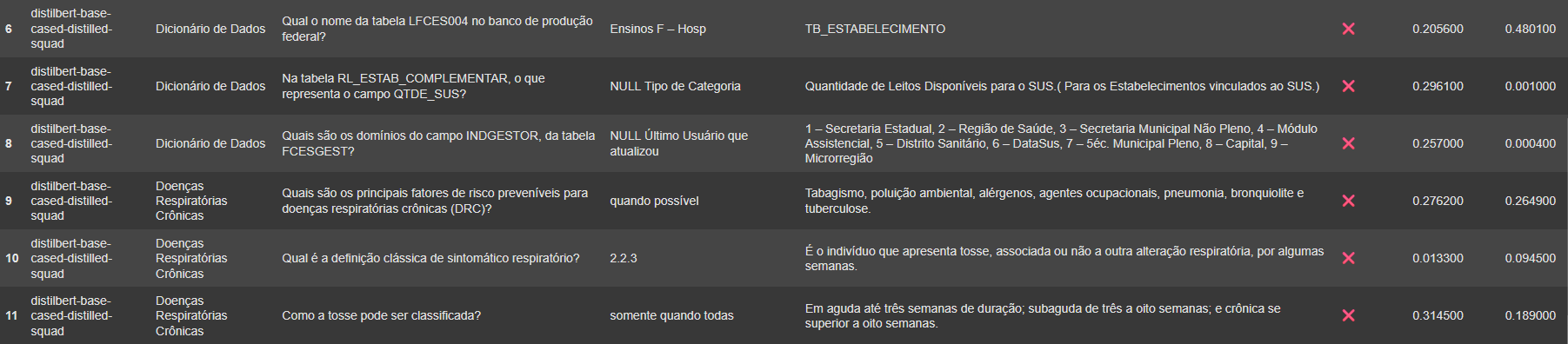
## 4.2. Experimento 2

As tabelas a seguir exibem os resultados detalhados para cada modelo, incluindo a pergunta, a resposta gerada, a resposta esperada e as métricas de avaliação.

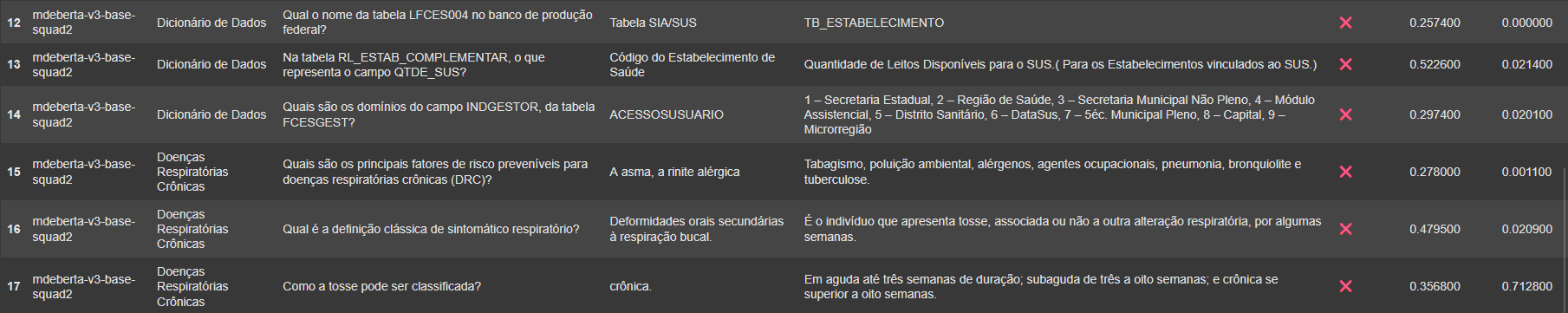
Resultados para: ***roberta-base-squad2***



