Evelyn Mylena Bezerra e Silva

Implementação de um Agente Inteligente para Atendimento Automatizado de Dúvidas Acadêmicas na UFRPE

Recife

Março de 2025

Evelyn Mylena Bezerra e Silva

Implementação de um Agente Inteligente para Atendimento Automatizado de Dúvidas Acadêmicas na UFRPE

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE
Departamento de Estatística e Informática
Curso de Bacharelado em Sistemas de Informação

Orientador: Gabriel Alves De Albuquerque Junior

Recife

Março de 2025

Implementação de um Agente Inteligente para Atendimento Automatizado de Dúvidas Acadêmicas na UFRPE

Evelyn Mylena Bezerra e Silva [1], **Gabriel Alves de Albuquerque Júnior** [2]

[1]Departamento de Estatística e Informática – Universidade Federal Rural de Pernambuco
Rua Dom Manuel de Medeiros, s/n, - CEP: 52171-900 – Recife – PE – Brasil

[evelyn.mylena@ufrpe.br, gabriel.alves@ufrpe.br]

Resumo. *Este trabalho teve como objetivo principal desenvolver e validar um agente inteligente baseado em técnicas de Processamento de Linguagem Natural (PLN) e Recuperação de Informação, voltado ao suporte acadêmico no curso de Bacharelado em Sistemas de Informação (BSI) da Universidade Federal Rural de Pernambuco (UFRPE). O sistema foi projetado para oferecer respostas rápidas, relevantes e contextualizadas a perguntas frequentes relacionadas à vida acadêmica dos estudantes, como carga horária, disciplinas, matrículas e procedimentos administrativos. A implementação envolveu a coleta e estruturação de documentos institucionais, a construção de um modelo vetorial para recuperação semântica de respostas e a integração com um módulo de memória conversacional. Para a validação, as respostas do agente inteligente foram comparadas com um FAQ do curso de BSI, utilizando como métrica a similaridade do cosseno, aplicada ao conteúdo semântico das respostas. Os resultados indicaram uma média de similaridade de aproximadamente* $0{,}6396$*, com mediana de* $0{,}6548$ *e baixa dispersão. A maioria das respostas apresentou alto ou médio grau de alinhamento semântico com o conteúdo oficial, sendo classificadas como semanticamente adequadas. Casos de baixa similaridade representaram uma minoria e estiveram, em geral, relacionados a perguntas de cunho prático-operacional não abordadas na base de dados do sistema. Em contrapartida, observou-se que, em alguns contextos, o agente inteligente forneceu respostas mais completas e fundamentadas do que aquelas presentes no próprio FAQ. Conclui-se que o sistema desenvolvido apresenta desempenho satisfatório e demonstra potencial para expansão como ferramenta institucional de apoio ao estudante, promovendo maior autonomia, agilidade e acessibilidade no acesso à informação acadêmica.*

Abstract. *This work aimed to develop and validate a intelligent agent based on Natural Language Processing (NLP) and Information Retrieval techniques, designed to provide academic support for the Bachelor of Information Systems (BSI) program at the Federal Rural University of Pernambuco (UFRPE). The system was designed to deliver fast, relevant, and contextualized responses to frequently asked questions related to students' academic life, such as workload, courses, enrollment, and administrative procedures. The implementation involved the collection and structuring of institutional documents, the construction of a vector-based model for semantic answer retrieval, and the integration of a conversational memory module. For validation, the intelligent agent's responses were compared to those found in BSI course FAQ, using cosine similarity*

as the metric, applied to the semantic content of the answers. The results indicated an average similarity of approximately 0.6396, a median of 0.6548, and low dispersion. Most responses demonstrated a high or moderate degree of semantic alignment with the official content, being classified as semantically adequate. Cases of low similarity were limited and generally related to operational or procedural questions not covered in the system's data sources. On the other hand, it was observed that in certain situations, the intelligent agent provided answers that were more complete and better substantiated than those found in the FAQ itself. It is concluded that the developed system demonstrates satisfactory performance and shows potential for expansion as an institutional tool to support students, promoting greater autonomy, agility, and accessibility in accessing academic information.

1. Introdução

No ambiente acadêmico, é comum que os estudantes tenham dúvidas sobre normas e regulamentos do curso, muitas vezes recorrendo a e-mails ou interações presenciais com coordenadores e professores para obter esclarecimentos. Com a crescente digitalização, torna-se essencial a adoção de tecnologias inovadoras no setor educacional. Entre essas soluções, os chatbots se destacam por oferecer diversas vantagens [da Silva 2018], promovendo colaboração, cooperação, interação e um processo de aprendizagem mais dinâmico e construtivo [Bii 2013].

Ao longo das décadas, os chatbots passaram por uma evolução significativa, impulsionada por avanços tecnológicos e pela crescente demanda por sistemas de conversação [Alvarenga et al. 2024]. Uma revisão da literatura conduzida por [Larisane et al. 2018] identificou que os principais propósitos da utilização de chatbots na educação incluem seu papel como tutor inteligente, o suporte a metodologias de ensino inovadoras, a facilitação da autoaprendizagem e o incentivo à aprendizagem colaborativa. Essas características evidenciam o potencial dessas ferramentas para transformar não apenas a experiência de aprendizado, mas também a gestão acadêmica.

Nesse contexto, estudos como o de [Smutný and Schreiberova 2020] destacam que os chatbots podem reduzir significativamente a carga de trabalho dos professores e coordenadores, ao automatizar respostas a perguntas frequentes e fornecer suporte contínuo aos estudantes. Além disso, pesquisas como a de [Winkler and Söllner 2018] evidenciam que a utilização de chatbots baseados em IA pode melhorar a satisfação dos alunos, ao oferecer respostas instantâneas e personalizadas, adaptadas às suas necessidades específicas. Esses sistemas também contribuem para a democratização do acesso à informação, garantindo que todos os estudantes, independentemente de sua disponibilidade ou localização, possam obter esclarecimentos de forma equitativa [Holmes et al. 2019].

Diante desse cenário, soluções baseadas em IA têm emergido como ferramentas promissoras, capazes de melhorar a comunicação entre estudante e instituição, promovendo agilidade e eficiência. Com o objetivo de otimizar esse processo, o presente trabalho propõe uma solução prática, desenvolvendo uma ferramenta destinada a agilizar o esclarecimento das dúvidas dos discentes. Com o uso desse sistema, os coordenadores de curso poderão economizar tempo, uma vez que não precisarão se dedicar a questões frequentes, possibilitando respostas mais rápidas aos alunos. Dessa forma, a implementação

de um agente inteligente não apenas otimiza processos administrativos, mas também fortalece a inclusão e a eficiência no ambiente educacional.

1.1. Objetivo geral

O objetivo central deste trabalho é desenvolver um agente inteligente com o propósito de atender às demandas informativas dos discentes do curso de Bacharelado em Sistemas de Informação (BSI) da Universidade Federal Rural de Pernambuco (UFRPE). A solução proposta tem como objetivo fornecer respostas ágeis, acuradas e contextualizadas a perguntas relacionadas ao contexto acadêmico, abrangendo temas como carga horária, componentes curriculares, processos de matrícula e procedimentos administrativos institucionais. Para garantir a confiabilidade das respostas, o agente inteligente será projetado para funcionar de forma independente de modelos de linguagem pré-treinaods, baseando suas respostas exclusivamente dos documentos oficiais do curso, instituição e regulamentações legais. Dessa forma, pretende-se otimizar a comunicação entre os estudantes e a instituição, reduzindo a burocracia e promovendo uma interação eficiente, acessível, atualizada e alinhada às necessidades do público acadêmico.

1.1.1. Objetivos específicos

- Coletar, organizar e estruturar a documentação acadêmica atualizada relevante para a UFRPE, incluindo FAQs, regulamentos e procedimentos institucionais;
- Desenvolver um modelo de embeddings baseado em tecnologias de processamento de linguagem natural (NLP) para realizar a indexação e recuperação eficiente de informações;
- Integrar o modelo de linguagem *LangChain* com uma arquitetura de Geração Aumentada de Recuperação (RAG) para garantir que as respostas fornecidas pelo agente inteligente sejam contextualizadas na documentação oficial, permitindo a recuperação de informações de forma independente;
- Implementar um sistema de memória conversacional, permitindo que o agente inteligente mantenha o contexto das interações com os usuários durante as sessões de conversa;
- Avaliar a precisão e relevância das respostas fornecidas pelo agente inteligente.

A estrutura do presente artigo está organizada da seguinte maneira: na seção de Trabalhos Relacionados, são discutidas principalmente as técnicas avançadas de inteligência artificial, destacando a integração entre linguagem natural e métodos eficientes de recuperação de informações. O Referencial Teórico aprofunda os conceitos e fundamentos relevantes ao tema. Em Materiais e Métodos, são detalhados o processo de desenvolvimento da ferramenta e a metodologia utilizada. A seção de Resultados apresenta as principais descobertas do estudo. Por fim, as Considerações Finais destacam as conclusões alcançadas e as implicações do trabalho realizado.

2. Referencial Teórico

Neste referencial teórico, abordaremos os principais conceitos utilizados que fundamentam a temática deste trabalho. Primeiramente, iremos compreender da linguagem de programação Python, que é a linguagem principal da implementação do agente inteligente, posteriormente, exploraremos o Processamento de Linguagem Natural (PLN), que

fundamenta a interação entre humanos e máquinas por meio da linguagem natural, e os Modelos de Linguagem de Grande Escala (LLMs), como os desenvolvidos pela *OpenAI*, que impulsionam a geração de respostas precisas e contextualizadas. Além disso, serão abordados o paradigma Geração Aumentada de Recuperação (RAG), frameworks como o *LangChain*, que facilitam a integração entre modelos de linguagem e ferramentas de recuperação, e tecnologias como o *Facebook AI Similarity Search (FAISS)* para armazenamento e busca vetorial. Por fim, o referencial apresenta o modelo *DeepSeek*, destacando sua aplicação prática no contexto acadêmico e sua relevância na solução de problemas específicos relacionados à recuperação de informações e o *Streamlit* que vai transformar o código Python em uma aplicação web interativa.

2.1. *Python*

O Python é uma linguagem de programação de alto nível, interpretada e de propósito geral, conhecida por sua sintaxe clara e legível, o que a torna ideal tanto para iniciantes quanto para desenvolvedores experientes. Criada por Guido van Rossum e lançada em 1991, o Python se destaca por sua filosofia de priorizar a simplicidade e a produtividade, encapsulada no mantra *"Zen of Python": "Beautiful is better than ugly. Explicit is better than implicit. Simple is better than complex"* [Foundation 2025].

Uma das principais características do Python é sua versatilidade. Ele é amplamente utilizado em diversas áreas, como desenvolvimento web, automação de tarefas, análise de dados, inteligência artificial, machine learning e computação científica. Sua vasta biblioteca padrão e ecossistema de pacotes de terceiros, como NumPy, Pandas, TensorFlow e Django, permitem que desenvolvedores implementem soluções complexas com poucas linhas de código [Oracle 2025].

