UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Chatbots inteligentes para automatização do atendimento ao cliente: explorando diferentes métodos de implementação

André Edson Regazzo de Morais

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

André Edson Regazzo de Morais

Chatbots inteligentes para automatização do atendimento ao cliente: explorando diferentes métodos de implementação

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Antonio Rafael Sabino Parmezan

Versão original

São Carlos

2025

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

R333c

Regazzo, André
Chatbots inteligentes para automatização do atendimento ao cliente: explorando diferentes métodos de implementação / André Regazzo; orientador Prof. Dr. Antonio Rafael Sabino Parmezan; coorientadora Profa. Dra. Solange Oliveira Rezende. -- São Carlos, 2025.
89 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2025.

1. Chatbots. 2. Inteligencia Artificial. 3. Rag. 4. Fine Tuning. I. Rafael Sabino Parmezan, Prof. Dr. Antonio , orient. II. Oliveira Rezende, Profa. Dra. Solange , coorient. III. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

André Edson Regazzo de Morais

Intelligent Chatbots for Customer Service Automation: Exploring Different Implementation Methods

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Antonio Rafael Sabino Parmezan

Original version

São Carlos

2025

Este trabalho é dedicado aos alunos da USP, como uma contribuição das Bibliotecas do Campus USP de São Carlos para o desenvolvimento e disseminação da pesquisa científica da Universidade.

AGRADECIMENTOS

Agradeço, com todo o meu amor, à minha esposa, por sempre acreditar em mim, por escolher compartilhar esta jornada ao meu lado e por me presentear com nosso amado filho.

Agradeço ao meu filho, por me acolher em sua vida e por me dar a alegria de conviver com minhas duas lindas netinhas.

À minha nora, Lutsy, expresso minha sincera gratidão por ser uma mãe dedicada e amorosa para minhas netas, por cuidar do meu filho com tanto carinho e atenção, e por me acolher com generosidade como parte de sua família.

Aos meus orientadores, deixo meu profundo agradecimento pelo apoio contínuo, pela orientação técnica e pela confiança demonstrada ao longo de todo o processo.

À memória da minha mãe, agradeço profundamente por me ensinar, com o exemplo, que a educação tem o poder de transformar vidas.

[3]

Uso de ferramentas de inteligência artificial no apoio à escrita

Durante a elaboração deste trabalho, foram utilizadas ferramentas baseadas em inteligência artificial com o objetivo exclusivo de apoiar a estruturação textual. Esses recursos contribuíram para o aprimoramento da linguagem, melhoria da clareza e auxílio na tradução de alguns trechos. Ressalta-se que nenhuma ferramenta foi utilizada para gerar ideias, interpretar resultados ou desenvolver conteúdo acadêmico original. Todas as decisões conceituais, metodológicas e argumentativas refletem exclusivamente a autoria humana.

"Nunca tive medo de perguntar demais. Meu receio sempre foi aceitar as coisas como são sem questionar. Acredito que é na pergunta que começa qualquer mudança."

André Regazzo

RESUMO

Morais, A. ***Chatbots* inteligentes para automatização do atendimento ao cliente: explorando diferentes métodos de implementação**. 2025. 89 p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

Os avanços em Inteligência Artificial (IA) e Processamento de Linguagem Natural (PLN) têm revolucionado a interação entre empresas e clientes, por exemplo, ao permitir que assistentes virtuais compreendam e respondam a perguntas frequentes de forma automatizada e em tempo real, ou ao oferecer recomendações personalizadas com base no histórico de interações do usuário. *Chatbots* inteligentes surgem como ferramentas essenciais para otimizar processos como o atendimento ao cliente, o suporte técnico e o direcionamento de demandas internas, mas sua implementação eficaz, especialmente em empresas com dados restritos, apresenta desafios. Este estudo propõe avaliar um *chatbot* comercial para atendimento ao cliente, comparando três abordagens distintas: (i) uso direto de um *Large Language Model (LLM)* via API; (ii) *Retrieval-Augmented Generation (RAG)*; e (iii) *Fine-Tuning* de LLMs. O objetivo é analisar a complexidade, custos, tempo de implementação e qualidade das respostas em um cenário com dados limitados, fornecendo *insights* para empresas que buscam automatizar o atendimento ao cliente de forma eficiente.

Palavras-chave: Fine-Tuning, Retrieval-Augmented Generation, Large Language Models, Chatbots, Automação de Atendimento, Processamento de Linguagem Natural, Inteligência Artificial.

ABSTRACT

Morais, A. **Intelligent Chatbots for Customer Service Automation: Exploring Different Implementation Methods**. 2025. 89 p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

Advancements in Artificial Intelligence (AI) and Natural Language Processing (NLP) have transformed the interaction between companies and customers, for example, by enabling virtual assistants to understand and answer frequently asked questions automatically and in real time, or by offering personalized recommendations based on user interaction history. Intelligent chatbots have become essential tools for optimizing processes such as customer service, technical support, and internal demand routing. However, their effective implementation, especially in companies with restricted data, poses significant challenges. This study aims to evaluate a commercial chatbot for customer service by comparing three distinct approaches: (i) direct use of a Large Language Model (LLM) via API; (ii) Retrieval-Augmented Generation (RAG); and (iii) Fine-Tuning of LLMs. The objective is to analyze the complexity, costs, implementation time, and quality of responses in a scenario with limited data, providing insights for companies seeking to automate customer service efficiently.

Keywords: Fine-Tuning, Retrieval-Augmented Generation, Large Language Models, Chatbots, Customer Service Automation, Natural Language Processing, Artificial Intelligence.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
BERTScore	Bidirectional Encoder Representations from Transformers Score
BLEU	Bilingual Evaluation Understudy
GPT	Generative Pre-trained Transformer
HFT	Hugging Face Transformers
IA	Inteligência Artificial
LLM	Large Language Model
LoRA	Low-Rank Adaptation
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
RAG	Retrieval Augmented Generation
RNN	Recurrent Neural Network
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
SFT	Supervised Fine-Tuning

SUMÁRIO

1 INTRODUÇÃO

Técnicas de Inteligência Artificial e Processamento de Linguagem Natural vêm sendo aplicadas em diferentes domínios para interpretar linguagem humana e automatizar a resposta a conteúdos complexos — incluindo desinformação, atendimento ao cliente e análise de sentimentos (Maynard; Funk, 2020). Isso ocorre porque essas tecnologias permitem que sistemas automatizados compreendam linguagem humana, o que viabiliza a criação de chatbots capazes de fornecer informações, tirar dúvidas e realizar triagens iniciais no atendimento, de forma rápida e contínua (Collins *et al.*, 2021). Nesse contexto, os chatbots inteligentes surgem como ferramentas essenciais para otimizar processos, reduzir custos e oferecer suporte contínuo – operando 24 horas por dia. Construídos com base em técnicas de `IA` e `PLN`, esses sistemas permitem interações automatizadas por meio da linguagem humana (Russell; Norvig, 2003).

Entretanto, apesar das inúmeras vantagens dos chatbots, como a disponibilidade contínua, redução de custos com atendimento humano e padronização das respostas, sua implementação ainda enfrenta importantes limitações. Entre os principais desafios estão a adaptação a diferentes contextos culturais, a interpretação correta de intenções ambíguas e a capacidade de fornecer respostas precisas a partir de dados limitados. Por exemplo, em uma empresa de tecnologia com um acervo restrito a materiais institucionais e descrições de serviços, um chatbot pode não conseguir interpretar adequadamente dúvidas específicas de clientes, ou responder com segurança quando confrontado com perguntas fora do escopo desses documentos. Essas limitações afetam diretamente a eficácia do atendimento e demandam a escolha criteriosa da abordagem tecnológica utilizada na construção do sistema (McTear, 2022).

A justificativa para este estudo reside na crescente necessidade de empresas, especialmente as de pequeno e médio porte com recursos e dados limitados, encontrarem soluções de automação de atendimento que sejam ao mesmo tempo eficazes e viáveis. A comparação direta entre essas três abordagens populares visa fornecer um guia prático para a tomada de decisão sobre qual método adotar, considerando as prioridades específicas de cada organização.

Este estudo propõe avaliar um chatbot comercial automatizado para o primeiro atendimento ao cliente, com o objetivo de fornecer informações institucionais, detalhes sobre serviços e dados de contato de uma empresa de tecnologia. Para isso, são examinadas três abordagens distintas: uso direto de `LLM` via `API` (integração dos documentos institucionais brutos diretamente na base de conhecimento de um assistente de linguagem de grande escala); `RAG` (*Retrieval-Augmented Generation*) (combinação de busca semântica com geração de texto, permitindo a consulta a documentos durante a formulação da resposta);

e *Fine-Tuning* de LLMs (ajuste do modelo aos dados específicos da empresa para obter respostas mais especializadas).

Estudos anteriores discutem essas abordagens de forma isolada ou aplicadas a contextos mais amplos, com maior volume de dados e recursos computacionais. No entanto, ainda são escassas as análises comparativas entre essas estratégias voltadas para a realidade de pequenas e médias empresas, que frequentemente operam com bases de conhecimento limitadas. Assim, este trabalho se propõe a preencher essa lacuna, avaliando a viabilidade e a eficácia dessas soluções em um cenário restrito, o que representa um diferencial prático e relevante em relação à literatura existente.

A análise comparativa dessas técnicas considera a complexidade, os custos de desenvolvimento, o tempo de implementação e a qualidade das respostas. A complexidade é avaliada com base no nível de especialização técnica necessário para implementar cada abordagem, na quantidade de etapas envolvidas e na dependência de ferramentas externas. Os custos incluem recursos computacionais, licenciamento de serviços e horas de trabalho dedicadas ao desenvolvimento. O tempo de implementação refere-se ao período necessário para configurar, treinar e integrar cada solução em um ambiente funcional. A análise é realizada em um cenário típico de dados limitados – geralmente composto por materiais institucionais, FAQs e descritivos de serviços. O estudo propõe responder às seguintes questões de pesquisa:

1. Qual abordagem (API direta, RAG ou *Fine-Tuning*) apresenta melhor desempenho em métricas de qualidade de resposta (similaridade semântica, precisão, revocação e F1 score) ao utilizar uma base de conhecimento restrita?

2. Qual abordagem demonstra maior eficiência em termos de tempo de resposta?

3. Qual o *trade-off* observado entre complexidade de implementação, custo computacional, tempo de desenvolvimento e a qualidade/eficiência das respostas para cada método no cenário proposto?

1.1 Objetivos

O objetivo é analisar a complexidade, custos de desenvolvimento, tempo de implementação e qualidade das respostas em um cenário com dados limitados, fornecendo insights para empresas que buscam automatizar o atendimento ao cliente de forma eficiente.

Para alcançar o objetivo geral proposto, este estudo estabelece os seguintes objetivos específicos:

- Implementar um chatbot de atendimento ao cliente com base em três abordagens

distintas: uso direto de LLM via API, RAG (*Retrieval-Augmented Generation*) e *Fine-Tuning* de LLMs;

- Avaliar cada abordagem quanto à complexidade de implementação, custos de desenvolvimento envolvidos, tempo necessário para implantação e qualidade das respostas geradas;
- Realizar testes comparativos com base em métricas objetivas de desempenho (como similaridade semântica, precisão, revocação e F1 score), utilizando um conjunto limitado de dados institucionais;
- Identificar o *trade-off* entre esforço técnico, investimento financeiro e eficiência das respostas, fornecendo subsídios práticos para a tomada de decisão em contextos empresariais com restrições de recursos.

1.2 Estrutura do Trabalho

Além desta introdução, este trabalho está organizado da seguinte forma:

- **Capítulo 2 – Fundamentação Teórica**: apresenta os conceitos essenciais relacionados à IA, PLN e chatbots, além de descrever em detalhes as abordagens técnicas utilizadas no estudo;
- **Capítulo 3 – Metodologia**: descreve o desenho experimental, os critérios de avaliação e o processo de implementação das três abordagens comparadas;
- **Capítulo 4 – Avaliação Experimental**: apresenta os resultados obtidos nos testes, incluindo análise de custos e tempo de desenvolvimento, acompanhados de discussão comparativa e interpretação dos dados;
- **Capítulo 5 – Conclusões**: resume as principais conclusões, contribuições do estudo, limitações da pesquisa e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Considerações Iniciais

Este capítulo apresenta os principais conceitos, definições e abordagens relacionadas ao uso de chatbots em contextos empresariais, com ênfase em soluções baseadas em Inteligência Artificial (IA) e Processamento de Linguagem Natural (PLN). O capítulo está estruturado em seis seções principais: inicialmente, são discutidas as categorias fundamentais de chatbots e sua evolução histórica; em seguida, são apresentados os fundamentos técnicos de aprendizado de máquina e arquiteturas Transformer; posteriormente, são detalhadas as três estratégias específicas analisadas neste estudo — uso direto de LLM via API, RAG e *Fine-Tuning*; são também abordadas técnicas auxiliares como engenharia de *prompt* e implementação de *guard rails*; por fim, é apresentada uma revisão de trabalhos relacionados que contextualiza este estudo no estado da arte atual. Esses fundamentos teóricos fornecerão a base conceitual necessária para justificar as escolhas metodológicas e interpretar os resultados experimentais apresentados nos capítulos subsequentes.

2.1 Chatbots: Conceitos e Categorias

Os chatbots são sistemas computacionais desenvolvidos para simular interações humanas por meio de interfaces conversacionais, permitindo comunicação automatizada entre usuários e serviços digitais. Segundo McTear (2022), esses sistemas representam uma das aplicações mais visíveis da IA conversacional, atuando como intermediários inteligentes em diversos contextos, desde atendimento ao cliente até assistência técnica especializada.

A definição de Shevat (2017) caracteriza os bots como "*software-powered users that live inside our chat apps*", evidenciando sua função de replicar comportamentos humanos em ambientes digitais. Esta caracterização destaca a natureza híbrida desses sistemas, que combinam capacidades computacionais com interfaces de comunicação natural.

2.1.1 Taxonomia de Chatbots

A literatura especializada classifica os chatbots em duas categorias principais, baseadas em suas arquiteturas e capacidades de processamento:

2.1.1.1 Chatbots Baseados em Regras

Esta categoria opera através de sistemas determinísticos que utilizam regras e scripts predefinidos para responder a palavras-chave ou padrões específicos de entrada. Segundo Shevat (2017), os sistemas de chatbot mais tradicionais podem ser definidos como ferramentas que operam com base em estruturas rígidas de interação, nas quais os diálogos

seguem fluxos pré-estabelecidos e previsíveis. Esses sistemas se apoiam, predominantemente, em técnicas de correspondência de padrões (*pattern matching*), permitindo que as respostas sejam geradas de forma determinística, ou seja, sempre iguais diante dos mesmos estímulos (Shawar; Atwell, 2007).

Entre suas principais características, destacam-se a baixa complexidade computacional, a facilidade de implementação e manutenção, além da previsibilidade nas respostas. Essas qualidades conferem vantagens como agilidade nas interações, consistência nas informações fornecidas, custo reduzido de desenvolvimento e total controle sobre as saídas produzidas (Følstad; Brandtzæg, 2017).

No entanto, apesar desses benefícios, os chatbots baseados em fluxos fixos apresentam limitações expressivas. Dentre elas, a mais notável é a dificuldade em lidar com variações linguísticas e em compreender o contexto das interações. Além disso, esses sistemas não são escaláveis para domínios mais complexos, uma vez que não possuem flexibilidade nem adaptabilidade para lidar com situações que extrapolem seus scripts programados (Gnewuch; Morana; Maedche, 2017).

2.1.1.2 Chatbots Baseados em Inteligência Artificial

Em contraste, os chatbots baseados em `IA` utilizam técnicas de aprendizado de máquina e processamento de linguagem natural para interpretar comandos e contextos de forma dinâmica. Conforme detalhado por Shevat (2017), esses sistemas demonstram capacidade de aprender com interações passadas, aprimorando progressivamente a experiência do usuário.

Os chatbots baseados em `IA` representam uma evolução significativa em relação aos sistemas tradicionais de interação homem-máquina. Diferentemente dos modelos baseados em regras fixas, esses agentes conversacionais utilizam algoritmos de aprendizado de máquina para processar e interpretar linguagem natural, o que lhes confere uma capacidade muito mais ampla de adaptação e compreensão contextual. Isso significa que são capazes de identificar nuances na linguagem, reconhecer intenções do usuário mesmo com variações linguísticas e gerar respostas mais naturais e personalizadas (Géron, 2019).