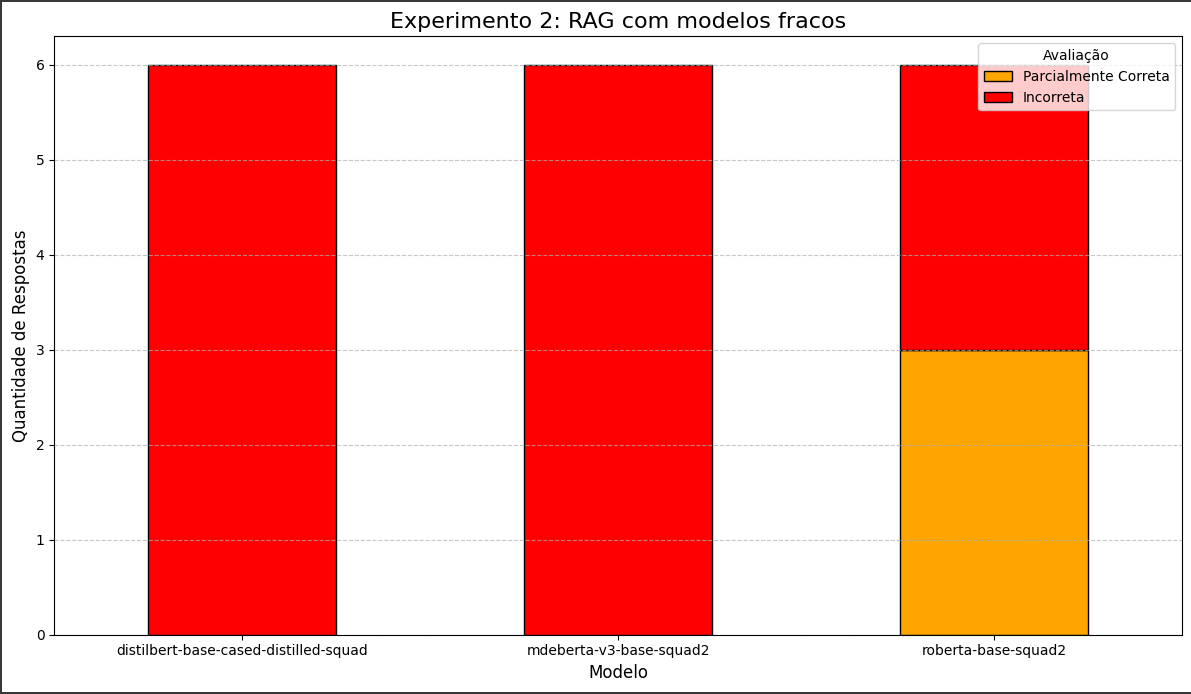
Resultados para: ***distilbert-base-cased-distilled-squad***



Resultados para: ***mdeberta-v3-base-squad2***



O gráfico de barras empilhadas abaixo resume a avaliação manual, mostrando a quantidade de respostas corretas, parcialmente corretas e incorretas para cada modelo.