2.2. Processamento de Linguagem Natural (PLN)

De acordo com a *International Business Machines Corporation*, o processamento de linguagem natural é uma área interdisciplinar que integra inteligência artificial, linguística computacional e aprendizado de máquina para possibilitar que computadores compreendam e interajam com a linguagem humana de maneira eficiente [IBM 2024].

Para a *International Organization for Standardization* o PLN é a força motriz por trás de muitas das tecnologias que usamos em nosso dia a dia, desde assistentes virtuais como Siri e Alexa até ferramentas de tradução de idiomas e a crescente precisão do texto preditivo. Em essência, ele permite que os computadores entendam os humanos – e falem como eles [ISO 2025].

As abordagens do PLN podem ser classificadas em três categorias principais: simbólica, estatística e baseada em redes neurais. A abordagem simbólica envolve a criação de um conjunto de regras ou padrões que podem ser usados para analisar e gerar dados de linguagem [Elastic 2025].

A abordagem estatística utiliza métodos de aprendizado de máquina para analisar grandes volumes de dados textuais, identificando padrões e tendências na linguagem. Por fim, a abordagem baseada em redes neurais, especialmente com o uso de deep learning, permite que os sistemas aprendam representações complexas da linguagem, capturando nuances e contextos de forma mais eficaz [IBM 2024].

O Processamento de Linguagem Natural atua como um intermediário na interação entre máquinas e seres humanos, conforme ilustrado na Figura 1.

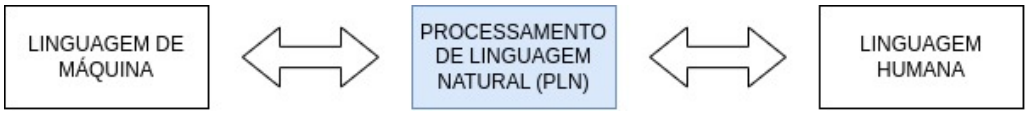

Figura 1. Interação do PLN.
Fonte: Autoria própria.

A Figura 1 ilustra que o PLN traduz a linguagem humana para que as máquinas possam entender, e posteriormente traduz a resposta da máquina de volta para linguagem humana, possibilitando uma comunicação eficaz entre humanos e sistemas inteligentes.

2.3. Modelos de Linguagem de Grande Escala (LLMs)

Os Modelos de Linguagem de Grande Escala (LLMs) possuem alta capacidade de processamento e compreensão de texto natural, sendo aplicados em diversas tarefas de Processamento de Linguagem Natural (PLN), como tradução, sumarização e recuperação de informações [Naveed et al. 2024]. Eles geram sequências de palavras com base em distribuições estatísticas de um vasto corpus textual, mas sem um real entendimento do mundo [Shanahan 2024].

Desde o lançamento do GPT-1 em 2018, os LLMs evoluíram significativamente, incorporando técnicas como aprendizado supervisionado e ajuste baseado em feedback humano, tornando-os mais eficientes e precisos [Yigci et al. 2024]. Modelos como o GPT (Generative Pre-trained Transformer) são amplamente utilizados em chatbots generativos, oferecendo respostas contextuais e personalizadas. Sua arquitetura inclui aprendizado profundo, pré-treinamento e ajuste fino para tarefas específicas [Yigci et al. 2024].

Os LLMs revolucionaram o PLN, com marcos importantes como GPT-2 (2019), GPT-3 (2020) e suas versões aprimoradas, como GPT-3.5 e GPT-4, capazes de lidar com tarefas complexas. Além deles, modelos como BERT (Google), PaLM e LLaMA ampliaram as possibilidades do aprendizado de máquina na linguagem humana [Dam et al. 2024]. A Figura 2 ilustra essa evolução.

Os LLMs se dividem em três categorias principais: modelos pré-treinados, ajuste fino e modelos multimodais [Parthasarathy et al. 2024]. Modelos Pré-Treinados (PLMs) aprendem distribuições probabilísticas sobre sequências de texto de forma não supervisionada, capturando padrões linguísticos e conhecimento factual. Após esse pré-treinamento, podem ser ajustados para tarefas como tradução e classificação [Wei et al. 2024].

A arquitetura Transformer, amplamente utilizada nesses modelos, inclui codificadores e/ou decodificadores para processar e gerar texto. Com o aumento da escala para centenas de bilhões de parâmetros, esses modelos demonstram melhorias significativas em PLN [Rae et al. 2022].

Os modelos multimodais avançam ao integrar diferentes formas de dados, como texto, imagens e áudio [Stryker 2024]. Eles podem descrever imagens em linguagem natural ou gerar imagens a partir de descrições textuais, ampliando aplicações em tradução automática, reconhecimento de fala e análise de sentimentos [Wu et al. 2023].

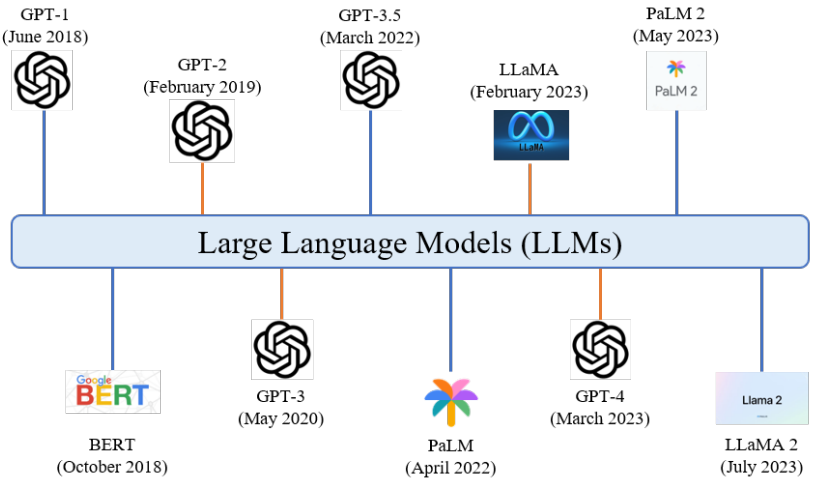

Figura 2. Linha do tempo dos modelos de linguagem.
Fonte: [Dam et al. 2024]

O ajuste fino adapta LLMs pré-treinados para cenários específicos, incorporando conhecimento adicional onde o modelo apresenta lacunas. Isso permite ajustar os formatos de entrada e saída e gerar respostas personalizadas. Métodos como aprendizado por reforço podem ser aplicados para alinhar as saídas do modelo às preferências humanas ou de sistemas de recuperação de dados [Gao et al. 2024]. A Figura 3 ilustra as categorias citadas anteriormente.

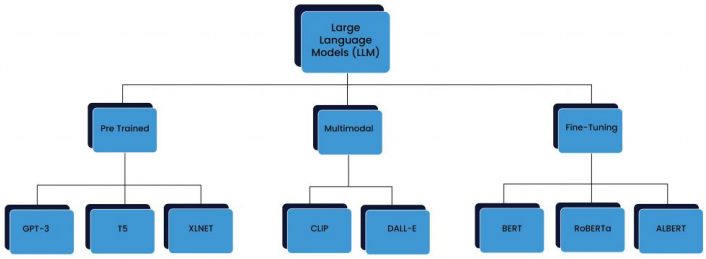

Figura 3. Modelos de Linguagem de Grande Escala.
Fonte: [Data 2024]

A Figura 3 mostra como os LLMs são organizados de acordo com sua função ou estratégia de uso: pré-treinados para tarefas gerais, multimodais para múltiplos formatos de entrada, e ajustados para aplicações específicas via fine-tuning. Na primeira categoria, *Pre-Trained*, estão modelos amplamente utilizados como o GPT-3, o T5 e o XLNet. Já a categoria *Multimodal* inclui modelos como CLIP e DALL-E. Por fim, na categoria *Fine-Tuning*, encontram-se modelos como BERT, RoBERTa e ALBERT.

2.4. Geração Aumentada de Recuperação (RAG)

A Geração Aumentada por Recuperação (RAG) é uma abordagem de inteligência artificial que integra sistemas de recuperação de informações com modelos de linguagem generativa de grande escala (LLMs). Essa combinação permite respostas mais precisas, relevantes e atualizadas, unindo o acesso a dados externos ao processamento avançado

de linguagem natural [Cloud 2025]. De acordo com [LangChain 2025b], essa abordagem permite que modelos de IA acessem bases de dados externas, recuperando informações relevantes antes de gerar uma resposta. Isso reduz a necessidade de reconfigurar os modelos para cada novo domínio ou contexto, mantendo a eficiência e a precisão.

Para um melhor entendimento, é pertinente referenciar os dois modelos destacados pela [AI 2024], sendo eles: modelos de recuperação e modelos generativos. Esses modelos possuem funções complementares que fundamentam a abordagem da geração aumentada por recuperação. A Figura 4 ilustra os conceitos fundamentais, a recuperação e geração citadas anteriormente.

- **Modelos de Recuperação:** são responsáveis por localizar informações relevantes em um conjunto de documentos ou base de conhecimento. Eles utilizam técnicas como recuperação de informações e pesquisa semântica para encontrar os dados mais pertinentes a uma consulta. Embora sejam eficazes em identificar informações específicas, não possuem a capacidade de criar conteúdos novos;
- **Modelos Generativos:** por outro lado, são projetados para produzir textos criativos e coerentes a partir de um contexto ou prompt. Utilizando grandes volumes de dados de treinamento, esses modelos aprendem padrões e estruturas da linguagem natural. No entanto, podem apresentar limitações quanto à precisão factual ou à adequação ao contexto da consulta.

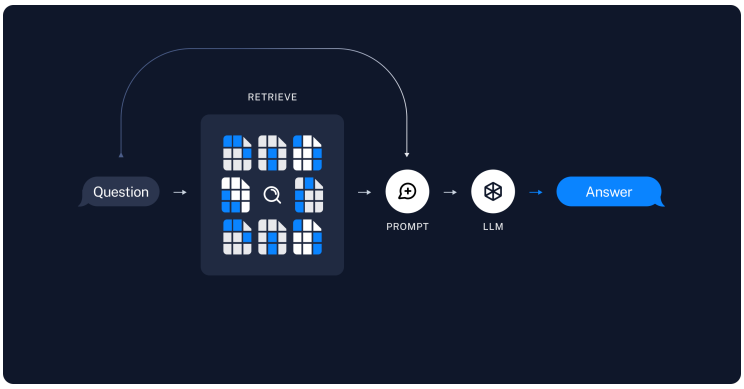

Figura 4. Recuperação e Geração.
Fonte: [LangChain 2025b]

A Figura 4 representa o fluxo da arquitetura RAG, utilizada no desenvolvimento deste trabalho. O processo inicia-se com uma pergunta feita pelo usuário, em linguagem natural. Em seguida, o sistema realiza uma etapa de *retrieval*, ou seja, busca por trechos relevantes em uma base de documentos previamente indexada. Esses trechos, selecionados com base em similaridade semântica, são combinados com a pergunta original para formar um *prompt* enriquecido.