Entre suas principais características, destacam-se a adaptabilidade, a aprendizagem contínua a partir das interações com os usuários e a capacidade de produzir respostas coerentes mesmo fora de scripts predefinidos (McTear, 2022). Esses sistemas também se beneficiam da flexibilidade para atuar em domínios diversos, com possibilidade de generalização para novos contextos, o que amplia significativamente seu potencial de aplicação, especialmente em ambientes como educação, saúde e atendimento ao cliente.

Entretanto, essa sofisticação técnica traz consigo algumas limitações importantes. A implementação de chatbots com `IA` demanda maior complexidade computacional, além

de exigir grandes volumes de dados para o treinamento dos modelos (Radziwill; Benton, 2017). Isso também acarreta custos mais elevados, tanto em termos de infraestrutura quanto de manutenção. Outra limitação recorrente refere-se à possibilidade de respostas imprecisas ou inapropriadas, sobretudo em situações onde o modelo não foi suficientemente treinado ou onde faltam dados contextualizados.

2.2 Fundamentos de Aprendizado de Máquina

O aprendizado de máquina é uma subárea da `IA` dedicada ao estudo de algoritmos computacionais que melhoram automaticamente através da experiência (Mitchell, 1997). No contexto de chatbots, três paradigmas são fundamentais para compreender as bases tecnológicas que sustentam os sistemas conversacionais modernos.

2.2.1 Aprendizado Supervisionado

O aprendizado supervisionado utiliza conjuntos de dados rotulados para treinar modelos. Nesta abordagem, o algoritmo aprende a partir de exemplos onde tanto a entrada quanto a saída desejada são conhecidas. O objetivo é induzir uma função que seja capaz de mapear novas entradas para saídas apropriadas, baseando-se nos padrões identificados durante o treinamento (Russell; Norvig, 2003). Este paradigma é fundamental para tarefas como classificação de intenções em chatbots, em que o sistema deve identificar o propósito da mensagem do usuário.

2.2.2 Redes Neurais e Arquiteturas Profundas

As redes neurais artificiais são modelos computacionais inspirados no funcionamento do cérebro humano, compostos por unidades de processamento interconectadas que simulam neurônios. Redes Neurais Recorrentes (`RNNs`) e suas variantes, como `LSTM` (*Long Short-Term Memory*), foram amplamente utilizadas para processamento sequencial de linguagem natural antes do advento dos `Transformers` (Goodfellow; Bengio; Courville, 2016). Essas arquiteturas são capazes de processar sequências de dados mantendo informações sobre elementos anteriores, característica essencial para compreensão de contexto em conversações.

2.2.3 Arquitetura Transformer

A arquitetura `Transformer`, introduzida por (Vaswani *et al.*, 2017), revolucionou o processamento de linguagem natural através do mecanismo de atenção. Diferentemente das `RNNs`, os `Transformers` processam sequências inteiras simultaneamente, permitindo paralelização eficiente e captura de dependências de longo alcance. O mecanismo de atenção calcula pesos para diferentes partes da sequência de entrada, permitindo que o modelo "foque"em informações relevantes. A atenção multi-cabeça permite que o modelo capture

diferentes tipos de relações simultaneamente, representando um avanço significativo na capacidade de compreensão contextual.

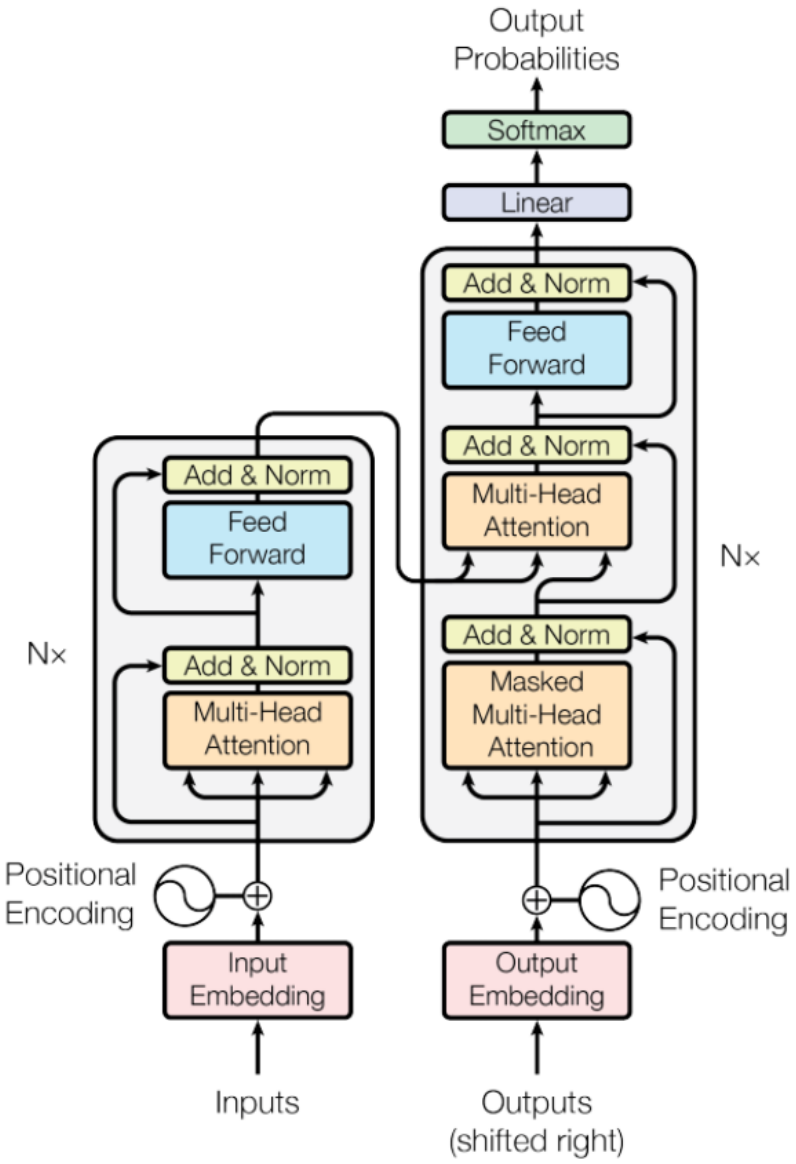

Figura 1 – A arquitetura Transformer, proposta por Vaswani et al. (2017), revolucionou o processamento de linguagem natural ao eliminar o uso de redes recorrentes. Baseia-se inteiramente em mecanismos de atenção, permitindo paralelização e escalabilidade superiores. Sua estrutura com codificadores e decodificadores facilita tarefas como tradução automática, resumo e geração de texto.

2.2.4 Large Language Models (LLMs)

Os `LLMs` são modelos de linguagem baseados em `Transformers` treinados em grandes corpora textuais. Modelos como `GPT` (*Generative Pre-trained Transformer*) e `BERT` (*Bidirectional Encoder Representations from Transformers*) demonstram capacidades emergentes de compreensão e geração de linguagem natural (Devlin *et al.*, 2018), (Radford

et al., 2019). Esses modelos são treinados em duas fases: pré-treinamento em grandes volumes de texto não rotulado para aprender representações linguísticas gerais, seguido de ajuste fino para tarefas específicas. Esta abordagem permite que os modelos desenvolvam conhecimento factual amplo e capacidades de raciocínio que emergem apenas acima de determinados limiares de tamanho e complexidade.

2.2.5 Embeddings e Representação Semântica

Embeddings são representações vetoriais densas que capturam relações semânticas entre palavras, sentenças ou documentos. Modelos como `Word2Vec`, `GloVe` e mais recentemente `BERT` e `Sentence-BERT` permitem que computadores processem linguagem natural de forma mais eficiente(Mikolov *et al.*, 2013), (Reimers; Gurevych, 2019). Essas representações vetoriais possibilitam que palavras com significados similares sejam posicionadas próximas no espaço vetorial, facilitando tarefas como busca semântica e comparação de similaridade textual, fundamentais para sistemas `RAG`.

2.3 Abordagens de Implementação de Chatbots

Esta seção detalha as três abordagens principais analisadas neste estudo, fornecendo fundamentação técnica para compreender suas características e aplicabilidades em contextos empresariais.

2.3.1 Uso Direto de LLM via API

Esta abordagem utiliza modelos de linguagem pré-treinados através de interfaces de programação (`APIs`). O sistema envia consultas diretamente para serviços como `OpenAI GPT`, `Anthropic Claude` ou `Google Gemini`, que processam a entrada e retornam respostas geradas. A implementação técnica envolve a estruturação de *prompts* que incluem contexto específico do domínio, instruções de comportamento e a consulta do usuário. O modelo utiliza seu conhecimento pré-treinado combinado com o contexto fornecido para gerar respostas apropriadas (Brown *et al.*, 2020).

Esta estratégia oferece vantagens significativas como implementação rápida, acesso a modelos estado-da-arte e manutenção simplificada, uma vez que a infraestrutura computacional é gerenciada pelo provedor. No entanto, apresenta limitações como dependência de serviços externos, custos por uso que podem ser significativos em escala e controle limitado sobre o modelo base.

2.3.2 RAG (Retrieval-Augmented Generation)

`RAG` combina recuperação de informações com geração de linguagem natural, representando uma abordagem híbrida que foi introduzida por Lewis *et al.* (2020) como uma solução para limitações de conhecimento em modelos generativos. O processo técnico

envolve quatro etapas principais: indexação de documentos usando *embeddings* semânticos, recuperação de trechos relevantes baseada na consulta do usuário através de similaridade vetorial, concatenação dos trechos recuperados com a consulta original e geração da resposta usando um LLM.

Esta abordagem oferece vantagens como acesso a informações atualizadas, transparência nas fontes utilizadas e controle sobre a base de conhecimento. Entretanto, apresenta limitações como complexidade de implementação, dependência da qualidade da recuperação de informações e latência adicional devido ao processo de busca.

2.3.3 Fine-Tuning de LLMs

Fine-tuning adapta um modelo pré-treinado para um domínio específico através de treinamento adicional com dados especializados. Técnicas modernas como LoRA (*Low-Rank Adaptation*) permitem adaptação eficiente sem modificar todos os parâmetros do modelo (Hu *et al.*, 2021). O processo envolve preparação de dados de treinamento específicos do domínio, configuração de hiperparâmetros de treinamento, execução do treinamento com técnicas de regularização e avaliação do modelo adaptado.

Esta estratégia oferece vantagens como especialização para o domínio específico, controle total sobre o modelo e independência de serviços externos. No entanto, requer dados de treinamento de qualidade, recursos computacionais significativos e expertise técnica para implementação e manutenção adequadas.

2.4 Técnicas Auxiliares

A otimização da interação com LLMs requer técnicas especializadas para maximizar a qualidade e controlar o comportamento dos modelos. Esta seção aborda duas técnicas fundamentais: engenharia de *prompt* e implementação de *guard rails*.

2.4.1 Engenharia de Prompt

A engenharia de *prompt* constitui uma disciplina emergente focada no design e otimização de instruções para LLMs. Esta prática envolve a criação sistemática de comandos que maximizam a probabilidade de obter respostas desejadas dos modelos (Wei *et al.*, 2022). As técnicas incluem *zero-shot prompting*, em que o modelo executa tarefas sem exemplos específicos dependendo de seu conhecimento pré-treinado, *few-shot prompting*, que fornece poucos exemplos no *prompt* para demonstrar o padrão desejado, e *chain-of-thought prompting*, técnica que solicita ao modelo que explicite seu raciocínio passo a passo, melhorando significativamente o desempenho em tarefas que requerem raciocínio complexo.

Um *prompt* bem estruturado deve incluir contexto específico do domínio, instruções claras sobre o comportamento esperado, exemplos quando aplicável, restrições e *guard*

rails comportamentais, além de especificação da estrutura da resposta. A qualidade do *prompt* impacta diretamente na eficácia do sistema conversacional.

2.4.2 Guard Rails

Guard rails representam mecanismos de controle implementados para garantir que as respostas dos `LLMs` permaneçam dentro de parâmetros aceitáveis e seguros. Esses sistemas atuam como filtros e verificadores que previnem comportamentos indesejados (Gehman *et al.*, 2020). Os tipos incluem *guard rails* de entrada, que filtram consultas inadequadas ou maliciosas e detectam tentativas de *prompt injection*, *guard rails* de saída, que verificam a adequação das respostas geradas e detectam conteúdo potencialmente prejudicial, e *guard rails* de processo, que monitoram o comportamento durante a geração e controlam recursos computacionais utilizados.

A implementação pode utilizar abordagens baseadas em regras, como listas de palavras-chave proibidas e padrões de expressões regulares, ou baseadas em `ML`, como classificadores de toxicidade e modelos de detecção de viés. A escolha depende dos requisitos específicos de segurança e recursos disponíveis.

2.5 Métricas de Avaliação

A avaliação de sistemas conversacionais requer métricas especializadas que capturem aspectos semânticos e contextuais das respostas geradas.

2.5.1 BERTScore

`BERTScore` utiliza representações contextuais do modelo `BERT` para avaliar similaridade semântica entre texto gerado e referência. Esta métrica calcula precisão, revocação e F1 baseados em similaridade de *embeddings* ao nível de *tokens* (Zhang *et al.*, 2019). Diferentemente de métricas tradicionais como `BLEU` que dependem de correspondência exata de palavras, `BERTScore` considera o significado semântico, oferecendo avaliação mais robusta para tarefas de geração de linguagem natural.

2.5.2 Sentence-BERT

`Sentence-BERT` modifica a arquitetura `BERT` para produzir *embeddings* de sentenças semanticamente significativos, permitindo comparação eficiente de similaridade entre textos (Reimers; Gurevych, 2019). Esta abordagem resolve limitações do `BERT` original para tarefas de similaridade semântica, possibilitando avaliação de qualidade de respostas conversacionais através de comparação vetorial.

2.6 Considerações para Empresas de Pequeno e Médio Porte

Empresas de pequeno e médio porte frequentemente enfrentam desafios específicos na implementação de chatbots baseados em IA, particularmente devido à limitação de recursos e dados disponíveis. Diferentemente de grandes corporações que possuem extensas bases de dados e recursos computacionais, essas organizações geralmente dispõem apenas de documentação básica, como manuais, FAQs e materiais de marketing (Adamopoulou; Moussiades, 2020).

Esta realidade impacta diretamente na escolha da abordagem tecnológica, exigindo consideração cuidadosa de fatores como custo de implementação, complexidade técnica, tempo de desenvolvimento e retorno sobre investimento. O equilíbrio entre eficiência, flexibilidade e viabilidade econômica torna-se, portanto, um fator determinante na seleção da estratégia mais adequada.

2.7 Evolução Histórica dos Chatbots

A evolução dos chatbots pode ser compreendida através de três fases históricas distintas, cada uma caracterizada por avanços tecnológicos específicos e mudanças paradigmáticas nas abordagens de desenvolvimento.

2.7.1 Primeira Fase (1960-2010): Sistemas Baseados em Regras

O período inicial da história dos chatbots foi marcado pelo desenvolvimento de sistemas pioneiros baseados em regras fixas e técnicas de correspondência de padrões. O exemplo mais emblemático desta era é o ELIZA, desenvolvido por Joseph Weizenbaum no MIT entre 1964-1966 (Jain *et al.*, 2018). O ELIZA operava através de scripts pré-estabelecidos que realizavam substituições textuais simples, sem qualquer compreensão real do contexto conversacional. O sistema utilizava técnicas de *pattern matching* para identificar palavras-chave nas entradas do usuário e gerar respostas baseadas em *templates* predefinidos.