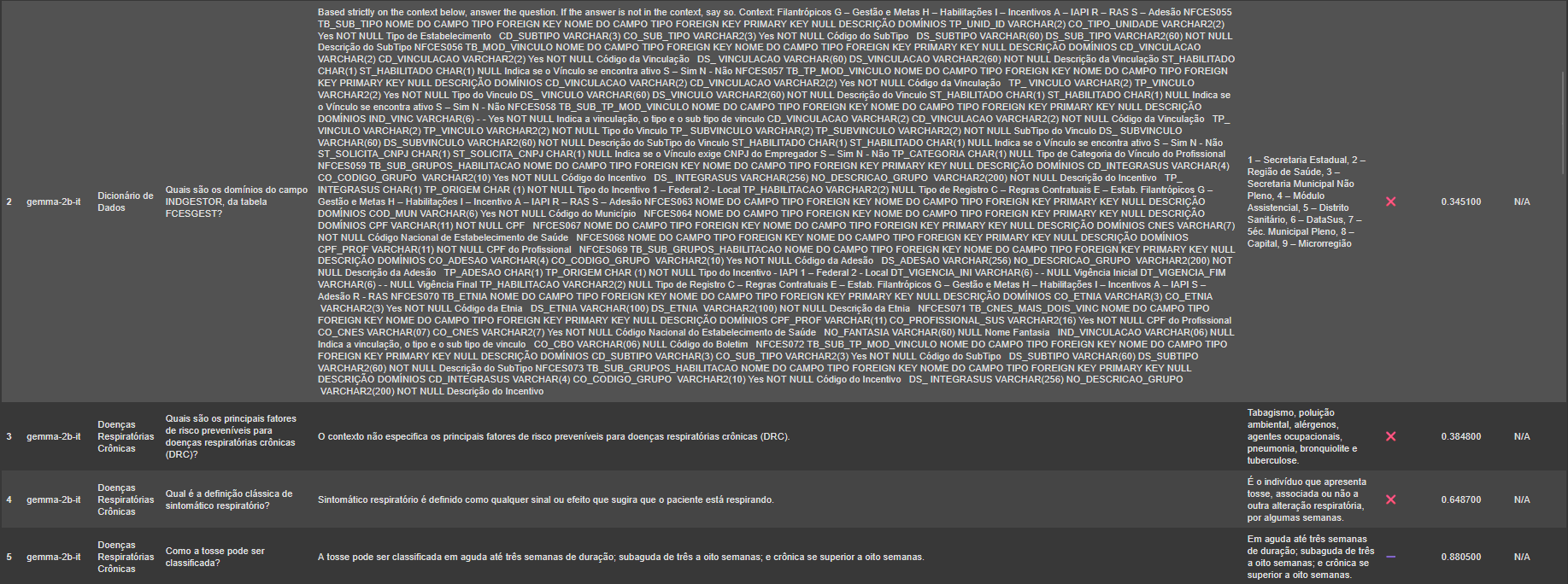


## 4.3. Experimento 3

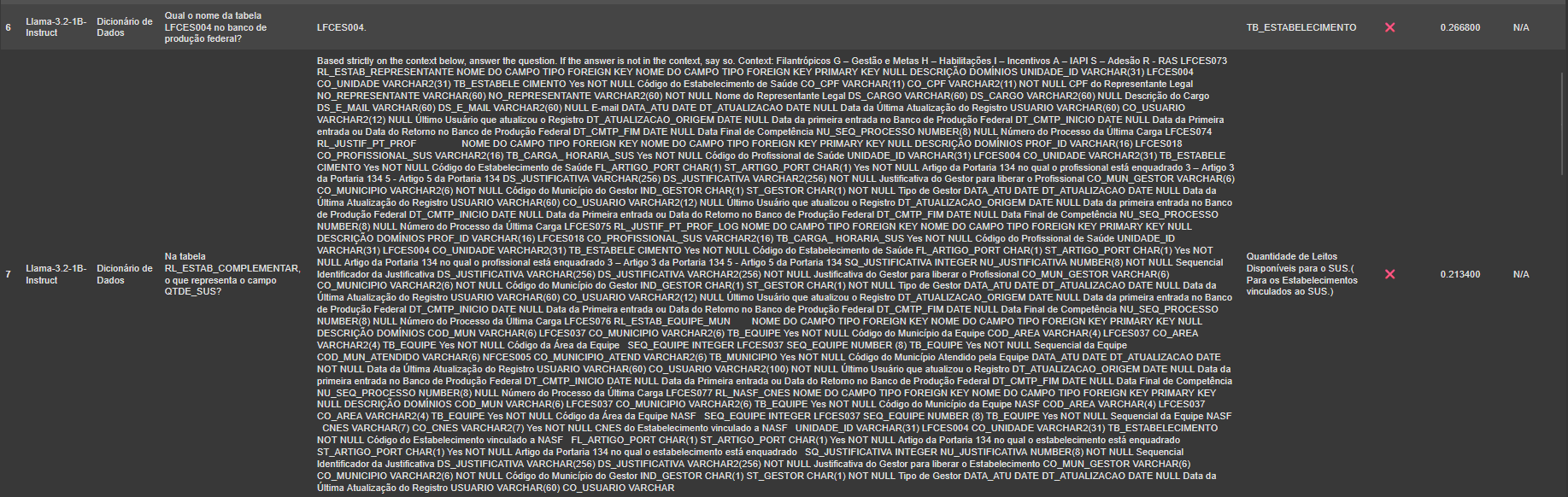
As tabelas a seguir exibem os resultados detalhados para cada modelo, incluindo a pergunta, a resposta gerada, a resposta esperada e as métricas de avaliação.