A RAG é especialmente vantajosa para cenários que exigem dados atualizados ou específicos de um domínio. Conforme explicado pela [AWS 2025], essa técnica alinha modelos de linguagem com dados externos, garantindo que as respostas sejam baseadas em informações mais recentes e contextuais, mesmo em situações de rápida mudança. Isso evita a limitação dos modelos de linguagem baseados apenas no treinamento prévio.

Outro aspecto importante do RAG é sua eficiência em termos de custo e recursos computacionais. Ao integrar diretamente mecanismos de recuperação, elimina-se a necessidade de treinamento contínuo dos modelos. A [LangChain 2025a] destaca que essa abordagem é ideal para sistemas empresariais e aplicações dinâmicas, pois permite escalar soluções de IA sem comprometer a relevância ou a precisão das respostas.

Por fim, a RAG se adapta a diferentes casos de uso, como FAQs dinâmicas, assistentes virtuais e chatbots em tempo real. Ao utilizar informações recuperadas em tempo real, a técnica amplia as capacidades dos modelos de linguagem em fornecer respostas detalhadas e contextualmente adequadas, promovendo uma interação mais natural com os usuários [LangChain 2025e].

2.5. *LangChain*

Lançado por Harrison Chase em outubro de 2022, o LangChain é um framework de código aberto projetado para facilitar o desenvolvimento de aplicações que utilizam Modelos de Linguagem de Grande Escala (LLMs). Disponível para as linguagens de programação Python e JavaScript, o LangChain simplifica a criação de aplicações como chatbots e agentes virtuais, fornecendo ferramentas e APIs que auxiliam na integração e orquestração desses modelos em diversos contextos [IBM 2023].

O LangChain oferece facilidades significativas na construção de um sistema baseado em Modelos de Linguagem de Grande Escala (LLMs). Para [Lima et al. 2024] a utilização do LangChain é essencial quando é demandado agilidade e precisão para as tomadas de decisão. O framework simplifica a integração de LLMs com fontes de dados externas, como bancos de dados, APIs e documentos, por meio de sua arquitetura modular e APIs bem documentadas. Essa modularidade permite que desenvolvedores utilizem apenas os componentes necessários, reduzindo a complexidade do código e acelerando o tempo de desenvolvimento [Leung 2023]. Além disso, o LangChain facilita a criação de fluxos de trabalho complexos, como cadeias de tarefas (chains) e agentes autônomos, que podem ser configurados para realizar inferências, pré-processamento e pós-processamento de dados de forma coordenada e eficiente [Jiang and Zhang 2023]. O LangChain se destaca por sua flexibilidade e extensibilidade, permitindo que atualizações e correções sejam feitas de forma ágil. Essa capacidade de adaptação é crucial para aplicações que precisam evoluir com novas demandas, garantindo que sistemas baseados em LLMs permaneçam precisos, atualizados e escaláveis ao longo do tempo. Para uma melhor compreensão, as principais características do LangChain destacadas por [LangChain 2025c], incluem:

- **Modularidade:** projetado como uma biblioteca modular, permitindo que os desenvolvedores utilizem apenas os componentes necessários. Ele fornece ferramentas e funções específicas que podem ser integradas facilmente em diferentes partes de uma aplicação;
- **Integração com Modelos de Linguagem:** a biblioteca facilita o uso de LLMs em fluxos de trabalho, seja para processamento de linguagem natural, geração de texto ou tarefas específicas, como sumarização e classificação;
- **Interação com Dados Externos:** suporta a integração com fontes de dados externas, como bancos de dados, APIs e documentos armazenados localmente ou em nuvem. Isso permite que os modelos de linguagem acessem informações em tempo real e forneçam respostas contextuais atualizadas;

- **Criação de Agentes:** oferece suporte à construção de agentes que podem tomar decisões autônomas, gerenciar fluxos de trabalho e realizar ações específicas com base nas entradas fornecidas pelo usuário;
- **Suporte para Cadeia de Tarefas:** permite a criação de ”cadeias”que combinam diferentes etapas de processamento. Isso é útil para orquestrar fluxos de trabalho complexos, como pré-processamento de dados, inferências e pós-processamento;
- **Foco em Ferramentas de LLM:** fornece utilitários especializados para explorar o potencial dos modelos de linguagem, incluindo ferramentas para ajuste fino, avaliação de desempenho e depuração;
- **Flexibilidade e Extensibilidade:** é possível personalizar as funcionalidades do LangChain, criando componentes específicos para suas aplicações ou ajustando os existentes.

Além disso, o LangChain oferece suporte para a integração com diversas fontes de dados e APIs externas, ampliando as capacidades dos LLMs ao permitir o acesso a informações atualizadas e específicas de diferentes domínios. Essa flexibilidade é essencial para o desenvolvimento de aplicações que necessitam de dados em tempo real ou que precisam interagir com outros sistemas e serviços [LangChain 2025d].

Resumidamente, o LangChain se destaca como uma ferramenta poderosa para desenvolvedores que buscam integrar LLMs em suas aplicações, oferecendo recursos robustos de orquestração, integração e gerenciamento de fluxos de trabalho complexos.

2.6. *Facebook AI Similarity Search (FAISS)*

O *Faiss (Facebook AI Similarity Search)* é uma biblioteca de código aberto desenvolvida pela Meta (anteriormente Facebook) para realizar buscas rápidas e eficientes em vetores de alta dimensão. Amplamente utilizado em aprendizado de máquina e recuperação de informações, o Faiss oferece soluções para buscas de similaridade e recuperação de dados vetoriais, atendendo demandas que vão desde recomendações personalizadas até classificação de imagens [Meta 2025].

Uma das principais características do Faiss é o foco no desempenho e na escalabilidade. A biblioteca foi projetada para processar bilhões de vetores de forma eficiente, utilizando técnicas como compressão vetorial e quantização para otimizar a busca e reduzir os custos de armazenamento sem sacrificar a precisão. Além disso, ela suporta tanto processadores GPU quanto CPU, permitindo sua utilização em diferentes tipos de infraestrutura [GitHub 2025].

A modularidade do Faiss facilita sua integração com frameworks de aprendizado de máquina, como PyTorch e TensorFlow, além de ferramentas de pipelines de IA, como LangChain. Essa flexibilidade tem contribuído para sua ampla adoção tanto pela comunidade científica quanto pela indústria, em áreas como visão computacional, processamento de linguagem natural e sistemas de recomendação [GitHub 2025].

Portanto, o Faiss se destaca como uma solução eficiente para buscar informações relevantes em grandes volumes de dados vetoriais, sendo uma peça central em sistemas que utilizam vetores como base para suas operações de recuperação e resposta.

2.7. *DeepSeek*

A *DeepSeek* é uma empresa que desenvolve inteligência artificial localizada em Hangzhou, China e a sua principal área de atuação é o desenvolvimento de modelos de linguagem de grande escala (LLMs) de código aberto [Kerner 2025]. O modelo foi projetado para competir com soluções de IA avançadas, como o GPT-4 da OpenAI, destacando-se por sua eficiência e custo reduzido de treinamento [Uren and Walsh 2025].

O *DeepSeek* utiliza uma arquitetura de *Mixture of Experts (MoE)*, que divide tarefas entre "especialistas"específicos, permitindo maior eficiência computacional e desempenho superior em tarefas complexas, como raciocínio lógico, geração de código e compreensão de linguagem natural. Essa abordagem é amplamente reconhecida na literatura de IA por sua capacidade de escalar modelos com bilhões de parâmetros sem aumentar proporcionalmente o custo computacional [Shazeer et al. 2017]. Com 671 bilhões de parâmetros, o modelo supera concorrentes em benchmarks de inteligência artificial, especialmente em áreas como matemática e programação [Uren and Walsh 2025].

2.8. *Streamlit*

O *Streamlit* é um framework em Python que simplifica a criação de aplicações web interativas, especialmente para ciência de dados e *machine learning*. Seus principais conceitos destacados por [Inc. 2025] incluem:

- **Scripts como Aplicações:** um script Python é transformado diretamente em uma aplicação web. Cada vez que o usuário interage com a interface (por exemplo, clicando em um botão ou ajustando um slider), o script é executado novamente, atualizando a aplicação em tempo real.
- **Widgets Interativos:** oferece uma variedade de widgets prontos para uso, como sliders, caixas de seleção, botões e campos de texto. Esses widgets permitem que os usuários interajam com os dados e ajustem parâmetros dinamicamente. Por exemplo, um slider pode ser usado para filtrar um conjunto de dados ou ajustar hiperparâmetros de um modelo de machine learning.
- **Caching para Desempenho:** para otimizar o desempenho, o *Streamlit* permite o uso de caching com o decorador `(st.cache)`. Isso é especialmente útil para operações demoradas, como o carregamento de grandes conjuntos de dados ou o treinamento de modelos, evitando que essas tarefas sejam repetidas desnecessariamente.
- **Layouts Flexíveis:** o streamlit oferece opções para organizar o layout da aplicação, como colunas `(st.columns)`, barras laterais `(st.sidebar)` e expansões `(st.expander)`. Isso permite criar interfaces mais organizadas e intuitivas.
- **Gráficos e Visualizações:** o framework suporta a integração com bibliotecas de visualização de dados, como Matplotlib, Plotly e Altair, permitindo a criação de gráficos interativos diretamente na aplicação.
- **Estado e Sessão:** o gerenciamento do estado da aplicação ocorre automaticamente, mas também permite o uso de sessões `(st.session_state)` para armazenar informações entre interações do usuário. Isso é útil para criar aplicações mais complexas que dependem de estados persistentes.
- **Deploy Fácil:** o streamlit facilita o deploy de aplicações, permitindo que elas sejam compartilhadas rapidamente com outras pessoas por meio de serviços como o Streamlit Sharing ou plataformas de nuvem.

3. Trabalhos Relacionados

Nesta seção serão apresentados e discutidos trabalhos que tratam do estudo do uso de Langchain, Geração Aumentada de Recuperação (RAG) e Modelos de Linguagem de Grande Escala (LLMs), com o objetivo de comparar metodologias e destacar as contribuições mais relevantes para o desenvolvimento do agente inteligente.