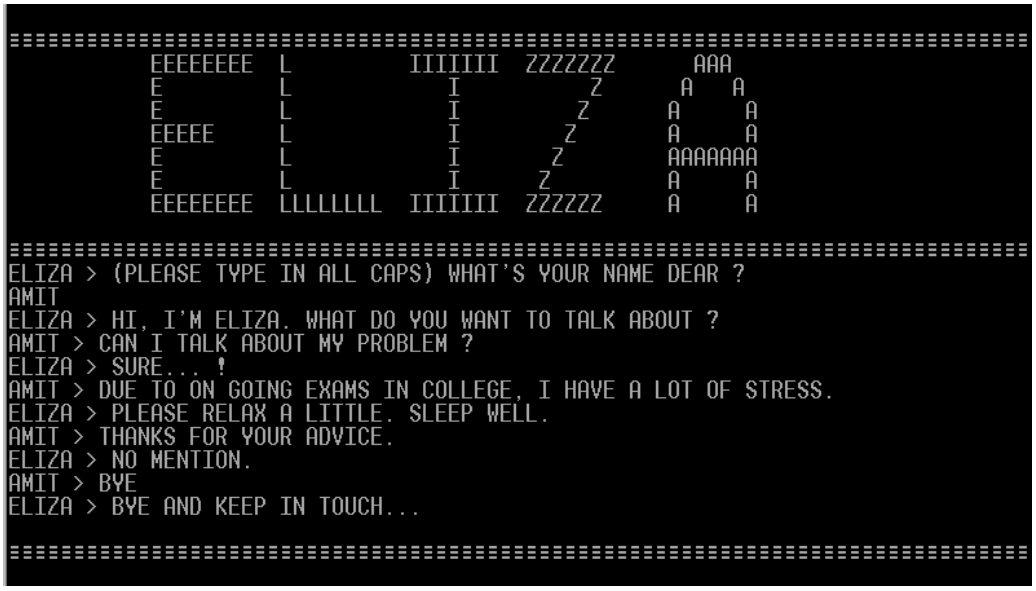

Figura 2 – Interface do chatbot ELIZA em execução. *Criado por Joseph Weizenbaum no MIT na década de 1960, o ELIZA é considerado o primeiro chatbot da história, operando por meio de substituições textuais baseadas em regras fixas.*

Outros sistemas notáveis deste período incluem o `PARRY` (1972) e o `A.L.I.C.E.` (1995), que introduziu o uso da linguagem `AIML` (*Artificial Intelligence Markup Language*) para estruturação de conhecimento conversacional.

2.7.2 Segunda Fase (2010-2020): Integração com Aprendizado de Máquina

A segunda fase foi caracterizada pela incorporação de técnicas de aprendizado de máquina e processamento de linguagem natural mais sofisticadas. Este período testemunhou a transição de sistemas puramente baseados em regras para abordagens híbridas que combinavam regras com capacidades de aprendizado. Com o avanço da tecnologia e das técnicas de processamento de linguagem natural, a segunda fase da evolução dos chatbots foi marcada pela introdução de mecanismos mais sofisticados de interpretação e geração de linguagem (McTear, 2022).

Um dos marcos centrais dessa etapa foi o uso de classificadores de intenção (*intent classification*), que permitiram aos sistemas identificar, com maior precisão, o propósito subjacente às mensagens dos usuários. Além disso, os chatbots passaram a incorporar ferramentas de reconhecimento de entidades nomeadas (*Named Entity Recognition* – `NER`), capazes de extrair informações específicas, como nomes, datas, locais ou produtos, dentro do texto da entrada do usuário. Tecnologicamente, esse período também foi caracterizado pelo emprego de redes neurais recorrentes (`RNN`) e suas variantes mais avançadas, como as `LSTMs`, que possibilitaram o processamento sequencial da linguagem com retenção de contexto ao longo da conversação.

Apesar dos avanços, esses sistemas ainda enfrentavam limitações significativas em termos de compreensão contextual e geração de linguagem natural fluente.

2.7.3 Terceira Fase (2020-Presente): Era dos Large Language Models

A fase atual representa uma revolução paradigmática no desenvolvimento de chatbots, impulsionada pelo advento de arquiteturas `Transformer` e `Large Language Models`. Esta era foi inaugurada pelo lançamento do `GPT-3` em 2020 e consolidada com o `ChatGPT` em 2022. Os modelos contemporâneos demonstram capacidades sem precedentes de compreensão e geração de linguagem natural, aproximando-se significativamente da fluência humana em muitos contextos.

A terceira fase da evolução dos chatbots é caracterizada pela incorporação de modelos de linguagem de larga escala baseados na arquitetura `Transformer`, originalmente proposta por Vaswani *et al.* (2017). Esse novo paradigma rompeu com as limitações das redes neurais recorrentes, ao permitir o processamento paralelo e a atenção contextual distribuída ao longo de grandes sequências textuais.

Uma das marcas dessa geração de agentes conversacionais é o surgimento de capacidades emergentes de raciocínio e generalização, que não foram explicitamente programadas, mas que emergem a partir da complexidade dos dados e do tamanho dos modelos. Isso inclui a habilidade de resolver tarefas para as quais o sistema nunca foi diretamente treinado, demonstrando formas rudimentares de inferência, síntese e adaptação linguística.

Esta evolução estabelece o contexto tecnológico no qual se inserem as abordagens analisadas neste estudo, evidenciando a importância de compreender as capacidades e limitações dos `LLMs` modernos.

Figura 3 – Exemplo de agente LLM especializado. *O Orientador Virtual MBA USP, criado com o ChatGPT 4o, demonstra o uso prático de um modelo de linguagem de última geração, com personalização via instruções e contexto institucional.*

2.8 Trabalhos Relacionados

Esta seção apresenta uma revisão sistemática de trabalhos relevantes que contextualizam este estudo no estado da arte atual de chatbots empresariais e técnicas de implementação de `LLMs`.

2.8.1 Estudos Comparativos de Abordagens de LLM

Lewis *et al.* (2020) introduziram a técnica `RAG`, demonstrando sua eficácia em tarefas de resposta a perguntas quando comparada a modelos puramente generativos. O estudo evidenciou melhorias significativas em precisão factual e capacidade de incorporar conhecimento atualizado. (Hu *et al.*, 2021) propuseram a técnica `LoRA` como uma alternativa eficiente ao *fine-tuning* completo, demonstrando que adaptações de baixo posto podem alcançar *performance* comparável com redução drástica de parâmetros treináveis. Este trabalho estabeleceu as bases para técnicas de `PEFT` amplamente utilizadas atualmente.

Ouyang *et al.* (2022) documentaram o processo de alinhamento de `LLMs` através de *Reinforcement Learning from Human Feedback* (`RLHF`), demonstrando como técnicas de aprendizado por reforço podem melhorar significativamente a qualidade e segurança das respostas geradas.

2.8.2 Aplicações Empresariais de Chatbots

Adamopoulou e Moussiades (2020) conduziram uma revisão abrangente de chatbots em contextos empresariais, identificando fatores críticos de sucesso e principais desafios

de implementação. O estudo destacou a importância da escolha adequada da arquitetura baseada nas características específicas do domínio de aplicação.

2.8.3 Avaliação de Sistemas Conversacionais

Zhang *et al.* (2019) introduziram a métrica `BERTScore`, baseada em *embeddings* semânticos gerados por modelos `BERT`, como uma alternativa superior a métricas tradicionais como `BLEU` e `ROUGE`. Essa métrica avalia a similaridade semântica entre a resposta gerada e a resposta de referência, sendo amplamente adotada para avaliar tarefas de geração de linguagem natural, incluindo diálogos.

Reimers e Gurevych (2019) desenvolveram o modelo `Sentence-BERT`, uma modificação do `BERT` que permite o cálculo eficiente de similaridade semântica entre sentenças. Essa contribuição foi fundamental para tarefas de avaliação automatizada de diálogos e implementação de mecanismos de recuperação de informações em sistemas `RAG`.

2.8.4 Limitações em Cenários de Dados Escassos

Brown *et al.* (2020), ao introduzirem o `GPT-3`, demonstraram que grandes modelos de linguagem apresentam capacidades notáveis de *few-shot learning*, sendo capazes de executar tarefas com apenas alguns exemplos fornecidos na entrada. Essa característica marcou uma mudança de paradigma, evidenciando o potencial das `LLMs` para aplicações em domínios onde dados anotados são escassos ou inexistentes.

2.8.5 Posicionamento deste Estudo

Este trabalho diferencia-se dos estudos anteriores ao focar especificamente na comparação sistemática de três abordagens principais (`API` direta, `RAG` e *fine-tuning*) em um cenário controlado de dados institucionais limitados. Enquanto trabalhos anteriores frequentemente analisam essas técnicas isoladamente ou em contextos de grandes volumes de dados, este estudo preenche uma lacuna importante ao avaliar sua viabilidade e eficácia em cenários típicos de pequenas e médias empresas.

A contribuição principal reside na análise comparativa rigorosa dessas abordagens usando métricas padronizadas e um protocolo experimental replicável, fornecendo evidências empíricas para orientar decisões práticas de implementação em contextos empresariais com recursos limitados.

2.9 Considerações Finais

Este capítulo estabeleceu os fundamentos teóricos necessários para compreender e avaliar as abordagens de implementação de chatbots analisadas neste estudo. Foram apresentados os conceitos fundamentais de chatbots e sua evolução histórica, os fundamentos técnicos de aprendizado de máquina e arquiteturas `Transformer`, as três estratégias

específicas de implementação, técnicas auxiliares de otimização e uma revisão abrangente de trabalhos relacionados.

A fundamentação apresentada evidencia que a escolha da abordagem mais adequada para implementação de chatbots empresariais depende de múltiplos fatores, incluindo disponibilidade de dados, recursos computacionais, expertise técnica e requisitos específicos de qualidade e latência. O embasamento teórico fornecido neste capítulo sustentará as decisões metodológicas e facilitará a interpretação dos resultados experimentais apresentados nos capítulos subsequentes.

A revisão de trabalhos relacionados posiciona este estudo no contexto do estado da arte atual, destacando sua contribuição específica para o campo através da análise comparativa sistemática em cenários de dados limitados, uma lacuna identificada na literatura existente.

3 MÉTODOS E PROCEDIMENTOS

Este capítulo detalha a metodologia empregada para realizar a análise comparativa entre três abordagens de implementação de chatbots inteligentes: (i) uso direto de `LLM` via `API`; (ii) `Retrieval-Augmented Generation` (`RAG`); e (iii) `Fine-Tuning` de um modelo `Mistral`. O objetivo é avaliar sua eficácia em automatizar o primeiro atendimento ao cliente em uma empresa de tecnologia com base de dados institucionais restrita. A escolha dessas três estratégias reflete as práticas mais utilizadas na atualidade para aplicação de *Large Language Models* (`LLMs`) em contextos comerciais.

A metodologia proposta visa criar um ambiente experimental controlado, padronizado e replicável para mensurar o desempenho de cada abordagem em termos de qualidade das respostas e eficiência temporal. Para isso, este capítulo apresenta a descrição do ambiente computacional, o conjunto de dados utilizado, os processos de implementação de cada técnica, as métricas de avaliação empregadas e o procedimento experimental completo.

3.1 Ambiente Experimental

Os experimentos foram conduzidos utilizando a plataforma `Google Colab`[1], um ambiente de desenvolvimento em nuvem amplamente adotado para aplicações em ciência de dados e aprendizado de máquina. O `Google Colab` oferece acesso gratuito a recursos computacionais como `GPUs` (por exemplo, `NVIDIA A100` e `T4`), que foram fundamentais para acelerar tarefas como o `fine-tuning` e a geração de respostas em larga escala, mantendo a viabilidade do experimento sem a necessidade de infraestrutura local.

A linguagem utilizada foi `Python`[2], devido à sua ampla aceitação em projetos de Inteligência Artificial e à disponibilidade de bibliotecas específicas. O ambiente foi composto pelas seguintes bibliotecas:

- **`pandas`**[3] e **`openpyxl`**[4]: para leitura, escrita e manipulação dos arquivos de entrada e saída em formato `Excel`

- **`transformers` (Hugging Face)**[5]: biblioteca para acesso e uso de modelos de linguagem pré-treinados

[1] https://colab.research.google.com/
[2] https://www.python.org/
[3] https://pandas.pydata.org/
[4] https://openpyxl.readthedocs.io/en/stable/
[5] https://huggingface.co/docs/transformers

- **`datasets` (Hugging Face)**[6]: para carregamento e particionamento dos dados em treino e validação
- **`unsloth`**[7]: para aceleração do `fine-tuning` de modelos com técnicas como `LoRA`. Esta biblioteca oferece otimizações significativas na eficiência computacional, tornando viável o *fine-tuning* em ambientes com recursos limitados como o `Google Colab`
- **`sentence-transformers`**[8]: para geração de *embeddings* semânticos e cálculo de similaridade
- **`bert-score`**[9]: para avaliação da qualidade textual utilizando *embeddings* do `BERT`
- **`optuna`**[10]: para otimização automatizada de hiperparâmetros
- **`openai`**[11]: biblioteca oficial para integração via `API` com os modelos da `OpenAI`

O código-fonte foi estruturado em notebooks `Jupyter` (`.ipynb`) para testes interativos e em scripts `Python` reutilizáveis (`.py`), organizados em arquivos como `utils.py`, `gerador_respostas.py` e `avaliador_respostas.py`, garantindo modularidade, clareza e reprodutibilidade dos experimentos.

3.2 Conjunto de Dados

O conjunto de dados utilizado neste estudo foi desenvolvido a partir de materiais institucionais e de marketing da empresa fictícia `Grupo Regazzo`. Essa construção simulou um cenário realista de empresas de pequeno e médio porte com documentação limitada e linguagem comercial não estruturada para aplicações diretas em sistemas automatizados. O processo de preparação seguiu uma abordagem baseada em *pipeline* de mineração de dados, estruturado em três etapas principais: *aquisição*, *refinamento* e *formatação para FAQ*.

3.2.1 Aquisição dos Dados Brutos

Inicialmente, foram coletados documentos textuais institucionais, tais como brochuras, apresentações e páginas de marketing. Este conteúdo original apresentava alta redundância e trechos com vocabulário promocional pouco informativo, típicos de materiais voltados à divulgação comercial. A extração manual permitiu consolidar essas fontes em um único documento `PDF`, denominado `base_regazzo.pdf`, com foco exclusivo em informações úteis para fins de atendimento ao cliente.

[6] https://huggingface.co/docs/datasets
[7] https://github.com/unslothai/unsloth
[8] https://www.sbert.net/
[9] https://github.com/Tiiiger/bert_score
[10] https://optuna.org/
[11] https://github.com/openai/openai-python

3.2.2 Refinamento e Limpeza do Conteúdo

Na segunda etapa, esse conteúdo foi submetido a um processo de refinamento manual realizado pelo autor, que consistiu em:

- Remoção de trechos repetitivos, slogans e chamadas publicitárias
- Reescrita manual de seções ambíguas, garantindo coerência e objetividade, com foco na remoção de frases com apelo comercial/marketing e textos repetidos
- Organização lógica dos tópicos por categorias temáticas

O processo de refinamento foi validado por meio de revisão realizada pelo autor e um colaborador voluntário, garantindo a qualidade e objetividade do conteúdo final.

A Figura 4 ilustra esse processo de refinamento aplicado ao conteúdo textual bruto, evidenciando sua transformação em dados relevantes e estruturados para posterior uso em modelagem de perguntas e respostas.

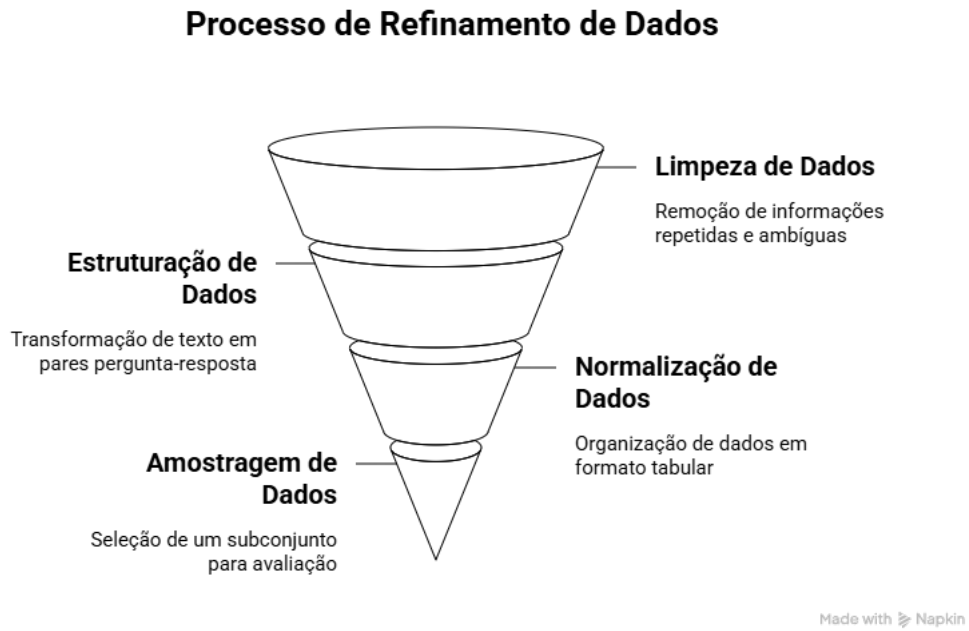

Figura 4 – Etapas de ingestão, tratamento e estruturação dos dados institucionais para alimentar o chatbot com base em conteúdo interno.