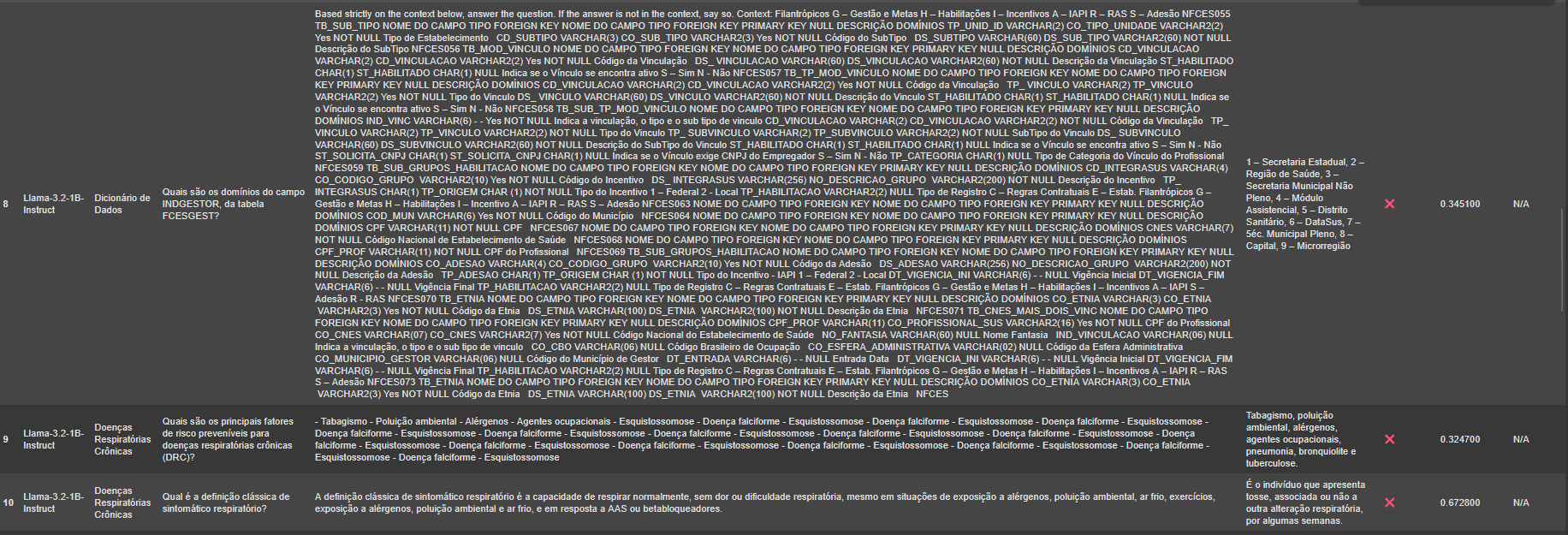
Resultados para: ***gemma-2b-it***

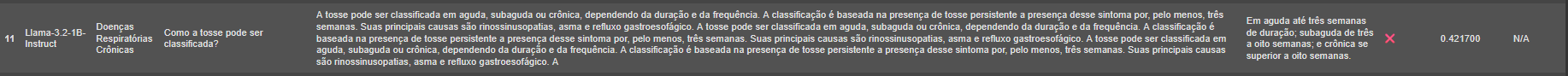




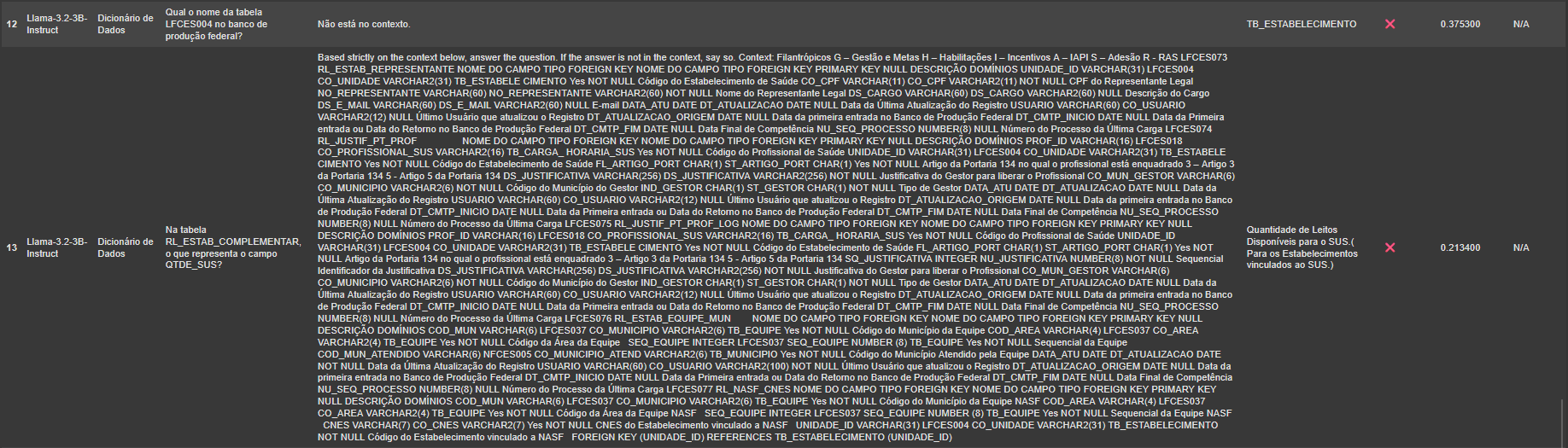
Resultados para: ***Llama-3.2-1B-Instruct***