O estudo conduzido por [G and K 2024] aborda o desenvolvimento de um chatbot avançado baseado no conceito de RAG, com o objetivo de fornecer respostas contextuais precisas a partir de documentos em PDF. O problema central discutido é a necessidade de criar agentes conversacionais capazes de gerar respostas coerentes e relevantes ao consultar grandes volumes de dados externos. Para resolver esse desafio, o trabalho utiliza uma combinação de LLMs, como Llama2 e GPT-3.5, com a abordagem RAG, que permite integrar respostas geradas com a recuperação de informações externas. A metodologia envolve várias etapas técnicas, como pré-processamento de dados, segmentação de texto e geração de embeddings, armazenados em bancos de dados vetoriais para facilitar a recuperação eficiente de informações durante a interação com o usuário. Os resultados mostram que a abordagem RAG aumentou significativamente a precisão das respostas geradas pelo chatbot, ao combinar geração de linguagem com a recuperação de informações de documentos em PDF. No entanto, como limitação, o artigo menciona o alto custo computacional da geração de respostas quando grandes volumes de dados são processados, o que foi mitigado pela utilização de técnicas de Reinforcement Learning (RL) para otimizar o uso de tokens de LLMs, reduzindo custos sem comprometer a qualidade das respostas. Esse estudo é similar ao nosso trabalho no uso de RAG e LLMs para processar documentos estruturados, mas difere no foco específico em PDFs em geral, enquanto nosso trabalho prioriza a integração com documentos acadêmicos institucionais.

O estudo realizado por [Wicks 2023] aborda o problema da sobrecarga informacional enfrentada por estudantes ao buscar respostas sobre normas e regulamentos acadêmicos. O desafio principal reside na dificuldade dos alunos em acessar e compreender rapidamente as resoluções do colegiado, o que gera dúvidas frequentes e repetitivas, demandando tempo tanto de professores quanto de coordenadores. Para resolver essa questão, os autores desenvolveram um chatbot alimentado por modelos de linguagem como o GPT-3 e tecnologias de Processamento de Linguagem Natural (PLN), como o ChatOpenAI (responsável pelo papel do LLM) e o Langchain (que desempenha o papel de integração e gerenciamento de fluxos de PLN). O sistema processa documentos em formato PDF contendo as resoluções acadêmicas e, por meio de uma divisão do texto em trechos e da criação de representações vetoriais, o chatbot é capaz de buscar e fornecer respostas contextuais baseadas nesses documentos. A integração das bibliotecas Langchain e ChatOpenAI foi fundamental para a geração das respostas. Os resultados demonstraram que o chatbot, ao ser validado com perguntas reais, alcançou uma média de acerto de 7,6, com algumas respostas sendo idênticas às resoluções do colegiado, enquanto outras falharam em apresentar todas as informações necessárias. A validação revelou uma eficácia geral, mas com margem para melhoria, especialmente em perguntas que exigiam respostas mais complexas. Esse trabalho é particularmente relevante para nossa pesquisa, pois compartilha o objetivo de automatizar respostas a dúvidas acadêmicas. No entanto, nosso estudo avança ao focar na validação específica para um curso (BSI) e ao utilizar métricas como a similaridade de cossenos, para avaliar a precisão semântica e textual das respostas.

A aplicação de chatbots no domínio jurídico também tem ganhado destaque, especialmente para auxiliar na interpretação e no acesso a leis, regulamentos e normas. Um estudo relevante é o de [Ashley 2017], que explora o uso de sistemas baseados em IA para análise de casos jurídicos e interpretação de leis, destacando a capacidade dessas ferramentas em fornecer respostas precisas e contextualizadas. Essa abordagem é particularmente útil em cenários onde a precisão e a confiabilidade das informações são críticas, como no caso de consultas a documentos legais complexos. Embora o foco seja diferente, a abordagem de recuperação e geração de respostas é similar à nossa, especialmente no uso de técnicas de PLN para processar textos complexos e garantir a confiabilidade das informações.

Os estudos analisados demonstram o potencial dos chatbots baseados em RAG e LLMs para melhorar a recuperação e geração de respostas em diversos domínios. O trabalho de [G and K 2024] destaca a eficiência da RAG na extração de informações de documentos em PDF, embora enfrente desafios relacionados ao alto custo computacional. Já [Wicks 2023] aborda a aplicação de chatbots para auxiliar estudantes no acesso a normas acadêmicas, evidenciando as melhorias na automação de respostas, mas ainda com limitações na precisão de algumas respostas mais complexas. E no âmbito jurídico, [Ashley 2017] explora a utilização de inteligência artificial na interpretação de leis, ressaltando a importância da confiabilidade das informações em contextos críticos.

No entanto, após uma análise detalhada, observa-se que há espaço para personalização e refinamento dos modelos em contextos específicos, como o acadêmico. Em nosso trabalho, esse espaço é explorado ao validar o agente inteligente para um curso específico (BSI), permitindo ajustes finos no modelo para atender às necessidades particulares dos estudantes.

4. Materiais e Métodos

Nesta seção, são descritos os procedimentos adotados para a construção e implementação do sistema de consulta automatizada desenvolvido neste estudo. A abordagem metodológica que pode ser observada na Figura 5 inclui a seleção, pré-processamento, transformação, mineração e avaliação dos dados.

O primeiro passo envolveu a seleção dos documentos em formato PDF e a extração dos textos contidos neles. Em seguida, foi realizado o pré-processamento e a transformação dos dados, segmentando-os em blocos menores para facilitar a análise. A etapa seguinte compreendeu a mineração de dados, onde os textos foram convertidos em representações vetoriais utilizando um modelo de embeddings e armazenados em um índice para recuperação eficiente de informações. Por fim, foi integrada uma camada de processamento de linguagem natural para permitir a interação dinâmica com o usuário, que realiza consultas sobre o conteúdo extraído.

Detalhes adicionais sobre cada uma dessas etapas são apresentados nas subseções a seguir, onde são discutidos os processos específicos de cada fase da implementação.

4.1. Seleção de Dados

A primeira etapa do processo envolveu a seleção de documentos relevantes para a análise como o Regulamento Geral da Universidade Federal Rural de Pernambuco (UFRPE) e

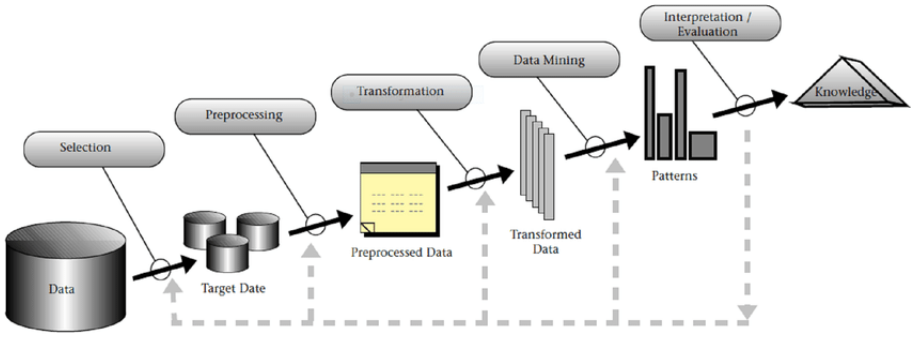

Figura 5. Visão geral das etapas do processo KDD.
Fonte: [Fayyad et al. 1996]

regulamentações de lei do Ensino Superior, que serviram como base de conhecimento para o sistema. Foram utilizados os seguintes arquivos em formato PDF:

- Disciplinas Optativas - Perfil Curricular SIN01.
- Disciplinas Optativas - Perfil Curricular SIN02.
- DECISÃO Nº 66/2017.
- DECRETO No 85.587, DE 29 DE DEZEMBRO DE 1980.
- DECRETO-LEI Nº 1.044, DE 21 DE OUTUBRO DE 1969.
- DECRETO-LEI Nº 715, DE 30 DE JULHO DE 1969.
- LEI 11.788, de 25 de Setembro de 2008 (nova cartilha esclarecedora sobre a lei do estágio).
- LEI No 6.202, DE 17 DE ABRIL DE 1975.
- LEI No 6.503, DE 13 DE DEZEMBRO DE 1977.
- LEI Nº 10.861, DE 14 DE ABRIL DE 2004.
- LEI Nº 13.796, DE 3 DE JANEIRO DE 2019.
- LEI Nº 5.540, de 28 de Novembro de 1968.
- LEI Nº 9.394, de 20 de Dezembro de 1996.
- ORIENTAÇÕES AOS ESTUDANTES SOBRE O ENADE.
- PARECER CNE/CES Nº: 281/2006.
- PARECER CNE/CES Nº: 67/2003.
- RESOLUÇÃO CEPE 431-07 - Regulamento do Diário de Classe - Para Professores.
- RESOLUÇÃO 154.2001 (CEPE - Desligamento de alunos).
- RESOLUÇÃO Nº 2, de 18 de Junho de 2007.
- RESOLUÇÃO Nº 298/2003 do CEPE.
- RESOLUÇÃO Nº 313/2003 (CEPE - Elaboração e Reformulação do PPP).
- RESOLUÇÃO Nº 431/2007 do CEPE.
- RESOLUÇÃO Nº 442/2006 do CEPE.
- RESOLUÇÃO Nº 494/2010.
- RESOLUÇÃO Nº 622/2010 do CEPE.
- RESOLUÇÃO CEPE/UFRPE Nº 526, de 21 de Outubro de 2022 (Regulamento Geral da Graduação).

Esses documentos foram selecionados por sua relevância para o contexto acadêmico, abrangendo leis, resoluções e pareceres que regulamentam a graduação na UFRPE.

4.2. Pré-Processamento dos Dados

Após a seleção, os documentos foram processados para extração e preparação do conteúdo textual. A leitura dos PDFs foi realizada utilizando a biblioteca PyMuPDF (Fitz), que permite a extração de texto de arquivos PDFs de forma eficiente, possibilitando a leitura e conversão do conteúdo das páginas dos documentos em um formato textual adequado para processamento. Essa fase é fundamental, pois permite que o sistema acesse e utilize as informações contidas nos documentos como base para suas consultas e respostas. Dessa forma, o texto foi concatenado em uma única string para facilitar o processamento subsequente. O código utilizado para essa etapa é apresentado no Quadro 1.

Quadro 1. Função get_pdf_text**.**

```
def get_pdf_text(pdf_docs):
    text = ""
    for pdf in pdf_docs:
        doc = fitz.open(pdf)
        for page in doc:
            text += page.get_text("text") + "\n"
    return text
```

4.3. Transformação de Dados

Com o texto extraído, a próxima etapa consistiu em transformar os dados em uma forma adequada para análise. Para isso, foi necessário segmentar o conteúdo dos documentos em blocos menores chamados de *"chuncks"* facilitando o processamento e garantindo a manutenção do contexto. Para isso, foi utilizada a classe *RecursiveCharacterTextSplitter*, um divisor de texto proveniente do *LangChain*, que divide o texto em fragmentos de tamanho controlado, utilizando "n" para quebra de linha, com uma sobreposição entre os blocos. A segmentação foi configurada da seguinte forma:

- Tamanho dos blocos: 800 caracteres.
- Sobreposição entre blocos: 200 caracteres.