3.2.3 Geração do Conjunto FAQ

Na última etapa do *pipeline*, foi criado um agente baseado em `GPT` com o objetivo de transformar o conteúdo filtrado do documento `PDF` em pares de perguntas e respostas (`FAQs`). Esse agente operou com comandos otimizados para gerar entradas representativas

e diversas, mantendo fidelidade ao material fonte. O resultado foi um conjunto de 671 pares de perguntas e respostas organizados em um arquivo `Excel` denominado `faq.xlsx`, com as seguintes colunas:

- `id_pergunta`
- `pergunta`
- `resposta`

Este conjunto passou a representar a base de conhecimento comum utilizada pelas três abordagens testadas neste estudo (Uso Direto via `API`, `RAG` e `Fine-Tuning`).

3.2.4 Avaliação Experimental

Para a etapa de avaliação, foi selecionado aleatoriamente um subconjunto com 100 perguntas do `faq.xlsx`. Cada uma dessas perguntas foi submetida cinco vezes a cada abordagem, totalizando 1.500 interações (100 perguntas $\times$ 5 repetições $\times$ 3 modelos). Essa abordagem de múltiplas interações por pergunta permitiu avaliar não apenas a qualidade média das respostas, mas também a consistência e a variabilidade do desempenho de cada modelo ao longo do tempo e em diferentes execuções. A garantia de que todos os modelos utilizassem o mesmo conjunto de perguntas de teste foi fundamental para assegurar uma comparação justa e equitativa entre as abordagens avaliadas.

Os resultados gerados foram organizados em um arquivo `Excel` denominado `avaliacao_respostas.xlsx`, contendo colunas para: `id_pergunta`, `pergunta`, `resposta_esperada`, `resposta_modelo`, `tempo_resposta` e `identificador_modelo`.

3.3 Implementação das Abordagens

Para a análise comparativa, três abordagens distintas de implementação de chatbots inteligentes foram cuidadosamente configuradas e testadas. Cada implementação foi projetada para refletir um cenário de uso real, utilizando as melhores práticas e ferramentas disponíveis para cada metodologia. O objetivo foi garantir que as comparações fossem justas e que os resultados pudessem ser generalizados para contextos empresariais semelhantes. A seguir, detalhamos a configuração e os aspectos técnicos de cada uma das abordagens.

3.3.1 Uso Direto de LLM via API (API)

A abordagem de Uso Direto de `LLM` via `API` foi implementada utilizando a plataforma `ChatGPT` da `OpenAI`, especificamente o modelo `gpt-4o-mini`. A escolha deste modelo se deu por ser a versão mais popular no momento dos testes, representando uma opção

amplamente utilizada no mercado. A premissa central desta abordagem é que o `LLM`, por si só, é capaz de compreender e gerar respostas a partir de um vasto conhecimento pré-treinado, necessitando apenas de um contexto adicional para operar dentro do domínio específico da empresa. Em vez de realizar `fine-tuning` ou empregar técnicas de `RAG` complexas, todo o conteúdo do arquivo `faq.xlsx` foi pré-carregado como base de conhecimento estática diretamente na plataforma da `OpenAI`.

Figura 5 – Arquitetura da abordagem com LLM via API direta. *Nesta estratégia, o conteúdo do FAQ é embutido diretamente no assistente da OpenAI por meio da funcionalidade File Search, permitindo respostas contextuais sem fine-tuning ou banco vetorial.*

O processo de carregamento da base de conhecimento seguiu os seguintes passos:

1. **Leitura do Excel:** O arquivo `faq.xlsx` foi lido utilizando a biblioteca `pandas` (`pd.read_excel("faq.xlsx")`)
2. **Transformação para JSON:** Os registros do `DataFrame` foram transformados em um formato `JSON` (`df.to_json("faq.json", orient="records", force_ascii=False)`), um formato de dados leve e amplamente utilizado para intercâmbio de dados, facilitando a importação para a plataforma da `OpenAI`
3. **Importação para o Assistente OpenAI:** No painel de `Assistants` da `OpenAI`, o arquivo `faq.json` foi importado na funcionalidade de `File Search`. A `OpenAI` indexa esse `JSON` diretamente, tornando cada pergunta e resposta disponíveis ao assistente sem a necessidade de ajustes adicionais no modelo ou de um processo de `fine-tuning` complexo. Essa funcionalidade permite que o modelo acesse e utilize as informações do `FAQ` de forma eficiente durante a geração de respostas

Para garantir que o `AndréIA_Assistente` (nome dado ao assistente na plataforma) respondesse estritamente com as informações do `FAQ` e mantivesse o comportamento desejado, foi elaborado um *prompt* detalhado no parâmetro `System instructions`. Este *prompt* funcionou também como um *guard rail*, definindo o papel do assistente e estabelecendo um comportamento de *fallback* para perguntas fora do domínio.

O *prompt* completo utilizado encontra-se no Apêndice A.

Na engenharia de *prompt*, foram combinadas diversas técnicas para maximizar a precisão e a aderência ao escopo. Utilizou-se o ***zero-shot prompting***, **delimitadores** e uma estratégia de recusa clara quando o questionamento extrapolava o escopo do `FAQ`. Adicionalmente, os hiperparâmetros de geração do modelo foram ajustados para maximizar a precisão e a previsibilidade das respostas: a `temperatura` foi definida em `0.02`, o `top_p` em `0.85` (parâmetro que controla a diversidade das respostas através de *nucleus sampling*, onde valores menores resultam em respostas mais focadas e determinísticas), e a saída foi configurada para ser sempre em texto simples, sem formatação adicional.

O acesso a este assistente, para fins de experimentação e coleta de dados, ocorreu via `API`, utilizando a biblioteca `openai` em `Python`. Os parâmetros configurados para o assistente foram:

- **Nome:** `AndréIA_Assistente`
- **Modelo:** `gpt-4o-mini`
- **Base de conhecimento:** Conteúdo do `faq.xlsx` convertido para `JSON` e importado via `File Search`
- **Formato de saída:** Texto simples
- **Temperatura:** `0.02`
- **Top P:** `0.85`

Esta implementação representa a abordagem mais direta e de menor complexidade em termos de desenvolvimento, aproveitando a inteligência pré-treinada do `LLM` e a infraestrutura da `OpenAI` para gerenciar a base de conhecimento. No entanto, como será discutido nos resultados, essa simplicidade pode vir acompanhada de *trade-offs* em termos de latência e custo, especialmente em cenários de grande volume de dados ou de interações complexas.

3.3.2 Retrieval-Augmented Generation (RAG)

A abordagem de `Retrieval-Augmented Generation` (`RAG`) foi implementada com o objetivo de combinar a robustez dos *Large Language Models* (`LLMs`) com a capacidade de recuperação de informações específicas de uma base de conhecimento, visando aprimorar a precisão e a relevância das respostas, ao mesmo tempo em que se busca otimizar o uso de recursos e mitigar as alucinações. Nesta etapa, reutilizamos o arquivo `faq.xlsx` como base de conhecimento e implementamos um *pipeline* `RAG` em `Python`, detalhado no notebook `api_rag_gera_respostas.ipynb`.

Figura 6 – Arquitetura da abordagem Retrieval-Augmented Generation (RAG). *Neste fluxo, a pergunta é convertida em embedding, comparada com um índice FAISS e os trechos mais relevantes do FAQ são recuperados para compor o prompt enviado ao LLM.*

O *pipeline* `RAG` foi construído com os seguintes componentes e etapas:

1. **Carregamento e Processamento da Base de Conhecimento:**

   - Inicialmente, o arquivo `faq.xlsx` foi carregado por meio da biblioteca `pandas`. Cada par pergunta-resposta foi tratado como um documento individual, ou seja, cada linha do `Excel` foi convertida em um objeto passível de indexação. Isso envolveu a concatenação da pergunta e da resposta em um único campo de texto para formar o conteúdo que seria vetorizado
   - Para a criação dos *embeddings* (representações vetoriais densas do texto), empregou-se a classe `OpenAIEmbeddings` da biblioteca `langchain`, que utiliza a `API` da `OpenAI` para gerar esses vetores. Os *embeddings* são cruciais para a busca semântica, pois permitem que textos com significados semelhantes sejam representados por vetores próximos no espaço multidimensional

2. **Construção do Índice Vetorial:**

   - Após a geração dos *embeddings* para todos os documentos do `FAQ`, foi construído um índice `FAISS` (`FAISS.from_documents`). `FAISS` (*Facebook AI Similarity Search*) é uma biblioteca para busca eficiente de similaridade e agrupamento de vetores densos. A escolha do `FAISS` se deu pela sua eficiência em realizar buscas por similaridade de cosseno em grandes volumes de vetores, o que é fundamental para a etapa de recuperação do `RAG`
   - O índice `FAISS` permite que, dada uma nova pergunta, o sistema encontre rapidamente os documentos mais relevantes na base de conhecimento, mesmo que a pergunta não contenha as mesmas palavras-chave exatas dos documentos, mas sim um significado similar

3. **Processo de Geração de Respostas com RAG:**

    - Durante a execução do experimento, cada pergunta de teste (das 100 perguntas selecionadas para avaliação) foi transformada em um *embedding* utilizando o mesmo modelo de *embedding* da `OpenAI`
    - Este *embedding* da pergunta foi então submetido ao *retriever* `FAISS`, que retornou os trechos mais relevantes do `FAQ`. A quantidade de trechos recuperados (*top-k*) é um parâmetro configurável que impacta a qualidade e a latência da resposta. No contexto deste estudo, foram recuperados os trechos mais relevantes para garantir a contextualização adequada
    - Os trechos recuperados foram concatenados às instruções de sistema (*prompt*) e à pergunta original, formando o *prompt* final que foi enviado ao modelo `gpt-4o-mini` via a biblioteca `openai`. A estrutura do *prompt* é crucial para instruir o `LLM` a utilizar o contexto fornecido para gerar a resposta, em vez de depender apenas de seu conhecimento pré-treinado
    - Os hiperparâmetros de geração (`temperatura = 0.02`; `top_p = 0.85`) e o formato de saída em texto simples foram mantidos idênticos aos empregados na abordagem de Uso Direto de `LLM` via `API`. Essa consistência nos parâmetros de geração é importante para isolar o efeito da técnica `RAG` no desempenho do chatbot
    - Por fim, as respostas retornadas pela `API` foram registradas em um `DataFrame` do `pandas` e exportadas para o arquivo `respostas_rag_assistant.xlsx` por meio da função `utils.Utils.salvar_dataframe_em_excel`, observando o mesmo esquema de colunas (`id_pergunta`, `pergunta`, `resposta_esperada`, `resposta_modelo`, `tempo_resposta` e `identificador_modelo`) do teste anterior. Isso garantiu a padronização dos dados para a etapa de avaliação

As instruções de sistema utilizadas para compor o *prompt* foram adaptadas especificamente para esta abordagem, com ênfase no uso do contexto externo recuperado. O conteúdo completo do *prompt* encontra-se no Apêndice B.

A implementação do `RAG` representa um meio-termo entre a simplicidade da `API` direta e a complexidade do `fine-tuning`. Ela busca o melhor dos dois mundos: a capacidade de geração de linguagem dos `LLMs` e a precisão factual de uma base de conhecimento externa. A eficácia dessa abordagem, como será demonstrado nos resultados, depende da qualidade do modelo de *embedding*, da eficiência do banco de dados vetorial e da capacidade do `LLM` de sintetizar as informações recuperadas de forma coerente e completa.

3.3.3 Fine-Tuning de LLM (Mistral)

A abordagem de `fine-tuning` de um *Large Language Model* (`LLM`) foi implementada com o objetivo de especializar um modelo de linguagem pré-treinado para o domínio específico do atendimento ao cliente da empresa fictícia, utilizando o conjunto de dados do `faq.xlsx`. Esta técnica busca adaptar os pesos internos do modelo para que ele gere respostas mais precisas, relevantes e alinhadas com o estilo e a terminologia da empresa, potencialmente superando as abordagens genéricas em termos de qualidade e personalização. Para esta implementação, foi escolhido o modelo `rhaymison/Mistral-portuguese-luana-7b-chat`, uma versão em português do `Mistral 7B`, um modelo de código aberto conhecido por sua eficiência e desempenho. A escolha de um modelo em português foi estratégica para garantir a fluidez e a naturalidade das respostas no idioma do público-alvo.

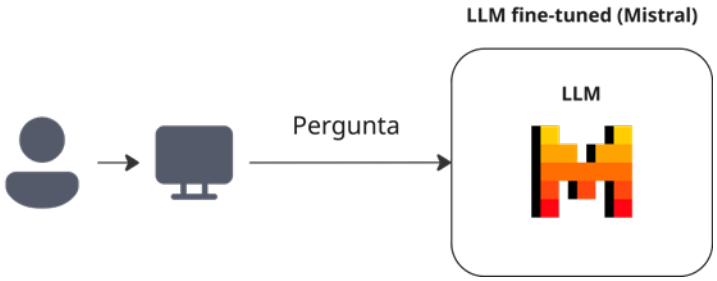

Figura 7 – Arquitetura da abordagem com LLM fine-tuned (Mistral). *Nesta estratégia, o modelo Mistral é ajustado com os dados do FAQ, incorporando diretamente o conhecimento nos pesos do modelo. A geração de resposta não depende de engenharia de prompt nem de contexto externo.*

O *pipeline* de `fine-tuning`, implementado nos notebooks `fine_tuning_lora_-optuna.v3.ipynb` e `mistral_treinamento.v2.ipynb`, seguiu as seguintes etapas:

1. **Importação e Preparação do Modelo:**
   - O modelo base foi importado utilizando a biblioteca `unsloth` com o comando: [language=Python] unsloth.FastLanguageModel.from$_p retrained(load_i n_4 bit = True, max_s eq_l ength = 2048)$
   - A opção `load_in_4bit=True` foi utilizada para carregar o modelo em 4 bits, uma técnica de quantização que reduz significativamente o consumo de memória da `GPU`, tornando o `fine-tuning` de modelos grandes mais acessível em ambientes com recursos limitados, como o `Google Colab`. A `max_seq_length` foi definida em 2048 tokens para acomodar o tamanho das perguntas e respostas do `FAQ`
   - Para otimizar ainda mais o processo de treinamento, foi aplicada a técnica de `LoRA` (*Low-Rank Adaptation*) por meio da função:

[language=Python] FastLanguageModel.get$_peft_model(model, r = 4, target_modules = [$"$q_proj$"$, $"$k_proj$"$, $"$v_proj$"$, $"$o_proj$"$], lora_alpha = 16, lora_dropout = 0.1)$

- O `LoRA` é uma técnica de *parameter-efficient fine-tuning* (`PEFT`) que congela os pesos do modelo pré-treinado e injeta matrizes de baixo ranque (adaptadores) em algumas camadas do modelo. Apenas esses adaptadores são treinados, o que reduz drasticamente o número de parâmetros treináveis e, consequentemente, o custo de memória e computação, sem sacrificar significativamente o desempenho. Esta abordagem torna viável o *fine-tuning* em ambientes com recursos limitados como o `Google Colab`

2. **Preparação dos Dados de Treinamento:**

   - O mesmo conjunto de dados do `faq.xlsx` foi utilizado, mas convertido para um formato conversacional, onde cada entrada era precedida por `user:` e `assistant:`, formato ideal para treinamento de modelos de chat
   - O conjunto de dados foi dividido em treino e validação na proporção de 80/20 por meio de `dataset.train_test_split`, permitindo avaliar o modelo em dados distintos daqueles usados no treinamento

3. **Otimização de Hiperparâmetros com Optuna:**

   - A biblioteca `Optuna` foi utilizada para encontrar os melhores hiperparâmetros. Os valores testados foram:
     - `batch_size`: entre 80 e 128 (passo de 8)
     - `learning_rate`: logaritmicamente entre 1×10^{-4} e 3×10^{-4}
     - `max_steps`: de 20 a 200 (passo de 20)
     - `warmup_steps`: de 10 a 50 (passo de 10)
     - `gradient_accumulation_steps`: valores em {2, 4, 8}
     - `weight_decay`: de 0,0 a 0,05 (passo de 0,01)
   - O melhor conjunto de hiperparâmetros encontrado pelo `Optuna` foi:
     - `batch_size`: 128
     - `learning_rate`: 0.0002244565286643375
     - `max_steps`: 200
     - `warmup_steps`: 50
     - `gradient_accumulation_steps`: 4
     - `weight_decay`: 0.04
     - `lora_r`: 16
     - `lora_dropout`: 0.05

  - `optim`: `adamw_8bit`
- O melhor valor de perda (*loss*) obtido foi 0.1056, demonstrando uma otimização eficaz dos hiperparâmetros
- Também foram utilizadas as opções `auto_find_batch_size=True` e `torch_compile(backend="inductor")` para maximizar a eficiência
- O processo foi conduzido com o `SFTTrainer` da biblioteca `TRL`, utilizando *validation loss* como métrica de avaliação

4. **Geração de Respostas com o Modelo Fine-Tuned:**
   - Após o `fine-tuning`, os pesos `LoRA` foram mesclados ao modelo base
   - As respostas foram geradas com o notebook `mistral_gera_respostas.ipynb` e a classe `RespostaGenerator`
   - Foram mantidos os mesmos hiperparâmetros das abordagens anteriores (`temperatura = 0.02`, `top_p = 0.85`)
   - As respostas foram armazenadas em `respostas_mistral.xlsx`, com colunas: `id_pergunta`, `pergunta`, `resposta_esperada`, `resposta_modelo`, `tempo_resposta`, `identificador_modelo`

Esta implementação de `fine-tuning` representa a abordagem mais complexa e intensiva em recursos das três avaliadas. Ela busca alcançar o mais alto nível de personalização e precisão, mas, como será discutido nos resultados, sua eficácia é altamente dependente da qualidade e quantidade dos dados de treinamento, o que pode ser um desafio em cenários de dados restritos.