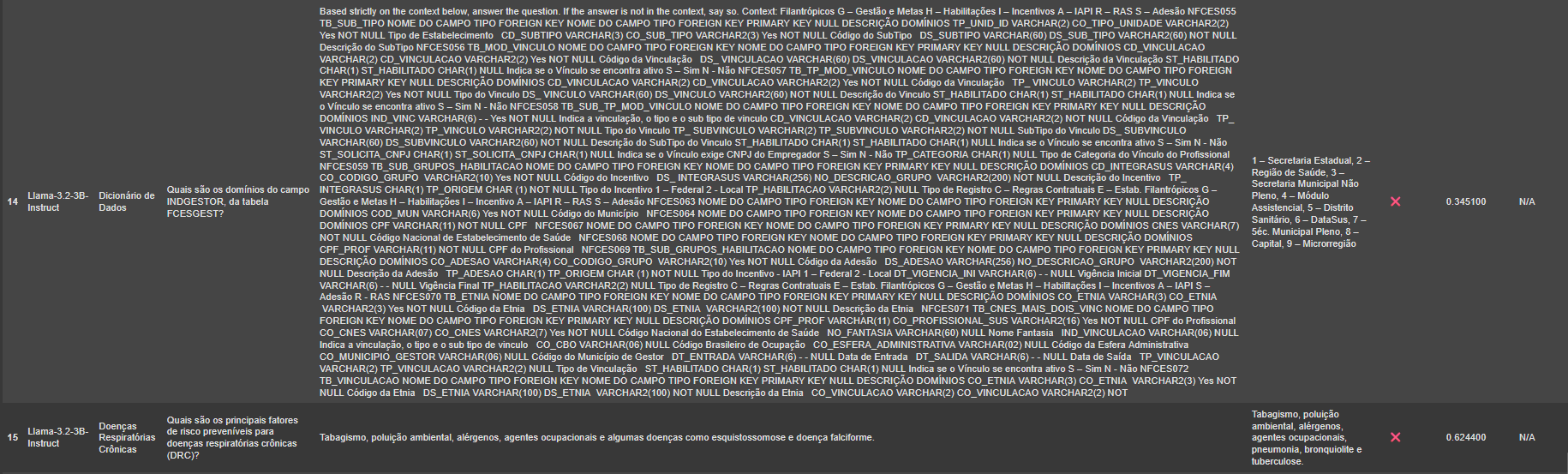


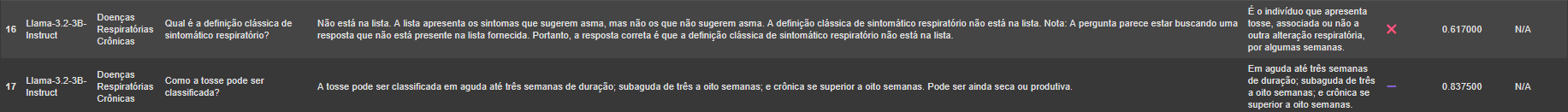




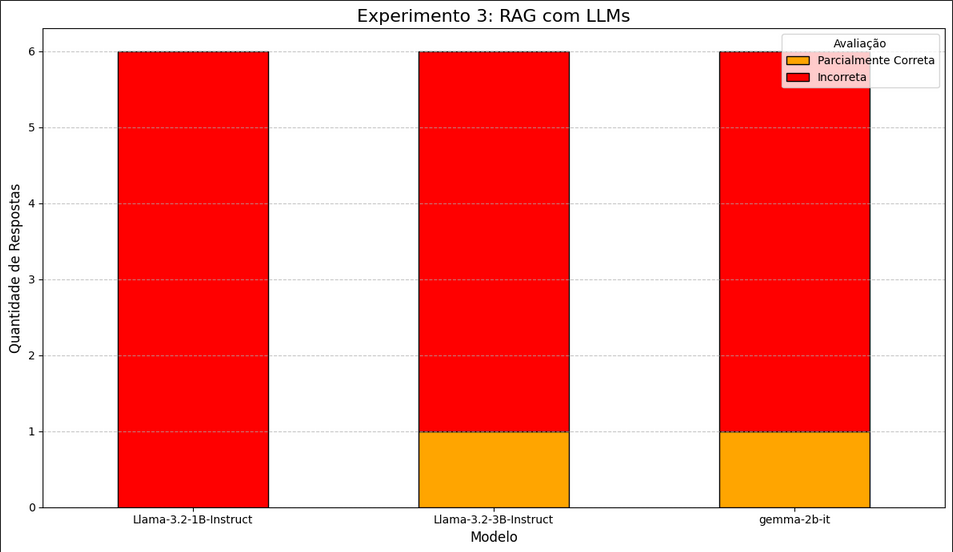
Resultados para: ***Llama-3.2-3B-Instruct***







O gráfico de barras empilhadas abaixo resume a avaliação manual, mostrando a quantidade de respostas corretas, parcialmente corretas e incorretas para cada modelo.



# 5. Conclusão

A atividade demonstrou que a escolha do modelo de Question Answering é crucial para o sucesso da tarefa. Modelos mais leves como o DistilBERT ou mesmo modelos robustos como o RoBERTa podem não ter o desempenho esperado em domínios específicos ou com textos em português, especialmente quando o conteúdo é estruturado (como em tabelas).  
 O modelo timpal0l/mdeberta-v3-base-squad2, de base multilíngue, mostrou-se mais eficaz para os documentos analisados, conseguindo extrair informações relevantes tanto do texto corrido do PDF quanto dos dados tabulares do DOCX. Isso reforça a importância de testar e validar diferentes arquiteturas de modelos para encontrar a mais adequada a um determinado caso. de usos nfdjfnjs s njndjfnsdjfn jnfj sdjn jsdnfj njdn js ndj jdnfjs jdsnfjsd  
 A avaliação também destacou que a pontuação de confiança de um modelo nem sempre se correlaciona diretamente com a precisão da resposta. Portanto, uma combinação de métricas quantitativas, como a similaridade de cosseno, e uma avaliação qualitativa manual é fundamental para uma análise de desempenho completa e confiável.