A divisão considerou separadores naturais, como quebras de parágrafo (`\n\n`), linhas (`\n`), pontuações (`.`) e espaços ( ), priorizando cortes semânticos coerentes. Essa abordagem permitiu uma preparação eficiente dos dados textuais, viabilizando sua posterior vetorização por meio da geração de *embeddings*, além de garantir compatibilidade com os limites de entrada dos modelos utilizados. O código correspondente à segmentação é apresentado no Quadro 2.

Quadro 2. Função $get\_text\_chuncks$**.**

```
def get_text_chunks(text):
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=800,
        chunk_overlap=200,
        separators=["\n\n", "\n", ".", " "],
        length_function=len
    )
    return text_splitter.split_text(text)
```

4.4. Geração de Embeddings e Recuperação de Informações

Com o texto segmentado, a próxima etapa envolveu a criação de uma representação vetorial dos dados. Para isso, foi utilizado um modelo de *embeddings* baseado na biblioteca *HuggingFace*, que converte cada bloco de texto em um vetor numérico. Esses vetores representam semanticamente o conteúdo dos textos, permitindo que o sistema compreenda o significado contextual das palavras e frases. Essa transformação é fundamental para habilitar buscas eficientes por similaridade, garantindo que o sistema possa recuperar informações relevantes de forma rápida e precisa.

A recuperação das informações relevantes foi facilitada pela utilização do índice *FAISS (Facebook AI Similarity Search)* , que permite uma busca ágil dos vetores gerados. Essa abordagem não apenas proporciona uma representação compacta e eficaz do texto, mas também capacita o sistema a interpretar a semântica subjacente, gerando respostas contextualizadas e precisas às perguntas dos usuários. O código utilizado para essa etapa é apresentado no Quadro 3.

Quadro 3. Função $get\_vectorstore$**.**

```
def get_vectorstore(text_chunks):
    embeddings = HuggingFaceEmbeddings()
    return FAISS.from_texts(texts=text_chunks, embedding=
       embeddings)
```

4.5. Integração com Modelo de Linguagem

Para permitir a interação dinâmica com o usuário e gerar respostas precisas, foi utilizado o modelo de linguagem *ChatOpenAI*, configurado para trabalhar com a API da *OpenAI*. Esse modelo desempenha um papel central na interpretação e resposta às perguntas dos usuários, atuando como um componente especializado em processar e contextualizar as informações.

Ao interagir com o *vectorstore*, que contém as representações vetoriais dos documentos PDF, o sistema não apenas recupera dados relevantes, mas também adapta a conversa ao contexto da interação, gerando respostas fluidas e contextualizadas. Essa abordagem dinâmica e interativa proporciona uma experiência mais natural e eficaz na resolução das dúvidas dos estudantes. O código utilizado para configurar o modelo é apresentado no Quadro 4.

Quadro 4. Função get_llm**.**

```
def get_llm():
    llm = ChatOpenAI(
        model='deepseek-chat',
        openai_api_key="your-api-key",
        openai_api_base='https://api.deepseek.com',
        temperature=0
    )
    return llm
```

4.6. Memória Conversacional

Para que o sistema não se limitasse somente a recuperação de informações, foi implementada uma memória conversacional que tem a capacidade de armazena um histórico de interações com os usuários. O componente *ConversationBufferWindowMemory* foi utilizado para registrar os diálogos anteriores, além de desempenhar um papel importante na manutenção do contexto das conversas. Ao armazenar e recuperar as interações anteriores, o sistema é capaz de gerar respostas mais coesas e contextualizadas, ajustando-se às particularidades de cada conversa. Essa estratégia não apenas aumenta a consistência das respostas ao longo das interações, criando dessa forma, uma experiência de diálogo mais natural e fluida para os usuários. O código responsável pela configuração da memória conversacional pode ser consultado no Quadro 5.

Quadro 5. Função get_chat_memory**.**

```
def get_chat_memory():
    return ConversationBufferWindowMemory(k=5, memory_key="
       chat_history", return_messages=True, output_key="answer")
```

4.7. Engenharia de Prompts

A geração de respostas pelo sistema foi conduzida com base em técnicas de *prompt engineering* (engenharia de prompts), utilizando o framework LangChain. Em vez de submeter diretamente a pergunta ao modelo de linguagem, optou-se por construir um *prompt* estruturado que guia o comportamento do modelo e delimita claramente o escopo da resposta.

Esse prompt foi implementado por meio do componente *ChatPromptTemplate*, fornecido pela biblioteca *LangChain*, o qual permite definir um modelo de instrução parametrizado com variáveis dinâmicas. O objetivo é garantir que a resposta seja elaborada com base exclusivamente nas informações contidas nos documentos institucionais fornecidos como contexto. O Quadro 6 mostra o modelo utilizado.

Quadro 6. Função $get\_prompt\_template$**.**

```
def get_prompt_template():
    prompt_text = """
    You are a university assistant. Coordinators or students
    will ask questions about rules, workload, courses, course
    registration, and other topics related to the Federal Rural
    University of Pernambuco and its programs. Respond
    exclusively based on the provided documents, ignoring any
    irrelevant context.

    Context: {context}
    Question: {question}
    Answer:
    """

    return ChatPromptTemplate.from_template(prompt_text)
```

As chaves {`context`} e {`question`} são preenchidas dinamicamente: o primeiro campo é alimentado com os trechos relevantes recuperados da base de dados por

meio do algoritmo de recuperação semântica, enquanto o segundo campo recebe a pergunta do usuário final.

Esse tipo de abordagem permite o reaproveitamento e a padronização dos prompts ao longo do sistema, além de facilitar a integração com outros módulos de *LangChain*, como *LLMChain* e *RetrievalQA*. O retorno desse prompt segue a estrutura esperada pelo modelo de linguagem, geralmente como um texto contínuo que responde diretamente à pergunta com base no contexto fornecido.

A utilização do *ChatPromptTemplate* demonstrou ser eficaz para manter a consistência das respostas e evitar alucinações, reforçando o papel do contexto documental na ancoragem semântica do modelo.

5. Sistema Desenvolvido

O sistema desenvolvido neste trabalho consiste em um agente inteligente projetado para auxiliar estudantes da UFRPE, especialmente do curso de Bacharelado em Sistemas de Informação (BSI), na obtenção de respostas precisas e contextualizadas sobre normas, regulamentos e procedimentos acadêmicos. O sistema é composto por duas partes principais: o sistema de consulta, responsável pelo processamento e recuperação de informações, e a interface web interativa, que permite a interação direta com os usuários.

5.1. Implementação do Sistema de Consulta

Para facilitar a interação com os usuários, foi implementada no Quadro 7 a classe ChatBotFAQ, que serve como interface entre os usuários e o sistema de consulta. O funcionamento dessa classe se inicia com a leitura dos arquivos PDF da pasta fornecida, onde os documentos são carregados, processados e segmentados. Após a construção dos vetores de similaridade, o sistema é capaz de buscar as respostas mais relevantes para as perguntas feitas. A classe ChatBotFAQ também implementa o histórico de conversas e a recuperação dinâmica de informações. Uma representação do fluxo geral do sistema de consulta pode ser observado no Quadro 6.

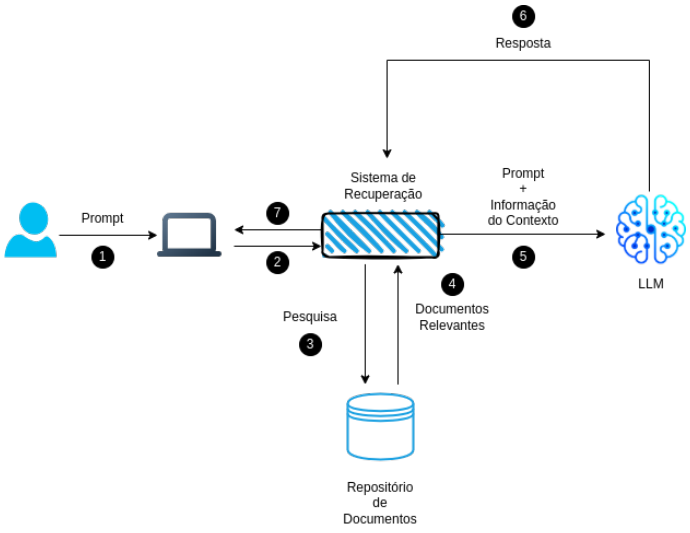

Figura 6. Funcionamento do sistema de consulta.
Fonte: Autoria própria

Com essa estrutura, a classe permite ao sistema realizar a recuperação de contexto dos documentos e gerar respostas de forma automatizada e eficiente. A classe *ChatBotFAQ* foi projetada para carregar, processar e dividir o conteúdo dos PDFs, além de integrar as funcionalidades necessárias para gerar respostas com base no conteúdo extraído. A seguir, mostramos o código central dessa classe.

Quadro 7. Classe $ChatBotFAQ$**.**

```
class ChatBotFAQ:
    def __init__(self, pdf_folder_path):
        pdf_files = load_pdfs_from_folder(pdf_folder_path)
        raw_text = get_pdf_text(pdf_files)
        text_chunks = get_text_chunks(raw_text)
        self.vectorstore = get_vectorstore(text_chunks)

        self.conversation_chain = get_conversation_chain(self.
           vectorstore)
        self.memory = get_chat_memory()

    def ask_question(self, question):
        response = self.conversation_chain({"question": question
           , "chat_history": self.memory.buffer})

        return response["answer"]
```

Essa implementação permite que o sistema processe perguntas de forma contínua, recuperando o contexto adequado e gerando respostas precisas com base no conteúdo dos documentos carregados.

5.2. Implementação da Interface Web Interativa

Para proporcionar uma experiência de usuário interativa e acessível, foi desenvolvida uma interface web utilizando o framework *Streamlit*. Essa interface permite a interação direta com o sistema de consulta, enviando perguntas e recebendo respostas em tempo real. Projetada de forma simples, porém eficiente, a interface garante que o acesso as funcionalidades do Agente Inteligente sem a necessidade de conhecimentos técnicos avançados. A Figura 7 ilustra a interface inicial do sistema, configurada com um título personalizado, ícone de robô e layout amplo para melhorar a usabilidade.

A interface foi construída com componentes como campo de entrada de texto, mensagens de chat e estilos personalizados, visando melhorar a estética e a experiência do usuário. Além disso, ela se integra diretamente à classe *ChatBotFAQ*, responsável pelo processamento das perguntas e geração das respostas. A configuração inicial do *Streamlit*, que define o título da página, o ícone e o layout, pode ser observada no Quadro 8.