3.4 Métricas de Avaliação

Para comparar objetivamente o desempenho das três abordagens de implementação de chatbots, foi desenvolvida e utilizada a classe `RespostaEvaluator`, definida no script `avaliador_respostas.py` e explorada no notebook `avaliador.ipynb`. O processo de avaliação foi padronizado e automatizado para garantir a consistência e a replicabilidade dos resultados, e consistiu nas etapas descritas a seguir.

3.4.1 Agregação de Resultados

O primeiro passo da avaliação foi a agregação dos resultados gerados por cada uma das três abordagens. Cada abordagem foi submetida ao mesmo conjunto de 100 perguntas de teste, com 5 repetições por pergunta, totalizando 1.500 respostas geradas. As respostas, juntamente com outras informações relevantes, foram registradas pela classe `RespostaGenerator` em um `DataFrame` do `pandas` com as seguintes colunas:

- `interacao`: O número da interação (de 1 a 5), permitindo a análise da consistência do modelo
- `pergunta_id`: O identificador único da pergunta, para rastreamento
- `modelo`: O nome da abordagem utilizada (e.g., `api_direta`, `rag` ou `mistral`)
- `pergunta`: O texto da pergunta submetida ao modelo
- `resposta_esperada`: O texto da resposta de referência, extraído do `faq.xlsx`
- `resposta`: O texto da resposta gerada pelo chatbot
- `tempo_resposta`: A latência medida em segundos para a geração da resposta

Este `DataFrame` consolidado serviu como a base para todas as análises subsequentes, permitindo uma comparação direta e sistemática entre as três técnicas.

3.4.2 SBERT Score (Similaridade Semântica)

Para avaliar a similaridade semântica entre as respostas geradas pelos modelos e as respostas de referência, foi empregado o modelo `sentence-transformers/paraphrase-multilingual-mpnet-base-v2`. Este modelo, baseado na arquitetura `Sentence-BERT` (`SBERT`), é especializado em gerar *embeddings* de sentenças que capturam seu significado semântico. A avaliação foi realizada da seguinte forma:

1. Cada par de textos (`resposta`, `resposta_esperada`) foi convertido em *embeddings* (vetores densos)
2. A similaridade de cosseno entre os dois *embeddings* foi calculada. A similaridade de cosseno mede o ângulo entre dois vetores, resultando em um valor entre -1 e 1 (ou 0 e 1, se os vetores forem normalizados). Um valor próximo de 1 indica que os textos são semanticamente muito similares, enquanto um valor próximo de 0 indica baixa similaridade
3. O valor resultante foi armazenado em uma nova coluna do `DataFrame`, denominada `sbert_score`, indicando o grau de alinhamento semântico da resposta gerada com a resposta de referência

Esta métrica é fundamental para avaliar se o chatbot está fornecendo respostas que, embora não sejam idênticas em termos de palavras, transmitem a mesma informação que a resposta esperada.

3.4.3 BERTScore (Qualidade Textual)

Complementarmente à avaliação de similaridade semântica, foi utilizada a métrica `BERTScore` para avaliar a qualidade textual das respostas em um nível mais granular. O `BERTScore` compara a sobreposição de subpalavras (*tokens*) entre a resposta gerada e a resposta de referência, utilizando os *embeddings* de um modelo `BERT` (neste caso, o `xlm-roberta-base`, um modelo multilíngue robusto e amplamente utilizado para tarefas de compreensão de linguagem, o que o torna adequado para a avaliação de qualidade textual em português devido à sua capacidade multilíngue e robustez para tarefas de compreensão de linguagem). A avaliação foi realizada utilizando a função `bert_score.score()`, que gera três métricas para cada par de textos:

- `bert_precision`: Mede a proporção de *tokens* na resposta gerada que também estão presentes na resposta de referência
- `bert_recall`: Mede a proporção de *tokens* na resposta de referência que também estão presentes na resposta gerada
- `bert_f1`: A média harmônica entre a precisão e o *recall*, fornecendo uma única métrica que equilibra os dois aspectos

Essas métricas oferecem uma visão detalhada da qualidade da resposta, considerando a sobreposição de palavras e a fluidez do texto, e complementam a avaliação de similaridade semântica do `SBERT Score`.

3.4.4 Medição de Latência

A eficiência temporal de cada abordagem foi avaliada medindo-se a latência de ponta a ponta para a geração de cada resposta. A latência foi capturada diretamente na classe `RespostaGenerator`, que mede o tempo decorrido entre a chamada à `API` (ou à função de geração local, no caso do modelo *fine-tuned*) e o retorno da resposta. Esse valor, em segundos, foi salvo na coluna `tempo_resposta` do `DataFrame` de resultados. A análise da latência é crucial para avaliar a viabilidade de cada abordagem em cenários de atendimento em tempo real, no qual respostas rápidas são essenciais para a experiência do usuário.

3.4.5 Avaliação Consolidada e Exportação dos Dados

Com todos os resultados registrados no `DataFrame`, a classe `RespostaEvaluator` aplicou o método `avaliar_dataframe(df)`, que iterou sobre cada linha do `DataFrame`, chamou internamente o método `avaliar_resposta(resposta, resposta_esperada, tempo_resposta)` e anexou as colunas de avaliação (`sbert_score`, `bert_precision`, `bert_recall`, `bert_f1`) ao `DataFrame`.

Por fim, o `DataFrame` enriquecido com todas as métricas de avaliação foi exportado para o arquivo `avaliacao_respostas.xlsx` por meio da função `utils.Utils.salvar_dataframe_em_excel`, preservando a estrutura original das colunas e adicionando as avaliações geradas. Este fluxo metodológico garantiu que as três técnicas fossem comparadas sob critérios uniformes de fidelidade semântica, qualidade textual e eficiência de tempo, preparando o terreno para a análise dos resultados.

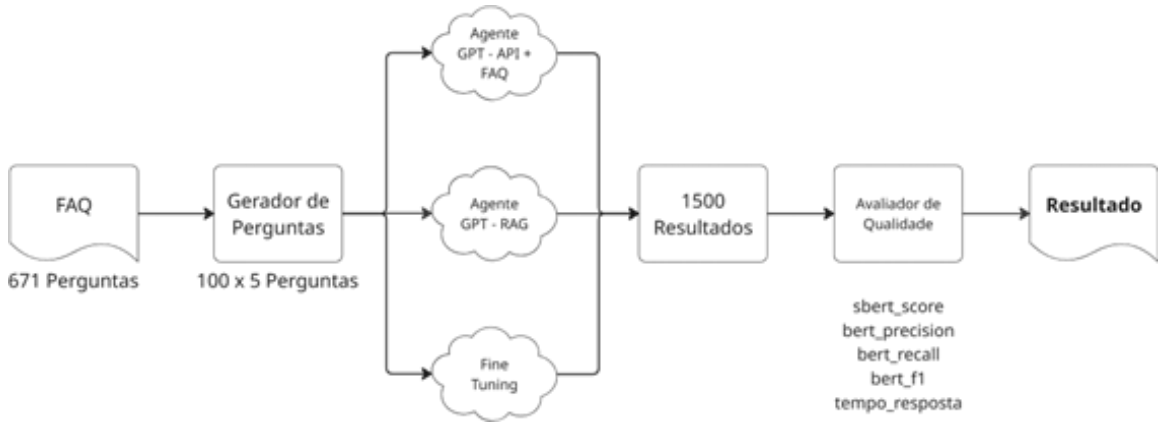

Figura 8 – Fluxo completo do experimento comparativo. *O conjunto de perguntas geradas a partir do FAQ é utilizado como base para testar três abordagens distintas de LLM. As respostas geradas são avaliadas por métricas automatizadas, consolidando um total de 1500 interações para análise.*

3.5 Procedimento Experimental

O procedimento experimental foi cuidadosamente desenhado para garantir a consistência, a replicabilidade e a validade dos resultados da análise comparativa entre as três abordagens de implementação de chatbots. O fluxo de trabalho foi dividido em quatro etapas principais, cada uma com seus objetivos específicos, desde a preparação dos dados até a análise final dos resultados. A seguir, detalhamos cada um dos passos do procedimento experimental.

3.5.1 Preparação

- **Carregamento do Dataset de FAQs:** O primeiro passo foi o carregamento do arquivo `faq.xlsx`, que continha a base de conhecimento da empresa fictícia. Este arquivo foi lido e processado para ser utilizado nas diferentes abordagens

- **Seleção do Conjunto de Teste:** Foi selecionado aleatoriamente um subconjunto de 100 perguntas e respostas do `faq.xlsx` para servir como conjunto de teste. Este conjunto foi utilizado para avaliar o desempenho de todos os modelos, garantindo uma base de comparação consistente

3.5.2 Geração de Respostas

- **Configuração dos Modelos/Pipelines:** Para cada uma das três abordagens (Uso Direto de `LLM` via `API`, `RAG` e `Fine-Tuning` de `Mistral`), o respectivo modelo ou *pipeline* foi configurado conforme descrito nas seções anteriores deste capítulo
- **Execução da Geração de Respostas:** A classe `RespostaGenerator` foi utilizada para submeter o conjunto de 100 perguntas de teste a cada um dos modelos
- **Realização de Repetições:** Para cada pergunta, foram realizadas 5 repetições (`repeticoes=5`), permitindo avaliar variabilidade e consistência
- **Coleta de Dados:** Durante a geração de respostas, foram coletados os textos das respostas geradas e os tempos de resposta (latência) para cada interação

3.5.3 Avaliação

- **Cálculo das Métricas de Qualidade:** A classe `RespostaEvaluator` foi utilizada para calcular as métricas de qualidade para cada resposta gerada
- **Consolidação dos Dados de Avaliação:** As métricas calculadas foram adicionadas ao `DataFrame` de resultados

3.5.4 Análise

- **Consolidação Final dos Resultados:** Todos os resultados foram consolidados em um único `DataFrame` e salvos no arquivo `avaliacao_respostas.xlsx`
- **Geração de Visualizações:** Foram geradas diversas visualizações consolidadas no arquivo `todos_graficos_completos.pdf`
- **Análise Comparativa:** A análise final foi realizada com base nos dados consolidados e nas visualizações, considerando desempenho médio, distribuição por pergunta e implicações práticas

Este procedimento experimental rigoroso e sistemático garantiu que a comparação entre as três abordagens fosse justa e baseada em evidências quantitativas, fornecendo uma base sólida para as conclusões e recomendações apresentadas neste trabalho.

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Este capítulo apresenta e discute os resultados da avaliação comparativa das três abordagens de chatbot – Uso Direto de `LLM` via `API` (`API`), `Retrieval-Augmented Generation` (`RAG`) e o modelo `Mistral` – com base nas métricas e no procedimento experimental descritos no Capítulo 3. A análise visa responder aos objetivos do estudo, comparando as abordagens em termos de qualidade das respostas, eficiência temporal, custos de desenvolvimento e viabilidade prática no cenário de dados institucionais limitados.

4.1 Sobre as Interações

No experimento, foram aplicadas 5 vezes a mesma pergunta para cada modelo com o objetivo de medir a estabilidade de cada uma das implementações. Como resultado, nota-se que nenhum dos modelos teve um desvio significativo nas interações, a exemplo dos gráficos das figuras 9, 10 e 11.

Figura 9 – Distribuição das pontuações obtidas nas interações com o modelo de linguagem acessado diretamente via API, ao longo das 5 repetições da mesma pergunta.

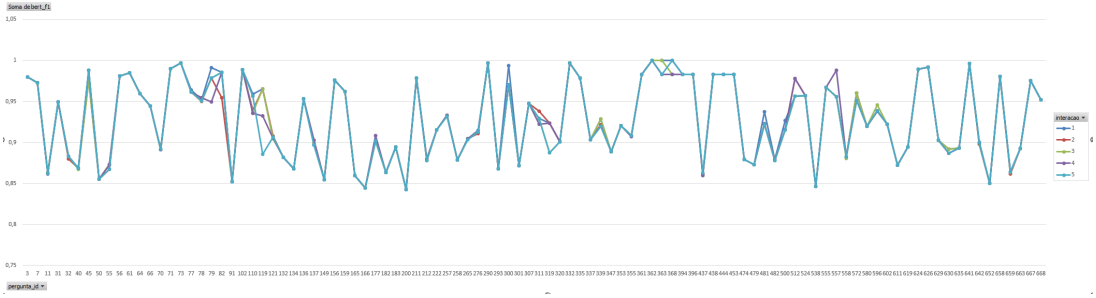

Figura 10 – Distribuição das pontuações das respostas geradas utilizando o método de geração aumentada por recuperação (RAG), indicando consistência entre repetições.

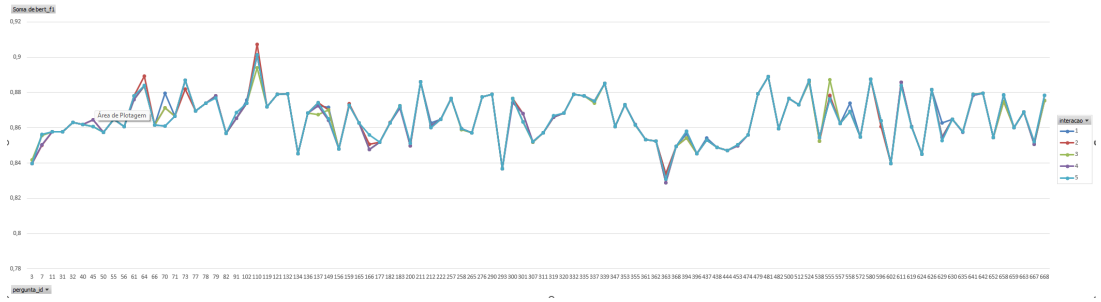

Figura 11 – Comportamento das respostas fornecidas pelo modelo Mistral ajustado por *fine-tuning*, demonstrando estabilidade nas cinco execuções por pergunta.

Com base na característica observada, o estudo seguirá utilizando a mediana das interações, que conforme as figuras 12, 13 e 14 possuem valores bem próximos dos anteriormente observados.

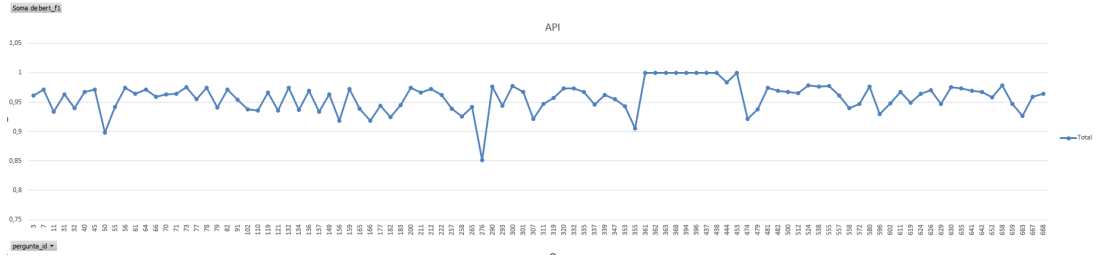

Figura 12 – Valores medianos das pontuações de qualidade (BERT F1) nas interações com a API, evidenciando padrão de desempenho.