Os três experimentos demonstraram que a forma de recuperar e selecionar o contexto é determinante para o desempenho. Modelos de QA tradicionais se beneficiam de receber o texto integral, enquanto LLMs maiores exigem estratégias de RAG. Entretanto, tabelas continuam sendo um desafio para a divisão em chunks. O Experimento 3 mostrou potencial, mas depende de otimização na recuperação e gerenciamento de contexto.

# 6. Referências

* Hugging Face. *Transformers Documentation*. Disponível em: <https://huggingface.co/docs/transformers/index.>
* Hugging Face. *Sentence-Transformers Documentation*. Disponível em: <https://huggingface.co/sentence-transformers.>
* Model Card. *deepset/roberta-base-squad2*. Disponível em: <https://huggingface.co/deepset/roberta-base-squad2.>
* Model Card. *distilbert-base-cased-distilled-squad*. Disponível em: <https://huggingface.co/distilbert-base-cased-distilled-squad.>
* Model Card. *timpal0l/mdeberta-v3-base-squad2*. Disponível em: <https://huggingface.co/timpal0l/mdeberta-v3-base-squad2.>

# 7. Repositório

<https://github.com/shykiu77/PLN_QA_LLM_2025_Melo_Carlos/tree/main>

# 8. Notebook

1ª Versão: [PLN\_QA\_LLM\_v1\_2025.ipynb](https://colab.research.google.com/drive/1v7O56jkR_l8kB-sMQdtzuRnsvMM4CG5B?usp=sharing)

2ª Versão: [PLN\_QA\_LLM\_v2\_2025.ipynb](https://colab.research.google.com/drive/1fdpZu_C6IEDouc1gf7OntnShtuxF2a3M?usp=sharing)

# 9. Vídeo

* [URL para o vídeo da apresentação]

# 10. Participação

* Arthur Matheus: Implementação do código e escrita do documento
* Carlos Melo: Implementação do código e escrita do documento