Quadro 8. Configuração do Streamlit.

```
st.set_page_config(page_title="Chatbot BSI UFRPE", page_icon="",
    layout="wide")
```

Para melhorar a estética e a usabilidade da interface, foram aplicados estilos CSS personalizados, como pode ser observado no Quadro 9. Esses estilos definem o visual das

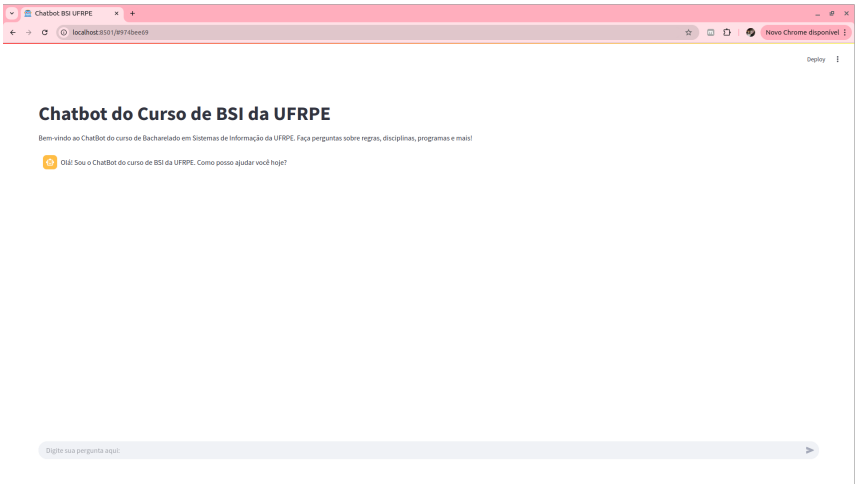

Figura 7. Interface Web.
Fonte: Autoria própria

mensagens do chat, incluindo bordas arredondadas, margens e cores de fundo diferenciadas para as mensagens do usuário e do assistente.

Quadro 9. Estilos Personalizados.

```
st.markdown("""
<style>
    .stChatMessage {
        padding: 10px;
        border-radius: 10px;
        margin-bottom: 10px;
    }
    .stChatMessage.user {
        background-color: #e3f2fd;
        margin-left: 20%;
    }
    .stChatMessage.assistant {
        background-color: #f5f5f5;
        margin-right: 20%;
    }
    .stTextInput input {
        font-size: 16px;
        padding: 10px;
        border-radius: 8px;
        border: 1px solid #ccc;
    }
</style>
""", unsafe_allow_html=True)
```

Além disso, o campo de entrada de texto foi personalizado para ser mais amigável, com tamanho de fonte aumentado e bordas arredondadas. Detalhes que podem ser vistos na Figura 8.

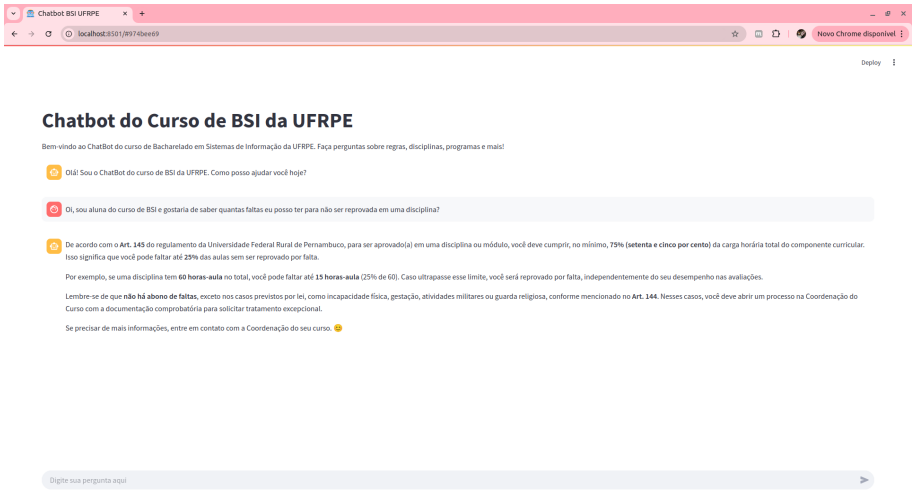

Figura 8. Interface Web.
Fonte: Autoria própria

Em seguida, para carregar corretamente os PDFs, o código apresentado no Quadro 10 verifica se a pasta contendo os arquivos PDF existe no sistema. Caso a pasta não seja encontrada, uma mensagem de erro é exibida, e a execução do aplicativo é interrompida. Essa validação é essencial para garantir que os documentos necessários estejam disponíveis para o funcionamento do agente inteligente.

Quadro 10. Carregamento de PDFs.

```
pdf_folder_path = "/home/embs/Downloads/pasta_pdfs_tcc"
if not os.path.exists(pdf_folder_path):
    st.error(f"Pasta '{pdf_folder_path}' não encontrada.
       Certifique-se de que os PDFs estão na pasta correta.")
    st.stop()
```

Posteriormente, o agente inteligente é inicializado e armazenado no estado da sessão (`st.session_state`). Essa abordagem evita a reinicialização desnecessária do agente inteligente durante a interação do usuário, mantendo a consistência e a eficiência da aplicação, como pode ser observado no Quadro 11.

Quadro 11. Inicialização do Chatbot.

```
if "chatbot" not in st.session_state:
    st.session_state.chatbot = ChatBotFAQ(pdf_folder_path)
```

Para manter o histórico de mensagens na tela, o chat é inicializado com uma mensagem de boas-vindas do assistente, como pode ser mostrado no Quadro 12.

Quadro 12. Histórico de Mensagens.

```
if "messages" not in st.session_state:
    st.session_state.messages = [
        {"role": "assistant", "content": "Olá! Sou o ChatBot do
           curso de BSI da UFRPE. Como posso ajudar você hoje?"}
    ]
```

Esse histórico é mantido durante toda a sessão, permitindo que o usuário visualize o contexto completo da conversa. E para melhorar a interação com os usuários, foi implementada no Quadro 13, que tem a função de diferenciar visualmente as mensagens do usuário e do assistente. Essa diferenciação é feita por meio de estilos CSS, que aplicam cores e margens específicas para cada tipo de mensagem.

Quadro 13. Exibição do Histórico.

```
for message in st.session_state.messages:
    with st.chat_message(message["role"]):
        st.write(message["content"])
```

Foi criado também um campo de entrada de texto (`st.chat_input`) que permite que o usuário digite suas perguntas. Assim que uma pergunta é enviada, ela é adicionada ao histórico de mensagens e exibida no chat. Em seguida, o agente inteligente processa a pergunta e gera uma resposta, que também é adicionada ao histórico e exibida para o usuário. Durante o processamento, um ícone de carregamento (`st.spinner`) é exibido para indicar que a aplicação está trabalhando na geração da resposta.

Quadro 14. Campo de Entrada e Processamento de Perguntas.

```
if prompt := st.chat_input("Digite sua pergunta aqui"):
    st.session_state.messages.append({"role": "user", "content":
        prompt})
    with st.chat_message("user"):
        st.write(prompt)

    with st.spinner("Processando sua pergunta..."):
        answer = st.session_state.chatbot.ask_question(prompt)
        st.session_state.messages.append({"role": "assistant", "
           content": answer})

    with st.chat_message("assistant"):
        st.write(answer)
```

5.3. Validação do Sistema

A validação das respostas do Agente Inteligente é crucial para garantir a precisão e confiabilidade do sistema de transmissão de informações acadêmicas. Esse processo assegura que as respostas estejam corretas, atualizadas e alinhadas com as diretrizes institucionais, atendendo às expectativas dos estudantes.

O método de validação utiliza perguntas retiradas do FAQ oficial do curso de BSI da UFRPE, que contém as dúvidas mais frequentes dos alunos. Cada pergunta é submetida ao Agente Inteligente, e as respostas geradas são comparadas com as informações dos documentos oficiais e com as respostas do FAQ.

Com base nisso, a avaliação das respostas será feita por meio da similaridade dos cossenos, que mede a semelhança entre as respostas do agente inteligente e as respostas ideais. Valores próximos a 1 indicam alta similaridade, enquanto valores próximos a 0 indicam pouca ou nenhuma semelhança. As tabelas de 1 até 5 a seguir são exemplos que detalham os resultados obtidos para cada pergunta avaliada, contendo as respostas do

agente inteligente e do FAQ, além do valor da similaridade. As demais tabelas apresentadas mostram apenas as perguntas e o valor da similaridade.

Tabela 1: Pergunta 1

Pergunta	Quantas faltas o aluno pode ter nas disciplinas para não reprovar por falta?
Resposta do Agente Inteligente	De acordo com o Art. 145 do regulamento da Universidade Federal Rural de Pernambuco, para ser aprovado(a) em uma disciplina ou módulo, você deve cumprir, no mínimo, 75% da carga horária total do componente curricular. Isso significa que você pode faltar até 25% das aulas sem ser reprovado por falta. Por exemplo, se uma disciplina tem 60 horas-aula no total, você pode faltar até 15 horas-aula (25% de 60). Caso ultrapasse esse limite, será reprovado por falta, independentemente do seu desempenho. Não há abono de faltas, exceto nos casos previstos por lei, como incapacidade física, gestação, atividades militares ou guarda religiosa (Art. 144). A LDB (Lei nº 9.394/1996) estabelece o limite máximo de faltas em 25% da carga horária da disciplina. Portanto: 25% de 72 aulas = 18 faltas.
Resposta do FAQ	A LEI nº 9.394/1996 (LDB), Art. 47, §3º, determina que é obrigatória a presença do aluno, salvo em cursos EaD. A frequência mínima de 75% é determinada pela Resolução nº 25/90 do CEPE/UFRPE. Isso representa 45h em uma disciplina de 60h, permitindo no máximo 15h de faltas. Como geralmente essas disciplinas são ministradas em duas horas por dia, o aluno pode faltar até 7 dias (14h). Ao faltar 8 dias (16h), será reprovado.
Similaridade dos Cossenos	0.73

Tabela 2: Pergunta 2

Pergunta	Caso eu reprove por falta, meu ranking/coeficiente de rendimento também será afetado?
Resposta do Agente Inteligente	Sim. Caso você seja reprovado por falta, sua média final será igual à média parcial, conforme o Parágrafo único do Art. 138. Isso impacta diretamente o coeficiente de rendimento, pois essa média é usada para seu cálculo. O Art. 137 define os critérios de aprovação, e se a frequência mínima não for cumprida, o(a) discente será reprovado(a), independentemente do desempenho nas avaliações.
Resposta do FAQ	Sim. Reprovar por falta equivale a reprovar com nota 0 (zero). Portanto, é preferível reprovar com nota maior que zero. O coeficiente de rendimento influencia em programas como mestrado e seleções de graduação.
Similaridade dos Cossenos	0.57