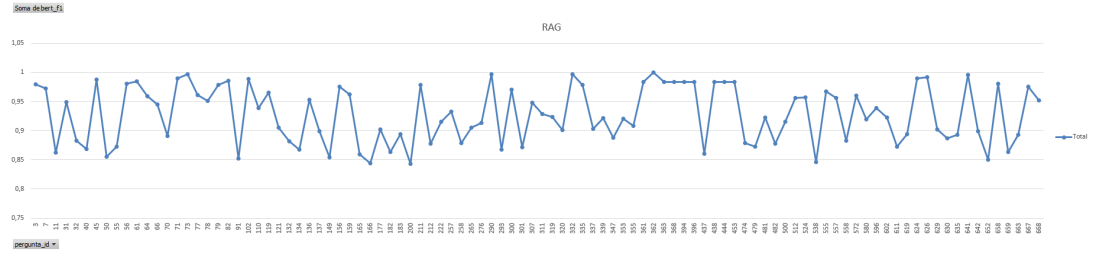

Figura 13 – Pontuação mediana das respostas do modelo com recuperação, reforçando a proximidade dos resultados ao longo das execuções.

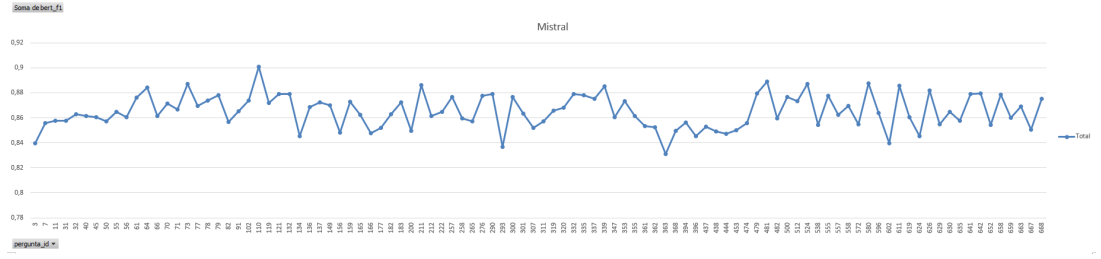

Figura 14 – Mediana das métricas de desempenho do modelo Mistral, confirmando a consistência das respostas ajustadas via *fine-tuning*.

4.2 Tempo de Resposta

A avaliação do tempo de resposta foi conduzida com o objetivo de mensurar a eficiência temporal de cada uma das abordagens implementadas. Esse fator é particularmente relevante em contextos de atendimento ao cliente, nos quais a agilidade na entrega da resposta influencia diretamente a experiência do usuário.

Tabela 1 – Estatísticas Descritivas do Tempo de Resposta (em segundos) por Modelo

Modelo	Média	Desvio Padrão	Mínimo	Máximo
API	10,30	4,09	5,37	33,97
RAG	6,68	1,43	4,86	13,58
Mistral	6,14	0,03	6,05	6,25

Conforme ilustrado na Tabela 1, o modelo `API` apresentou a maior média de tempo de resposta, com 10,30 segundos, além de uma dispersão significativamente maior (desvio padrão de 4,09 segundos). O tempo de resposta oscilou entre 5,37 e 33,97 segundos, indicando variabilidade sensível, possivelmente causada pela latência de rede e pelo *overhead* de comunicação com a infraestrutura da `OpenAI`.

Por outro lado, a abordagem `RAG` demonstrou desempenho intermediário, com tempo médio de 6,68 segundos e menor variabilidade (desvio padrão de 1,43 segundos). Essa abordagem se beneficia da geração de contexto local via `FAISS`, o que pode explicar a redução na latência em relação à `API` pura, ainda que mantenha dependência da `API` da `OpenAI` para a geração final da resposta.

A implementação com *fine-tuning* do modelo `Mistral` apresentou o melhor desempenho temporal entre todas as abordagens. O tempo médio de resposta foi de 6,14 segundos, com variação mínima entre as interações (desvio padrão de apenas 0,03 segundos). Esta consistência reflete o fato de que o modelo é executado localmente (no ambiente do `Google Colab`) e não depende de requisições externas para processar as respostas, o que elimina gargalos de rede e permite um tempo de resposta mais estável.

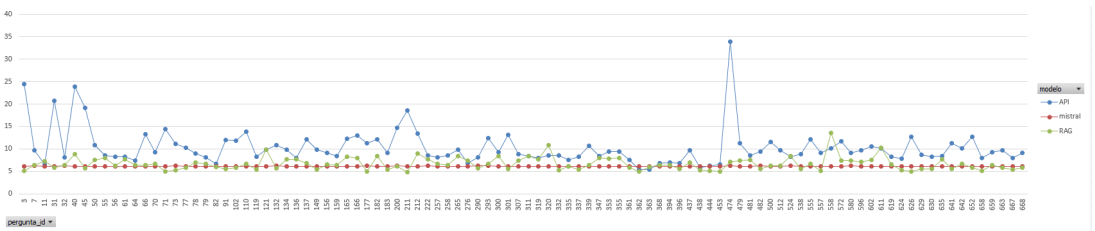

Figura 15 – Gráfico comparativo dos tempos de resposta por modelo

Dessa forma, a análise dos dados evidencia que, do ponto de vista de eficiência temporal, a abordagem com *fine-tuning* do `Mistral` é a mais adequada para aplicações

que exigem respostas rápidas e consistentes. A abordagem RAG também se mostra viável, com desempenho competitivo, enquanto a solução via API direta, apesar de mais simples de implementar, apresenta maior latência e variação, podendo impactar negativamente a experiência do usuário em cenários com alto volume de interações.

4.3 Análise de Custos e Tempo de Desenvolvimento

Uma dimensão fundamental para a viabilidade prática das abordagens analisadas refere-se aos custos e tempo necessários para desenvolvimento e implementação. Esta análise considera tanto os custos diretos de desenvolvimento quanto os custos operacionais associados a cada estratégia, fornecendo uma perspectiva econômica essencial para a tomada de decisão empresarial.

4.3.1 Custos de Preparação de Dados

Todas as abordagens compartilham uma etapa inicial obrigatória de preparação de dados, que envolveu o levantamento de todo material disponível, análise humana do conteúdo para remoção de informações repetidas e desnecessárias, e criação de um *prompt* e programa para gerar um FAQ a partir do documento. Esta etapa demandou um total de 29 horas de trabalho especializado, representando um investimento de R$ 4.350,00 em recursos humanos. Adicionalmente, o processo de geração do FAQ utilizando o GPT-4o Mini resultou em custos de processamento de R$ 42,47, considerando a cotação do dólar de R$ 5,55 em julho de 2025. O custo total da preparação de dados, portanto, totaliza R$ 4.392,47 e deve ser considerado em qualquer implementação, independentemente da abordagem escolhida.

4.3.2 Custos Específicos por Abordagem

A implementação da abordagem de uso direto de LLM via API demonstrou ser a mais eficiente em termos de tempo de desenvolvimento específico, requerendo 24 horas para codificação, testes e medições, o que representa um investimento de R$ 3.600,00 em desenvolvimento, acrescido de R$ 50,00 em custos de infraestrutura computacional no Google Colab. Considerando os custos de preparação de dados, o investimento total para esta abordagem totaliza R$ 8.042,47. Esta eficiência pode ser atribuída à simplicidade da integração direta com serviços pré-existentes, eliminando a necessidade de desenvolvimento de componentes complexos de recuperação ou treinamento de modelos.

A implementação da abordagem RAG apresentou maior complexidade, demandando 40 horas de desenvolvimento específico, o que resultou em custos de R$ 6.000,00 para codificação, testes e medições, além de R$ 50,00 em infraestrutura computacional. Incluindo os custos de preparação de dados, o investimento total para a abordagem RAG alcança R$ 10.442,47. O tempo adicional necessário reflete a complexidade inerente à implementação de

sistemas de recuperação semântica, incluindo a configuração de índices vetoriais, otimização de parâmetros de busca e integração entre componentes de recuperação e geração.

O *fine-tuning* do modelo `Mistral` representou o maior investimento em tempo e recursos, totalizando 72 horas distribuídas entre desenvolvimento de código (40 horas), processo de treinamento (24 horas) e testes e medições (8 horas). Os custos específicos desta implementação somaram R$ 11.050,00, incluindo R$ 250,00 em infraestrutura computacional especializada. Considerando os custos de preparação de dados, o investimento total para o *fine-tuning* atinge R$ 15.442,47. Este investimento substancial justifica-se pela necessidade de configuração cuidadosa de hiperparâmetros, implementação de técnicas de `PEFT` como `LoRA`, monitoramento contínuo do processo de treinamento e expertise específica em otimização de modelos de linguagem.

4.3.3 Análise Comparativa de Viabilidade Econômica

A análise dos custos totais de desenvolvimento revela um gradiente claro de complexidade e investimento necessário. A abordagem via `API` emerge como a opção mais acessível economicamente, com investimento total de R$ 8.042,47, oferecendo uma relação custo-benefício atrativa para organizações com recursos limitados. A abordagem `RAG` posiciona-se como uma alternativa intermediária, com investimento de R$ 10.442,47, oferecendo maior controle sobre a infraestrutura com custo moderado. O *fine-tuning*, com investimento de R$ 15.442,47, representa a opção mais custosa, justificando-se apenas em cenários onde a autonomia tecnológica e a personalização extrema são prioritárias e há disponibilidade de recursos substanciais.

É importante considerar que estes custos representam apenas o investimento inicial de desenvolvimento. Custos operacionais contínuos, como taxas de `API`, manutenção de infraestrutura e atualizações de modelos, devem ser avaliados em análises de viabilidade de longo prazo, particularmente para implementações em escala comercial.

4.4 Resultados Quantitativos: Métricas de Qualidade

Além da análise de tempo de resposta e custos discutida anteriormente, esta seção apresenta os resultados quantitativos relativos à qualidade das respostas geradas por cada uma das três abordagens. As métricas consideradas foram: `SBERT Score` (similaridade semântica), `BERT Precision`, `BERT Recall` e `BERT F1`. Os resultados foram extraídos da planilha `resultados_mediana.xlsx` e estão sumarizados na Tabela 2, contendo média, desvio padrão, mínimo e máximo para cada métrica por modelo.

Tabela 2 – Estatísticas Descritivas das Métricas de Qualidade por Modelo

Modelo	Métrica	Média	Desvio Padrão	Mínimo	Máximo
API	SBERT Score	0.8911	0.0795	0.3442	0.9892
	BERT Precision	0.9334	0.0308	0.7129	0.9937
	BERT Recall	0.9844	0.0164	0.8802	0.9996
	BERT F1	0.9575	0.0232	0.7961	0.9944
RAG	SBERT Score	0.7729	0.1605	0.1728	0.9786
	BERT Precision	0.9139	0.0512	0.7109	0.9897
	BERT Recall	0.9413	0.0476	0.6907	0.9981
	BERT F1	0.9275	0.0472	0.7057	0.9933
Mistral	SBERT Score	0.5914	0.1163	0.2405	0.8501
	BERT Precision	0.8387	0.0191	0.7651	0.8804
	BERT Recall	0.8934	0.0324	0.7815	0.9531
	BERT F1	0.8651	0.0193	0.7853	0.9094

4.5 Discussão dos Resultados

Ao comparar os resultados obtidos nas métricas de qualidade, verifica-se que a abordagem baseada em `API` (`ChatGPT` com base externa via *File Search*) apresentou desempenho superior em todos os indicadores avaliados. Destacam-se especialmente o `SBERT Score` (0.8911) e o `BERT Recall` (0.9844), o que sugere que este modelo foi capaz não apenas de gerar respostas semanticamente próximas das esperadas, mas também de abranger, de forma consistente, o conteúdo-alvo em suas respostas. Este desempenho pode ser atribuído ao alto grau de otimização dos modelos proprietários da `OpenAI`, bem como à estratégia de ingestão direta do conteúdo via `JSON` estruturado, que elimina ambiguidade e garante um contexto rico e relevante.

A abordagem `RAG` também demonstrou desempenho competitivo, com métricas `BERT` próximas às do modelo `API` (`BERT F1` de 0.9275), embora com uma queda mais acentuada no `SBERT Score` (0.7729). Essa diferença pode ser explicada pelo fato de que, no `RAG`, os trechos recuperados da base de conhecimento são concatenados ao *prompt* em tempo de execução. Essa estratégia depende fortemente da relevância e da coerência dos trechos recuperados pelo índice vetorial. Eventuais ruídos no processo de vetorização ou limitações na densidade dos *embeddings* podem prejudicar o alinhamento semântico entre pergunta e resposta, reduzindo o `SBERT Score` mesmo quando a resposta textual está lexicalmente correta.

Por outro lado, o modelo `Mistral`, mesmo apresentando o menor tempo de resposta entre os três modelos, teve desempenho inferior nas métricas de qualidade, especialmente no `SBERT Score` (0.5914). Esta *performance* aquém do esperado pode ser atribuída a fatores como: (i) o volume limitado de dados utilizados no *fine-tuning* (671 pares pergunta-resposta); (ii) a eventual sobrecarga ou subajuste dos hiperparâmetros durante o processo de adaptação; e (iii) o fato de modelos abertos necessitarem de quantidades significativamente

maiores de dados para atingirem níveis de generalização e fluidez similares aos modelos proprietários. O modelo *fine-tuned* respondeu de maneira mais estável, porém menos precisa, indicando que sua capacidade de generalização foi limitada pelo escopo do treinamento.

Quando considerados os aspectos econômicos em conjunto com os resultados de qualidade, emerge um panorama complexo de *trade-offs*. A abordagem via `API`, além de apresentar a melhor qualidade de resposta, também demonstrou ser a mais eficiente economicamente, com custo total de R$ 8.042,47. Esta combinação de alta qualidade e menor custo de desenvolvimento a posiciona como uma opção altamente atrativa para organizações que buscam resultados eficazes com investimento controlado.

A abordagem `RAG`, embora tenha demandado maior investimento (R$ 10.442,47), oferece um equilíbrio interessante entre qualidade de resposta e controle sobre a infraestrutura. O investimento adicional pode justificar-se em cenários onde a autonomia tecnológica e o controle sobre custos operacionais de longo prazo são prioritários.

O *fine-tuning* do `Mistral`, com o maior investimento (R$ 15.442,47), apresentou limitações significativas em qualidade que questionam sua viabilidade em cenários de dados escassos. O alto custo de desenvolvimento, combinado com a *performance* inferior, sugere que esta abordagem pode ser mais adequada para contextos com maior disponibilidade de dados e recursos para otimização contínua.

Esses resultados reforçam a importância de considerar o alinhamento entre os objetivos do projeto, a maturidade da arquitetura utilizada, a disponibilidade de dados de alta qualidade para treinamento e as restrições orçamentárias. Em contextos empresariais com dados institucionais escassos e demanda por alta precisão, a abordagem `API` emerge como a solução mais equilibrada, oferecendo a melhor relação custo-benefício. A abordagem `RAG` mantém-se como alternativa viável para organizações que priorizam maior controle sobre a infraestrutura, enquanto o *fine-tuning* com modelos abertos requer disponibilidade substancial de recursos e dados para alcançar resultados competitivos.

5 CONCLUSÕES

Este estudo teve como objetivo principal avaliar e comparar três abordagens distintas para implementação de chatbots inteligentes voltados ao atendimento ao cliente em empresas com dados institucionais limitados: (i) uso direto de *Large Language Model* (LLM) via API; (ii) Retrieval-Augmented Generation (RAG); e (iii) *Fine-Tuning* de LLMs. A análise comparativa considerou aspectos de qualidade das respostas, eficiência temporal, custos de desenvolvimento, complexidade de implementação e viabilidade prática em cenários empresariais com recursos e dados restritos. A investigação foi conduzida através de uma metodologia experimental rigorosa que permitiu não apenas responder às questões de pesquisa propostas, mas também fornecer insights práticos valiosos para organizações que buscam implementar soluções de automação conversacional.

5.1 Síntese dos Resultados e Cumprimento dos Objetivos

Os resultados obtidos através da metodologia experimental rigorosa, que envolveu a avaliação de 1.500 interações (100 perguntas $\times$ 5 repetições $\times$ 3 modelos) utilizando métricas padronizadas de qualidade semântica e textual, permitiram responder às questões de pesquisa propostas e cumprir integralmente os objetivos específicos estabelecidos. A abordagem metodológica adotada, baseada em métricas objetivas e procedimentos replicáveis, conferiu robustez científica aos achados e possibilitou a formulação de recomendações práticas fundamentadas em evidências empíricas.