Tabela 3: Pergunta 3

Pergunta	Qual a quantidade de dias letivos de um semestre?
Resposta do Agente Inteligente	Segundo o Art. 120, §2º do regulamento da UFRPE, os períodos letivos regulares têm duração mínima de 18 semanas. Considerando 5 dias úteis por semana, isso resulta em um mínimo de 90 dias letivos. Se o calendário incluir sábados, a quantidade pode variar conforme cada curso.
Resposta do FAQ	A quantidade de dias letivos por semestre pode variar conforme o calendário acadêmico. No entanto, a LDB (Lei nº 9.394/96, Art. 31) exige no mínimo 200 dias por ano, ou 100 dias por semestre. Geralmente, o calendário tem pouco mais de 100 dias para compensar eventuais suspensões. Sábados e pontos facultativos podem ser considerados dias letivos.
Similaridade dos Cossenos	0.72

Tabela 4: Pergunta 4

Pergunta	Qual a diferença entre disciplina optativa e disciplina eletiva?
Resposta do Agente Inteligente	• **Disciplinas Complementares Optativas**: Escolhidas dentro de uma relação aprovada pelo Colegiado do Curso, complementam a formação em determinada área/subárea e contam créditos obrigatórios. • **Disciplinas Eletivas**: Não fazem parte da matriz curricular obrigatória. Podem ser cursadas como suplemento, escolhidas entre disciplinas ofertadas por outras unidades acadêmicas.
Resposta do FAQ	Optativas são escolhidas entre as ofertadas pelo Colegiado do Curso, contribuindo para os créditos exigidos e focando na área de interesse do estudante. Eletivas são aquelas fora do perfil curricular do curso e não contribuem para a integralização.
Similaridade dos Cossenos	0.89

Tabela 5: Pergunta 5

Pergunta	Posso me matricular em quaisquer disciplinas?
Resposta do Agente Inteligente	De acordo com as regras da UFRPE, a matrícula em disciplinas deve observar a sequência estabelecida pela matriz curricular do curso (§ 2º do Art. 2º). Isso significa que você não pode se matricular em quaisquer disciplinas de forma aleatória, mas sim seguir a ordem e os pré-requisitos definidos no currículo do seu curso. Além disso, se você é um aluno de curso de entrada única, fica obrigado a matricular-se nas disciplinas não cursadas nos períodos anteriores no primeiro semestre em que estas forem oferecidas (§ 3º do Art. 2º). Para disciplinas complementares optativas e eletivas, você pode escolher dentro da relação aprovada pelo Colegiado do Curso, mas a matrícula também deve respeitar a sequência curricular e os limites semestrais de matrícula em disciplinas. Portanto, a resposta é: Não, você não pode se matricular em quaisquer disciplinas. A matrícula deve seguir a sequência e os pré-requisitos da matriz curricular do seu curso.
Resposta do FAQ	No caso de disciplinas obrigatórias, recomenda-se que os alunos se matriculem nas disciplinas do curso, mesmo que disciplinas equivalentes sejam oferecidas em outros cursos. No caso de alunos acompanhados, esta matrícula será realizada pelo coordenador do curso diretamente nas disciplinas do curso. Casos excepcionais, em que haja choque de horário entre disciplinas oferecidas no curso, o coordenador poderá realizar a matrícula em disciplinas de outros cursos, para que o aluno não retarde a conclusão do curso. Trabalho não justifica a matrícula em disciplinas de outros cursos/turnos. No caso de disciplinas optativas, há uma maior flexibilização, desde que a disciplina seja prevista na matriz curricular do curso.
Similaridade dos Cossenos	0.82

Tabela 6: Perguntas e similaridade do cosseno

Pergunta	**Similaridade**
Como ficar por dentro do que acontece no curso?	0.4794
Onde encontro as principais informações sobre o curso?	0.6519
Como acessar o wi-fi da UFRPE?	0.2001
Como acessar o AVA?	0.5621
Como faço o meu email institucional?	0.6009
Como solicitar a carteira de estudante?	0.5454
Como faço o cadastro na biblioteca?	0.5981
Como faço o cadastro no restaurante universitário?	0.4999
Como faço minha matrícula?	0.7201
O que devo saber antes de realizar a matrícula?	0.6284

Pergunta	**Similaridade**
Fiz minha matrícula, mas aparece ”Pendente em Fila de Espera”ou ”Indeferido”	0.7042
Como me matriculo em Educação Física?	0.6548
Como realizo o trancamento do semestre?	0.6620
Como realizo o trancamento do semestre fora do prazo?	0.4690
O que é aluno acompanhado?	0.7190
O que é o jubilamento/desligamento?	0.6790
O que é aluno blocado?	0.5951
Qual a prioridade para a matrícula nas disciplinas? Tenho vaga garantida?	0.6991
Qual a diferença entre cancelamento com ônus e sem ônus?	0.7564
Caso eu reprove por falta, meu ranking/coeficiente de rendimento também será afetado?	0.5712
Posso me matricular em quaisquer disciplinas?	0.8168
Posso me matricular em uma disciplina de outro curso?	0.5806
Quantas disciplinas preciso cursar para me formar?	0.6315
Qual é a diferença entre disciplina optativa e disciplina eletiva?	0.8955
Quais são as disciplinas optativas que posso cursar?	0.6547
Como é contabilizada a carga horária de uma disciplina?	0.7394
Podemos usar educação a distância (EaD)?	0.5392
Qual a quantidade de dias letivos de um semestre?	0.7187
Quem pode suspender as aulas?	0.7402
O professor precisa repor aulas?	0.6442
O professor pode repor aulas aos sábados?	0.5735
O professor pode repor aulas em um turno diferente?	0.5173
Quantas faltas o aluno pode ter?	0.7289
Quais casos abonam faltas?	0.7677
Não frequentei as aulas no início do semestre... o professor pode colocar falta?	0.7535
Caso o aluno esteja reprovado por falta, ainda pode fazer as VAs?	0.5960
O aluno precisa realizar as 3 VAs?	0.5419
Qual é o intervalo mínimo entre as VAs?	0.6164
Qual é o período para a realização das VAs?	0.7322
Como solicito a revisão da nota de uma VA?	0.7018
O aluno sempre pode fazer a prova final?	0.5909
Fui inscrito no ENADE. Que procedimentos preciso fazer ANTES?	0.7220
Quem precisa fazer o ENADE?	0.7211
Posso realizar a prova em um local diferente da Universidade?	0.4847
Como sei o local em que irei realizar a prova?	0.7052
Não poderei/pude fazer a prova do ENADE...	0.6933

Pergunta	Similaridade
Qual o problema de estar com situação IRREGULAR junto ao INEP?	0.5199
Estou em situação IRREGULAR junto ao INEP, como me REGULARIZAR?	0.6191
Como o coordenador solicita o acesso ao sistema do ENADE?	0.6328
No BSI o estágio é obrigatório?	0.7192
A partir de que período posso estagiar?	0.7420
Quem deve assinar o termo de estágio?	0.5184
Quais são as restrições para o estágio?	0.7029
Preciso de uma declaração que estou apto a estagiar?	0.6154
Quais são os tipos de atividades complementares...	0.7756
Como incluo as atividades complementares no meu histórico escolar?	0.6941
Como solicito uma declaração de vínculo?	0.2346
Como solicito a dispensa de educação física?	0.7065
Como solicito a dispensa de disciplinas?	0.7837
O que são CCD, NDE e COAA?	0.7590
Quais são as atribuições do coordenador do curso?	0.4491
Como saber mais sobre abreviação de curso e equivalência excepcional...	0.7408
Estou com problemas nos serviços digitais da UFRPE...	0.6755

Tabela 7: Estatísticas descritivas das similaridades do cosseno

Métrica	Valor
Média aritmética	0.6396
Mediana	0.6548
1º Quartil (Q1)	0.5771
2º Quartil (Q2)	0.6548
3º Quartil (Q3)	0.7393
Desvio padrão	0.1207

Tabela 8: Legenda da Similaridade do Cosseno

Valor da Similaridade	Significado
1	Os vetores são idênticos — mesma direção e sentido.
0	Os vetores são ortogonais — não há similaridade entre os textos.
-1	Os vetores são opostos — direção completamente contrária.

6. Discussão

A comparação entre as respostas geradas pelo Agente Inteligente e aquelas presentes no FAQ de BSI permitiu avaliar a capacidade do sistema em produzir respostas semanticamente coerentes com as informações oficiais. Para além dos valores numéricos de similaridade, esta seção se propõe a analisar qualitativamente a performance do modelo, com foco nos casos mais representativos de acerto e divergência.

Um aspecto essencial desta análise é a definição do que constitui uma resposta satisfatória. Para este trabalho, propõe-se a seguinte categorização qualitativa, que serviu como base para as interpretações:

- **Alta similaridade semântica:** quando há correspondência direta de conteúdo, linguagem técnica compatível e estrutura argumentativa equivalente à resposta oficial.
- **Média similaridade semântica:** quando o conteúdo central está presente, mas com lacunas, simplificações ou ausência de fundamentos normativos detalhados.
- **Baixa similaridade semântica:** quando há desvio de escopo, falta de precisão conceitual ou ausência de informações relevantes para o entendimento da questão.

Durante a análise, foi possível observar que as maiores diferenças ocorreram, predominantemente, nos casos de baixa similaridade, que representaram cerca de 11% das perguntas analisadas. Essas discrepâncias estão fortemente concentradas em perguntas de caráter mais prático e operacional, como "Como acessar o wi-fi da UFRPE?" ou "Como faço o cadastro no restaurante universitário?" (ver Tabela 6), cujo conteúdo envolve orientações de uso de sistemas e procedimentos administrativos específicos.

Essas questões, embora pertinentes ao cotidiano acadêmico, normalmente não estão descritas em documentos normativos e regulamentações institucionais, mas sim em páginas informativas, comunicados ou canais auxiliares de atendimento. Como consequência, o modelo não teve acesso a esse tipo de conteúdo durante a etapa de treinamento, o que compromete a qualidade de suas respostas nesses casos. Portanto, a baixa similaridade observada não se deve necessariamente a falhas no processamento ou compreensão por parte do agente inteligente, mas sim à ausência dessas informações específicas na base textual utilizada.

Em contraste, houve também situações em que o agente inteligente demonstrou um desempenho superior ao do FAQ. Em diversas respostas, especialmente aquelas relacionadas a normativas acadêmicas, o Agente Inteligente forneceu explicações mais completas, atualizadas e juridicamente fundamentadas. Um exemplo claro é a pergunta sobre a quantidade de faltas permitidas (ver Tabela 1), em que a resposta do FAQ se restringe a um caso exemplificativo, enquanto o agente inteligente cita diretamente os artigos relevantes da LDB e do regulamento da UFRPE, apresentando o raciocínio legal com base percentual e contextualização mais ampla.

Esse tipo de ocorrência levanta um debate metodológico importante: a métrica de similaridade, embora útil, pode penalizar respostas mais completas ou fundamentadas quando comparadas com respostas oficiais mais objetivas ou simplificadas. Ou seja, uma menor similaridade não implica necessariamente em erro por parte do agente inteligente, pode, em certos casos, indicar um nível maior de profundidade, precisão técnica e contextualização normativa.