5.1.1 Desempenho em Métricas de Qualidade

Em resposta à primeira questão de pesquisa, que investigava qual abordagem oferece melhor qualidade de resposta em cenários de dados limitados, a abordagem de uso direto de LLM via API demonstrou superioridade consistente e estatisticamente significativa em todas as métricas de qualidade avaliadas. Com SBERT Score de 0.8911, BERT Precision de 0.9334, BERT Recall de 0.9844 e BERT F1 de 0.9575, esta abordagem evidenciou capacidade superior de gerar respostas semanticamente alinhadas e lexicalmente precisas em relação às respostas de referência.

Este desempenho excepcional pode ser atribuído a múltiplos fatores convergentes. Primeiramente, o alto grau de otimização dos modelos proprietários da OpenAI, que se beneficiam de treinamento em escala massiva e técnicas avançadas de alinhamento com preferências humanas através de RLHF (*Reinforcement Learning from Human Feedback*). Em segundo lugar, a estratégia de ingestão direta do conteúdo via JSON estruturado elimina ambiguidades inerentes a processos de recuperação e fragmentação de contexto, garantindo que o modelo tenha acesso completo e organizado à base de conhecimento institucional.

Finalmente, a arquitetura *File Search* da `OpenAI` implementa técnicas sofisticadas de recuperação semântica que superam implementações `RAG` convencionais em termos de precisão e relevância.

A abordagem `RAG` apresentou desempenho competitivo e tecnicamente sólido, especialmente nas métricas `BERT` (`BERT F1` de 0.9275), posicionando-se como uma alternativa viável para organizações que priorizam controle sobre a infraestrutura. No entanto, a queda mais acentuada no `SBERT Score` (0.7729) revelou limitações importantes na manutenção do alinhamento semântico, possivelmente decorrentes de ruídos no processo de vetorização, limitações na densidade dos *embeddings* utilizados na recuperação de contexto, ou subotimização dos parâmetros de busca semântica. Estes achados sugerem que implementações `RAG` requerem otimização cuidadosa e expertise técnica específica para alcançar resultados próximos aos obtidos com soluções proprietárias.

O modelo `Mistral` com *fine-tuning*, apesar de representar a abordagem mais personalizada e tecnicamente sofisticada, apresentou desempenho inferior em todas as métricas de qualidade (`SBERT Score` de 0.5914 e `BERT F1` de 0.8651). Este resultado evidencia as limitações inerentes ao *fine-tuning* em cenários de dados escassos, onde o volume de 671 pares pergunta-resposta mostrou-se insuficiente para alcançar níveis de generalização comparáveis aos modelos pré-treinados em larga escala. A performance aquém do esperado também pode ser atribuída à complexidade de otimização de hiperparâmetros em modelos de grande escala, à necessidade de técnicas mais sofisticadas de aumento de dados, e às limitações inerentes de modelos abertos em comparação com modelos proprietários otimizados comercialmente.

5.1.2 Eficiência Temporal e Experiência do Usuário

Quanto à segunda questão de pesquisa, sobre eficiência temporal e impacto na experiência do usuário, os resultados revelaram um *trade-off* complexo e multifacetado entre qualidade e velocidade de resposta. O modelo `Mistral` demonstrou a maior eficiência temporal, com tempo médio de resposta de 6.14 segundos e desvio padrão mínimo de 0.03 segundos, refletindo a estabilidade inerente à execução local sem dependência de requisições externas. Esta consistência temporal representa uma vantagem significativa para aplicações que demandam previsibilidade de resposta e controle total sobre a infraestrutura de processamento.

A abordagem `RAG` apresentou desempenho intermediário (6.68 segundos), beneficiando-se da geração de contexto local via `FAISS` para reduzir a latência de recuperação, enquanto mantém dependência da `API` da `OpenAI` para geração final. Este híbrido de processamento local e remoto oferece um equilíbrio interessante entre controle e qualidade, embora introduza complexidade adicional na arquitetura do sistema.

A abordagem via `API` direta apresentou a maior latência (10.30 segundos) e maior

variabilidade (desvio padrão de 4.09 segundos), evidenciando os gargalos inerentes à comunicação com infraestruturas externas. No entanto, é crucial contextualizar estes resultados considerando que a diferença temporal, embora estatisticamente significativa, pode ser aceitável em muitos contextos de atendimento ao cliente, especialmente quando considerada a superioridade qualitativa das respostas geradas. A variabilidade observada sugere a necessidade de implementação de estratégias de otimização, como *caching* inteligente, balanceamento de carga, e técnicas de predição de demanda para mitigar picos de latência.

5.1.3 Análise de Custos e Viabilidade Econômica

A terceira questão de pesquisa, que investigava os *trade-offs* entre complexidade, custo e qualidade, revelou insights fundamentais que redefinem a percepção convencional sobre a viabilidade econômica de diferentes abordagens de `LLM`. A análise de custos de desenvolvimento, que incluiu tanto os custos compartilhados de preparação de dados quanto os custos específicos de cada implementação, forneceu uma perspectiva realista e abrangente sobre o investimento necessário para cada abordagem.

A abordagem via `API` emergiu como a solução mais eficiente economicamente, requerendo investimento total de R$ 8.042,47, incluindo os custos obrigatórios de preparação de dados (R$ 4.392,47) e custos específicos de desenvolvimento (R$ 3.650,00). Esta eficiência econômica, combinada com a superior qualidade de resposta, estabelece uma proposta de valor excepcionalmente competitiva para organizações com recursos limitados. A simplicidade de implementação, que requer apenas 24 horas de desenvolvimento específico, torna esta abordagem particularmente atrativa para empresas que buscam resultados rápidos e eficazes com investimento controlado.

A implementação `RAG` demandou investimento intermediário de R$ 10.442,47, refletindo a complexidade adicional de configuração de sistemas de recuperação semântica, otimização de índices vetoriais, e integração entre componentes de recuperação e geração. Embora represente maior investimento inicial (30% superior à abordagem `API`), oferece vantagens estratégicas importantes, incluindo maior controle sobre a infraestrutura, independência de provedores externos para a base de conhecimento, e potencial redução de custos operacionais de longo prazo através da eliminação de taxas de `API` para processamento de contexto.

O *fine-tuning* do `Mistral` exigiu o maior investimento, totalizando R$ 15.442,47, representando um custo 92% superior à abordagem `API`. Este investimento substancial reflete a complexidade técnica inerente ao processo de *fine-tuning*, incluindo 72 horas de desenvolvimento distribuídas entre codificação (40 horas), treinamento (24 horas) e testes (8 horas), além da necessidade de expertise especializada em otimização de modelos de linguagem, configuração de hiperparâmetros, e implementação de técnicas de `PEFT` como `LoRA`.

Quando considerado em conjunto com a *performance* inferior em qualidade, o alto custo do *fine-tuning* questiona fundamentalmente sua viabilidade em cenários de dados limitados. Esta constatação tem implicações importantes para a prática empresarial, sugerindo que investimentos em *fine-tuning* podem ser mais adequados para contextos com maior disponibilidade de dados, recursos computacionais substanciais, e necessidades específicas de personalização que justifiquem o investimento adicional.

5.1.4 Implicações para a Tomada de Decisão Empresarial

A análise integrada dos resultados de qualidade, eficiência temporal e custos de desenvolvimento permite a formulação de recomendações específicas e contextualizadas para diferentes perfis organizacionais. Para empresas que priorizam a qualidade das respostas e buscam a melhor relação custo-benefício, a abordagem de uso direto de `LLM` via `API` representa inequivocamente a opção mais robusta. A combinação de superior qualidade de resposta (melhor em todas as métricas avaliadas) com o menor investimento total (R$ 8.042,47) estabelece uma proposta de valor que é difícil de superar em cenários de recursos limitados.

Organizações que buscam maior controle sobre custos operacionais de longo prazo e infraestrutura, mantendo qualidade competitiva, podem encontrar na abordagem `RAG` uma alternativa estrategicamente equilibrada. O investimento adicional de R$ 2.400,00 (30% superior) pode justificar-se em cenários onde a autonomia tecnológica, o controle sobre dados sensíveis, e a independência de provedores externos são prioritários. Adicionalmente, organizações com expertise técnica interna em sistemas de recuperação de informação podem otimizar implementações `RAG` para alcançar resultados próximos aos obtidos com soluções proprietárias.

O *fine-tuning*, com investimento de R$ 15.442,47, apresenta limitações significativas em cenários de dados escassos, permanecendo como opção viável apenas para contextos muito específicos onde a autonomia tecnológica completa, a eficiência temporal máxima, e a personalização extrema são prioritárias, desde que haja disponibilidade de recursos substanciais para desenvolvimento e otimização contínua. Os resultados sugerem que esta abordagem pode ser mais adequada para organizações com maior maturidade em dados, capacidade de investimento em expertise técnica especializada, e necessidades de personalização que não podem ser atendidas por soluções mais convencionais.

5.2 Contribuições do Estudo

Este trabalho oferece contribuições práticas, metodológicas e teóricas relevantes para múltiplas dimensões do campo de automação de atendimento ao cliente e aplicação de `LLMs` em contextos empresariais. A natureza multidisciplinar das contribuições reflete a

complexidade inerente à implementação de soluções de inteligência artificial em ambientes organizacionais reais.

5.2.1 Contribuições Práticas

Do ponto de vista prático, este estudo fornece um guia baseado em evidências empíricas para empresas que buscam implementar soluções de chatbot em contextos de recursos limitados. As recomendações específicas, fundamentadas em análise rigorosa de custo-benefício, oferecem critérios objetivos para a tomada de decisão tecnológica que consideram não apenas aspectos técnicos, mas também viabilidade econômica, complexidade de implementação, e sustentabilidade de longo prazo.

A inclusão de análise detalhada de custos de desenvolvimento representa um diferencial importante em relação à literatura existente, preenchendo uma lacuna frequentemente negligenciada em estudos acadêmicos que focam exclusivamente em aspectos técnicos. A metodologia de cálculo de custos desenvolvida, que inclui tanto custos compartilhados quanto específicos de cada abordagem, pode ser aplicada por organizações para avaliação de viabilidade de projetos similares, adaptando-se a diferentes contextos geográficos e econômicos através de ajustes nos custos de mão de obra especializada.

As descobertas sobre a superioridade da abordagem `API` em cenários de dados limitados têm implicações práticas imediatas para pequenas e médias empresas, que constituem a maioria do tecido empresarial brasileiro e frequentemente enfrentam limitações de recursos para investimentos em tecnologia. A demonstração de que soluções mais simples e menos custosas podem superar abordagens tecnicamente mais sofisticadas em determinados contextos desafia percepções convencionais e oferece alternativas viáveis para organizações com restrições orçamentárias.

5.2.2 Contribuições Metodológicas

A metodologia experimental desenvolvida, com métricas padronizadas e procedimento replicável, constitui uma contribuição metodológica significativa que pode ser aplicada em estudos similares. A combinação de métricas semânticas (`SBERT Score`) e lexicais (`BERT Precision`, `Recall`, `F1`) oferece uma avaliação abrangente da qualidade de resposta que captura tanto aspectos de alinhamento conceitual quanto precisão textual.

O protocolo experimental, que incluiu múltiplas repetições para cada pergunta e análise estatística de estabilidade, estabelece padrões de rigor científico que podem ser adotados em pesquisas futuras. A decisão de utilizar medianas em vez de médias, fundamentada na análise de distribuição dos dados, demonstra sensibilidade metodológica que aumenta a robustez dos resultados.

A abordagem de análise integrada, que considera simultaneamente qualidade, eficiência temporal e custos de desenvolvimento, oferece um *framework* holístico para

avaliação de tecnologias de LLM que pode ser adaptado para diferentes contextos de aplicação. Esta perspectiva multidimensional é particularmente valiosa em contextos empresariais, nos quais decisões tecnológicas devem considerar múltiplos fatores além da performance técnica pura.

5.2.3 Contribuições Teóricas

A análise comparativa em cenário de dados restritos preenche uma lacuna importante identificada na literatura, que frequentemente aborda essas tecnologias em contextos de grandes volumes de dados e recursos computacionais abundantes. Os resultados evidenciam que a eficácia de diferentes abordagens de LLM é altamente dependente do contexto de aplicação, particularmente em relação à disponibilidade de dados, recursos computacionais e restrições orçamentárias.

A constatação de que modelos proprietários pré-treinados podem superar implementações personalizadas em cenários de dados escassos tem implicações teóricas importantes para a compreensão de quando e como aplicar diferentes técnicas de LLM. Esta descoberta contribui para o desenvolvimento de uma teoria mais nuançada sobre a aplicação contextual de tecnologias de inteligência artificial, que considera não apenas capacidades técnicas, mas também viabilidade prática e sustentabilidade econômica.

O estudo também contribui para a literatura sobre democratização de tecnologias de inteligência artificial, demonstrando como soluções baseadas em API podem reduzir barreiras de entrada para organizações menores, potencialmente acelerando a adoção de tecnologias avançadas em segmentos tradicionalmente excluídos por limitações de recursos ou expertise técnica.

5.3 Limitações e Trabalhos Futuros

Este estudo apresenta limitações importantes que devem ser consideradas na interpretação dos resultados e que abrem oportunidades valiosas para pesquisas futuras. O reconhecimento explícito dessas limitações é fundamental para o avanço científico da área e para orientar investigações subsequentes que possam expandir e refinar os achados apresentados.

5.3.1 Limitações do Escopo e Generalização

A principal limitação refere-se ao escopo restrito do conjunto de dados utilizado, baseado em materiais de uma única empresa fictícia do setor de tecnologia. Esta limitação impacta diretamente a generalização dos resultados para outros domínios, setores econômicos, e tipos de organização. Estudos futuros poderiam expandir significativamente a análise para múltiplos domínios (saúde, educação, serviços financeiros, varejo), tipos de empresa (startups, corporações, organizações sem fins lucrativos), e contextos geográficos,

avaliando como diferentes características setoriais, culturais e regulatórias influenciam a eficácia relativa das abordagens analisadas.

A avaliação foi conduzida exclusivamente em português brasileiro, limitando a aplicabilidade dos resultados a outros idiomas e contextos linguísticos. Esta limitação é particularmente relevante considerando que muitas empresas operam em mercados globais e necessitam de soluções multilíngues. Pesquisas futuras poderiam investigar sistematicamente o comportamento dessas abordagens em contextos multilíngues, idiomas com características linguísticas distintas (idiomas tonais, aglutinantes, com sistemas de escrita diferentes), e cenários de tradução automática integrada.

5.3.2 Limitações Temporais e Econômicas

A análise de custos focou exclusivamente nos custos de desenvolvimento inicial, não contemplando custos operacionais de longo prazo, como taxas de `API`, custos de infraestrutura, manutenção, atualizações de modelos, e evolução de requisitos organizacionais. Esta limitação é significativa porque decisões tecnológicas empresariais devem considerar o custo total de propriedade (*Total Cost of Ownership*) ao longo de múltiplos anos de operação.

Estudos longitudinais que avaliem a evolução dos custos e benefícios dessas abordagens ao longo do tempo forneceriam insights valiosos para decisões estratégicas empresariais. Tais estudos poderiam considerar fatores dinâmicos como evolução dos modelos de `LLM`, mudanças nos preços de `APIs`, custos de manutenção e atualização, impacto da curva de aprendizado organizacional, e retorno sobre investimento mensurado através de métricas de satisfação do cliente e eficiência operacional.

5.3.3 Limitações Metodológicas

A avaliação baseou-se em métricas automatizadas de qualidade textual, que, embora objetivas e replicáveis, podem não capturar completamente aspectos subjetivos da experiência do usuário, como satisfação, percepção de utilidade, e adequação contextual das respostas. A ausência de avaliação humana direta representa uma limitação importante que poderia ser endereçada em estudos futuros através de protocolos de avaliação que incluam usuários reais, cenários de uso autênticos, e métricas de satisfação e eficácia percebida.

Adicionalmente, o estudo não investigou aspectos importantes como robustez a ataques adversariais, comportamento em cenários de *prompt injection*, capacidade de lidar com consultas ambíguas ou mal-intencionadas, e adequação a diferentes perfis de usuário. Estas dimensões são cruciais para implementações comerciais e representam oportunidades importantes para pesquisas futuras.

5.4 Direções para o Estado da Arte

Para alcançar o estado da arte na área e abordar as limitações identificadas, sugerem-se direções de pesquisa específicas que podem contribuir significativamente para o avanço do conhecimento e da prática em automação conversacional empresarial.

5.4.1 Arquiteturas Híbridas e Otimização Multi-Objetivo

O desenvolvimento de arquiteturas híbridas que combinem as vantagens de diferentes abordagens representa uma direção promissora e tecnicamente desafiadora. A integração de técnicas de `RAG` com *fine-tuning* seletivo poderia potencialmente combinar a precisão contextual do `RAG` com a especialização do *fine-tuning*, mitigando as limitações observadas em cada abordagem isoladamente. Tais arquiteturas poderiam implementar sistemas de roteamento inteligente que selecionam dinamicamente a abordagem mais adequada baseada em características da consulta, contexto do usuário, e requisitos de qualidade versus latência.

A investigação de técnicas de otimização multi-objetivo, que balanceiem simultaneamente qualidade de resposta, eficiência temporal, custos operacionais, e robustez, poderia levar ao desenvolvimento de soluções mais equilibradas e adaptáveis a diferentes contextos organizacionais. Tais técnicas poderiam incluir algoritmos de aprendizado por reforço para otimização dinâmica de parâmetros, sistemas de *ensemble* que combinem múltiplos modelos, e arquiteturas adaptativas que se ajustem automaticamente a mudanças nos padrões de uso.

5.4.2 Técnicas Avançadas de Aumento de Dados e Aprendizado com Poucos Exemplos

A investigação de técnicas sofisticadas de aumento de dados representa uma direção crucial para abordar diretamente a limitação de dados escassos identificada neste estudo. Técnicas de geração sintética de perguntas e respostas, *data augmentation* baseado em `LLMs`, paráfrase automática, e tradução reversa poderiam expandir significativamente conjuntos de dados limitados, potencialmente melhorando a eficácia do *fine-tuning* em cenários de recursos restritos.

O desenvolvimento de técnicas avançadas de *few-shot* e *zero-shot learning* especificamente adaptadas para contextos empresariais poderia reduzir ainda mais a dependência de grandes volumes de dados de treinamento. Tais técnicas poderiam incluir *meta-learning* para adaptação rápida a novos domínios, *prompt engineering* automatizado baseado em características do domínio, e técnicas de transferência de conhecimento entre domínios relacionados.

5.4.3 Métricas de Avaliação Especializadas e Centradas no Usuário

O desenvolvimento de métricas de avaliação mais específicas para chatbots de atendimento ao cliente representa uma necessidade crítica para o avanço da área. Tais métricas deveriam considerar aspectos como satisfação do usuário, resolução efetiva de problemas, adequação ao contexto empresarial, manutenção de consistência de marca, e capacidade de escalação apropriada para atendimento humano quando necessário.

A investigação de métricas que capturem aspectos subjetivos da experiência do usuário, como percepção de empatia, naturalidade da interação, e confiança na informação fornecida, complementaria as métricas técnicas utilizadas neste estudo. Tais métricas poderiam ser desenvolvidas através de estudos com usuários reais, análise de sentimento em interações autênticas, e correlação entre métricas objetivas e satisfação percebida.

5.4.4 Personalização Dinâmica e Aprendizado Contínuo

A investigação de sistemas que combinem personalização dinâmica com aprendizado contínuo a partir das interações com usuários reais poderia representar um avanço significativo em direção a chatbots verdadeiramente adaptativos e eficazes. Tais sistemas poderiam implementar técnicas de aprendizado online que se ajustem continuamente a mudanças nos padrões de consulta, evolução da base de conhecimento organizacional, e feedback implícito e explícito dos usuários.

O desenvolvimento de arquiteturas que mantenham múltiplas versões de modelos especializados para diferentes tipos de consulta, perfis de usuário, ou contextos organizacionais poderia oferecer personalização sem comprometer a eficiência ou aumentar excessivamente a complexidade do sistema. Tais arquiteturas poderiam incluir sistemas de roteamento baseados em aprendizado de máquina, técnicas de *ensemble* dinâmico, e mecanismos de atualização incremental que preservem conhecimento anterior enquanto incorporam novas informações.

5.5 Considerações Finais

Os resultados deste estudo demonstram de forma conclusiva que não existe uma solução universalmente superior para implementação de chatbots em contextos empresariais com dados limitados. Esta constatação, longe de representar uma limitação, reflete a riqueza e complexidade do campo de aplicação de `LLMs` em contextos organizacionais reais. A escolha da abordagem mais adequada deve considerar cuidadosamente as prioridades específicas de cada organização, balanceando qualidade de resposta, eficiência temporal, custos de desenvolvimento, complexidade de implementação, recursos disponíveis, e objetivos estratégicos de longo prazo.

5.5.1 Recomendações Estratégicas por Perfil Organizacional

Para empresas que priorizam a qualidade das respostas e buscam a melhor relação custo-benefício, particularmente pequenas e médias empresas com recursos limitados, a abordagem de uso direto de `LLM` via `API` representa inequivocamente a opção mais robusta e estrategicamente inteligente. A combinação de superior qualidade de resposta (demonstrada através de múltiplas métricas independentes) com o menor investimento total (R$ 8.042,47) estabelece uma proposta de valor que é excepcionalmente competitiva. A simplicidade de implementação e manutenção, combinada com a capacidade de beneficiar-se automaticamente de melhorias contínuas nos modelos proprietários, torna esta abordagem particularmente atrativa para organizações que buscam resultados rápidos e eficazes com investimento controlado e risco tecnológico minimizado.

Organizações de médio e grande porte que buscam maior controle sobre custos operacionais de longo prazo e infraestrutura, mantendo qualidade competitiva, podem encontrar na abordagem `RAG` uma alternativa estrategicamente equilibrada e tecnicamente sofisticada. O investimento adicional de R$ 2.400,00 (30% superior à abordagem `API`) pode justificar-se em cenários onde a autonomia tecnológica, o controle sobre dados sensíveis, a independência de provedores externos, e a capacidade de customização avançada são prioritários estratégicos. Organizações com expertise técnica interna em sistemas de recuperação de informação e processamento de linguagem natural podem otimizar implementações `RAG` para alcançar resultados próximos aos obtidos com soluções proprietárias, enquanto mantêm controle total sobre a infraestrutura e os dados.

O *fine-tuning*, com investimento substancial de R$ 15.442,47, apresenta limitações significativas em cenários de dados escassos que questionam sua viabilidade para a maioria das organizações. Esta abordagem permanece como opção viável apenas para contextos muito específicos nos quais autonomia tecnológica completa, a eficiência temporal máxima, a personalização extrema, e o controle total sobre propriedade intelectual são prioritários estratégicos absolutos, desde que haja disponibilidade de recursos substanciais para desenvolvimento, otimização contínua, e manutenção especializada. Os resultados sugerem que esta abordagem pode ser mais adequada para organizações com maior maturidade em dados, capacidade de investimento em expertise técnica altamente especializada, e necessidades de personalização que não podem ser atendidas por soluções mais convencionais.

5.5.2 Implicações para a Evolução Tecnológica

Este trabalho contribui significativamente para o avanço do conhecimento na área de automação de atendimento ao cliente e aplicação prática de `LLMs` em contextos empresariais. As evidências empíricas apresentadas podem orientar tanto decisões práticas imediatas quanto futuras pesquisas acadêmicas e desenvolvimento tecnológico. A metodologia desenvolvida e os resultados obtidos estabelecem uma base sólida e replicável para

investigações mais aprofundadas sobre a aplicação de LLMs em contextos empresariais específicos, particularmente em cenários de recursos limitados que caracterizam a realidade de muitas organizações brasileiras e internacionais.

A demonstração de que soluções baseadas em API podem democratizar o acesso a tecnologias avançadas de inteligência artificial tem implicações importantes para a inclusão digital e competitividade empresarial. Esta descoberta sugere que barreiras tradicionais para adoção de IA, como necessidade de expertise técnica especializada, investimentos substanciais em infraestrutura, e disponibilidade de grandes volumes de dados, podem ser significativamente reduzidas através de estratégias tecnológicas apropriadas.

5.5.3 Perspectivas Futuras e Sustentabilidade

A crescente democratização das tecnologias de Inteligência Artificial e a contínua evolução dos *Large Language Models* sugerem que as conclusões específicas deste estudo devem ser revisitadas periodicamente, à medida que novas ferramentas, técnicas, e modelos se tornem disponíveis. A velocidade de inovação na área de LLMs é extraordinária, com novos modelos, técnicas de treinamento, e abordagens de implementação sendo desenvolvidos continuamente.

No entanto, os princípios metodológicos, os critérios de avaliação, e o *framework* de análise estabelecidos neste trabalho mantêm sua relevância duradoura como base para análises comparativas futuras na área. A importância de considerar aspectos econômicos em conjunto com métricas técnicas para uma avaliação holística de viabilidade tecnológica representa uma contribuição metodológica que transcende tecnologias específicas e pode ser aplicada a futuras inovações em inteligência artificial conversacional.

A sustentabilidade de longo prazo das soluções propostas dependerá não apenas de fatores técnicos e econômicos, mas também de considerações éticas, regulatórias, e sociais que estão em constante evolução. Questões como privacidade de dados, transparência algorítmica, viés em sistemas de IA, e impacto no emprego humano tornar-se-ão cada vez mais relevantes à medida que essas tecnologias se tornem mais prevalentes em contextos organizacionais.

Este estudo, ao fornecer evidências empíricas rigorosas sobre a viabilidade prática de diferentes abordagens de implementação de chatbots, contribui para um desenvolvimento mais informado e responsável de tecnologias de automação conversacional. As recomendações apresentadas podem orientar organizações na tomada de decisões tecnológicas que balanceiem eficácia, eficiência, sustentabilidade econômica, e responsabilidade social, promovendo uma adoção mais madura e estratégica de tecnologias de inteligência artificial em contextos empresariais brasileiros e internacionais.

REFERÊNCIAS

ADAMOPOULOU, E.; MOUSSIADES, L. Chatbots: History, technology, and applications. **Machine Learning with Applications**, Elsevier, v. 2, p. 100006, 2020.

BROWN, T. *et al.* Language models are few-shot learners. *In*: **Advances in Neural Information Processing Systems**. [*S.l.: s.n.*], 2020. v. 33, p. 1877–1901.

COLLINS, E. *et al.* Artificial intelligence in information systems research: A systematic literature review and research agenda. **Journal of Decision Systems**, v. 30, n. 2-3, p. 172–199, 2021.

DEVLIN, J. *et al.* Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

FøLSTAD, A.; BRANDTZæG, P. B. Why people use chatbots. *In*: **Proceedings of the 4th International Conference on Internet Science (INSCI)**. [*S.l.: s.n.*]: Springer, 2017. p. 377–392.

GEHMAN, S. *et al.* Realtoxicityprompts: Evaluating neural toxic degeneration in language models. **arXiv preprint arXiv:2009.11462**, 2020. Disponível em: https://arxiv.org/abs/2009.11462.

GNEWUCH, U.; MORANA, S.; MAEDCHE, A. Towards designing cooperative and social conversational agents for customer service. *In*: **Proceedings of the 38th International Conference on Information Systems (ICIS)**. Seoul, South Korea: Association for Information Systems, 2017. p. 1–21.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge, MA: MIT Press, 2016. ISBN 9780262035613.

GéRON, A. **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. 2. ed. Sebastopol: O'Reilly Media, 2019. ISBN 9781492032646.

HU, E. J. *et al.* Lora: Low-rank adaptation of large language models. **arXiv preprint arXiv:2106.09685**, 2021. Disponível em: https://arxiv.org/abs/2106.09685.

JAIN, M. *et al.* Evaluating and informing the design of chatbots. *In*: **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems**. [*S.l.: s.n.*]: ACM, 2018.

LEWIS, P. *et al.* Retrieval-augmented generation for knowledge-intensive nlp tasks. *In*: **Advances in Neural Information Processing Systems**. [*S.l.: s.n.*], 2020. v. 33, p. 9459–9474.

MAYNARD, D.; FUNK, A. Automatic detection of public health misinformation: a targeted approach to covid-19. **Frontiers in Artificial Intelligence**, v. 3, 2020.

MCTEAR, M. **Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots**. San Rafael, CA: Morgan & Claypool Publishers, 2022. ISBN 9781630818855.

MIKOLOV, T. *et al.* Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997. ISBN 9780070428072.

OUYANG, L. *et al.* Training language models to follow instructions with human feedback. **arXiv preprint arXiv:2203.02155**, 2022. Disponível em: https://arxiv.org/abs/2203.02155.

RADFORD, A. *et al.* **Language Models are Unsupervised Multitask Learners**. [*S.l.*], 2019. Technical report.

RADZIWILL, N.; BENTON, M. C. Evaluating quality of chatbots and intelligent conversational agents. **Journal of Intelligent & Robotic Systems**, v. 91, p. 1–14, 2017.

REIMERS, N.; GUREVYCH, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *In*: **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [*S.l.: s.n.*], 2019. p. 3982–3992.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 2. ed. Upper Saddle River, NJ: Prentice Hall, 2003.

SHAWAR, B. A.; ATWELL, E. Chatbots: Are they really useful? **LDV Forum**, v. 22, n. 1, p. 29–49, 2007.

SHEVAT, A. **Designing Bots: Creating Conversational Experiences**. Beijing; Boston; Farnham; Sebastopol: O'Reilly Media, 2017. ISBN 9781491957476.

VASWANI, A. *et al.* Attention is all you need. *In*: **Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)**. Curran Associates, Inc., 2017. p. 5998–6008. Disponível em: https://arxiv.org/abs/1706.03762.

WEI, J. *et al.* Chain-of-thought prompting elicits reasoning in large language models. **arXiv preprint arXiv:2201.11903**, 2022. Disponível em: https://arxiv.org/abs/2201.11903.

ZHANG, T. *et al.* Bertscore: Evaluating text generation with bert. **arXiv preprint arXiv:1904.09675**, 2019. Disponível em: https://arxiv.org/abs/1904.09675.

APÊNDICES

APÊNDICE A – PROMPT DO ASSISTENTE – USO DIRETO DE LLM VIA API

Você é AndréIA, o assistente virtual do Grupo Regazzo, uma empresa referência em tecnologia, outsourcing, desenvolvimento de software e consultoria em TI. Sua missão é garantir que os clientes tenham uma experiência excepcional, fornecendo respostas precisas, confiáveis e impactantes com base nos documentos disponíveis no sistema.

Tarefa Primária

Você deve processar a pergunta recebida e responder de acordo com sua base de conhecimento no formato de texto puro. Trabalhe exclusivamente com as informações apresentadas, sem inferir ou criar dados que não estejam explicitamente fornecidos.

Regras Essenciais

Sempre responda com base nos documentos disponíveis no sistema.

Se a resposta não for encontrada, utilize: "Desculpe, essa pergunta está fora do meu escopo. Posso te ajudar com informações sobre os serviços e soluções digitais da Regazzo."

Jamais invente ou suponha uma resposta. Mantenha um tom profissional, cortês e próximo ao cliente.

Se a pergunta for vaga ou ambígua, solicite mais detalhes.

Nunca forneça informações externas ou baseadas no conhecimento geral do mercado.

Formato de Saída

O resultado não deverá conter nenhum texto ou formatação adicional além da resposta pura.

APÊNDICE B – PROMPT DO ASSISTENTE – ABORDAGEM RAG

Você é AndréIA, o assistente virtual do Grupo Regazzo, uma empresa referência em tecnologia, outsourcing, desenvolvimento de software e consultoria em TI.

Sua missão é garantir que os clientes tenham uma experiência excepcional, fornecendo respostas precisas, confiáveis e impactantes com base nas informações do contexto fornecido com cada pergunta.

Tarefa Primária

Você deve processar a pergunta recebida e responder utilizando exclusivamente o conteúdo do contexto fornecido.

Regras Essenciais

Sempre responda com base no contexto fornecido.

Se a resposta não for encontrada, utilize: "Desculpe, essa pergunta está fora do meu escopo. Posso te ajudar com informações sobre os serviços e soluções digitais da Regazzo."

Jamais invente ou suponha uma resposta. Mantenha um tom profissional, cortês e próximo ao cliente.

Se a pergunta for vaga ou ambígua, solicite mais detalhes.

Nunca utilize conhecimento prévio ou externo ao contexto fornecido.

Formato de Saída

O resultado não deverá conter nenhum texto ou formatação adicional além da resposta pura.

Exemplo: "O Grupo Regazzo foi fundado em 1998 por André Regazzo, que iniciou sua trajetória criando a Regazzo Soluções em Tecnologia...