Portanto, a avaliação do desempenho do modelo deve ir além dos valores numéricos e considerar a qualidade informacional, o enquadramento jurídico e a utilidade prática das respostas. Essa abordagem crítica permite compreender melhor os acertos e limitações do sistema, além de indicar caminhos para melhorias futuras, como a incorporação de fontes complementares de informação.

7. Resultados

O objeto de análise desta pesquisa compreende a avaliação das respostas geradas por um assistente virtual inteligente, denominado Agente Inteligente, em comparação com as respostas oficiais disponibilizadas em um FAQ do curso de Bacharelado em Sistemas de Informação. O conjunto de dados utilizado para a análise foi composto por 64 perguntas frequentes relacionadas ao funcionamento acadêmico e administrativo da instituição.

O processo de comparação foi realizado por meio do cálculo da similaridade por cosseno entre cada par de respostas, a fornecida pelo agente inteligente e a registrada no FAQ. Essa métrica, amplamente utilizada em tarefas de Recuperação de Informação e Processamento de Linguagem Natural, quantifica o grau de alinhamento semântico entre dois textos vetorizados, retornando um valor entre 0 e 1, onde valores mais próximos de 1 indicam maior semelhança.

O objetivo principal desta etapa consiste em mensurar o desempenho do agente inteligente em termos de aderência ao conteúdo oficial, considerando o quanto suas respostas se aproximam das informações institucionais de referência. Para isso, foram empregadas técnicas estatísticas descritivas aplicadas ao conjunto de similaridades obtidas, a fim de caracterizar o comportamento geral do modelo, identificar padrões de desempenho e detectar eventuais discrepâncias.

A seguir, apresentamos os resultados estatísticos descritivos obtidos com base nos valores de similaridade calculados e posteriormente, uma análise por faixa dos valores observados na Tabela 6.

7.1. Média Aritmética, Mediana e Quartis

A média aritmética foi utilizada para representar a tendência central das similaridades do cosseno. Seu cálculo é dado por:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Substituindo-se os valores das 64 observações:

$$\bar{x} = \frac{40{,}9315}{64} \approx 0{,}6396$$

A mediana, por sua vez, representa o valor central da amostra ordenada. Neste caso, foi calculada como o 32º valor da série ordenada:

$$\text{Mediana (Q2)} = 0{,}6548$$

Os quartis foram obtidos conforme os percentis 25%, 50% e 75%, sendo:

- **1º Quartil (Q1)**: $0{,}5771$
- **2º Quartil (Q2) - Mediana**: $0{,}6548$
- **3º Quartil (Q3)**: $0{,}7393$

Essas medidas mostram que 50% das similaridades ficaram entre $0{,}5771$ e $0{,}7393$, com forte concentração em torno da média. A Figura 9 apresenta a distribuição dessas similaridades em um histograma.

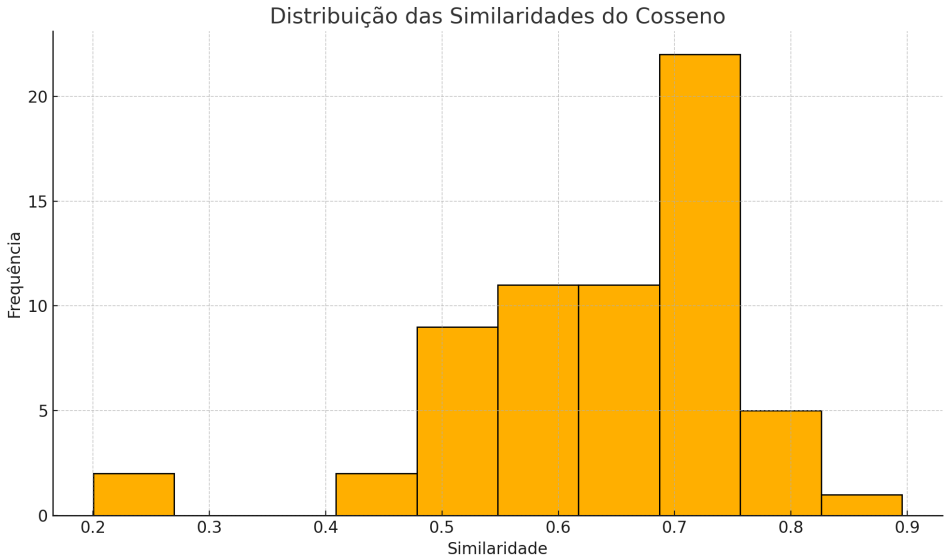

Figura 9. Gráfico de distribuição das Similaridades do Cosseno.
Fonte: Autoria própria.

7.2. Desvio Padrão

O desvio padrão é uma medida de dispersão que indica o quanto os dados se afastam da média. Sua fórmula é expressa por:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

A partir da média $\bar{x} = 0{,}6396$, os desvios de cada valor de similaridade foram elevados ao quadrado, somados e divididos por $n = 64$, resultando em:

$$\sigma = \sqrt{\frac{0{,}9318}{64}} \approx \sqrt{0{,}01456} \approx 0{,}1207$$

O valor do desvio padrão indica que, em média, as similaridades se dispersam cerca de $0{,}12$ unidades em torno da média, o que sugere uma variação moderada no grau de semelhança entre as respostas analisadas. Essa medida é fundamental para compreender o comportamento do modelo além das médias, permitindo identificar o quanto os casos individuais se distanciam do padrão central de desempenho do agente inteligente.

7.3. Análise por Faixas de Similaridade

Para aprofundar a interpretação dos resultados, as 64 respostas analisadas foram classificadas em faixas de similaridade, com o intuito de diferenciar o grau de aproximação semântica entre as respostas do agente inteligente e aquelas fornecidas pelo FAQ institucional.

As faixas foram estabelecidas com base em critérios interpretativos sobre a similaridade do cosseno:

- **Alta similaridade ($\geq$ 0,70)**: respostas praticamente equivalentes, com forte alinhamento conceitual;
- **Média similaridade (0,50 $\leq$ x $<$ 0,70)**: respostas que compartilham o mesmo núcleo semântico, mas podem divergir em termos de completude ou forma;
- **Baixa similaridade ($<$ 0,50)**: respostas com pouco ou nenhum alinhamento semântico perceptível.

A Tabela 9 apresenta a distribuição das perguntas em cada uma dessas faixas, acompanhada de exemplos representativos.

Tabela 9: Distribuição por faixas de similaridade

Faixa de Similaridade	**Qtd. de Perguntas**	**Exemplos**
Baixa ($<$ 0,50)	7	0,48, 0,20, 0,50...
Média (0,50 $\leq$ x $<$ 0,70)	31	0,65, 0,56, 0,60...
Alta ($\geq$ 0,70)	25	0,72, 0,70, 0,72...

Verifica-se que aproximadamente 56% das respostas analisadas apresentaram similaridade igual ou superior a $0{,}70$, o que indica um alto grau de consistência entre as respostas do Agente Inteligente e aquelas contidas no FAQ de BSI. Por outro lado, apenas 7 perguntas obtiveram valores inferiores a $0{,}50$, representando cerca de 11% do total. Esse desempenho inferior, na maioria dos casos, está relacionado a limitações da base de dados utilizada pelo agente inteligente, que foi composta majoritariamente por documentos normativos e regulamentos do ensino superior, deixando de fora conteúdos operacionais e práticos presentes no FAQ.

A classificação por faixas de similaridade reforça a utilidade da métrica como ferramenta para avaliar o desempenho semântico do sistema e oferece subsídios para interpretações mais qualitativas, como as desenvolvidas na seção de Discussão.

De modo geral, os resultados obtidos tanto nas análises estatísticas quanto na categorização por faixas de similaridade corroboram a viabilidade da utilização de sistemas baseados em inteligência artificial como ferramenta complementar de apoio à informação no ambiente acadêmico. A adoção de um agente inteligente institucional pode facilitar o acesso de discentes e demais membros da comunidade universitária a informações relevantes, promovendo maior autonomia, agilidade e acessibilidade na resolução de dúvidas.

8. Considerações Finais e Trabalhos Futuros

Este trabalho apresentou o desenvolvimento e a avaliação de um assistente virtual, o Agente Inteligente, com o objetivo de apoiar os discentes do curso de Bacharelado em

Sistemas de Informação (BSI) da Universidade Federal Rural de Pernambuco (UFRPE) na obtenção de informações acadêmicas. A proposta consiste em oferecer, por meio de uma interface de chat, acesso facilitado a conteúdos frequentemente disponibilizados apenas em documentos institucionais, como regulamentos, normativas e legislações educacionais.

A implementação do sistema envolveu a aplicação de técnicas de Processamento de Linguagem Natural (PLN) e Recuperação da Informação, integradas a uma base de dados composta por documentos normativos oficiais da universidade e do ensino superior. O Agente Inteligente surge, assim, como uma solução tecnológica que visa tornar o acesso à informação mais ágil, contextualizado e acessível aos estudantes.

Durante a avaliação, o desempenho do agente inteligente foi analisado com base na similaridade do cosseno entre suas respostas e as contidas no FAQ institucional do curso. Os resultados obtidos apontaram uma média de similaridade de aproximadamente $0{,}6396$, com mediana de $0{,}6548$ e desvio padrão de $0{,}1207$, indicando consistência nos resultados. A análise por faixas de similaridade revelou que a maioria das respostas encontra-se semanticamente bem alinhada ao conteúdo oficial.

Adicionalmente, foi constatado que, em determinados contextos, o Agente Inteligente produziu respostas mais completas, fundamentadas e tecnicamente mais robustas do que aquelas presentes no próprio FAQ. Por outro lado, os casos de baixa similaridade ocorreram majoritariamente em perguntas de natureza prática e operacional, como instruções de acesso a sistemas ou serviços, que não estavam cobertas pela base de dados utilizada.

Apesar do desempenho satisfatório, o estudo identificou limitações que abrem espaço para melhorias. Entre elas, destacam-se a restrição do escopo da base de conhecimento, que impacta diretamente a cobertura de perguntas do cotidiano discente.

Um ponto que também merece aprofundamento diz respeito à engenharia de prompts. Embora o prompt utilizado neste trabalho tenha seguido boas práticas, com instrução clara, contexto documental e separação da pergunta, investigações futuras podem explorar ajustes mais sofisticados. Isso inclui reformulações na instrução, na organização do contexto, ou até mesmo no estilo da resposta gerada. Avaliar o impacto desses ajustes nos níveis de similaridade pode oferecer insights valiosos sobre o papel da formulação do prompt na performance do sistema.

Durante o desenvolvimento do Agente Inteligente, algumas dificuldades foram enfrentadas, como o ajuste fino do prompt para garantir respostas relevantes, a seleção adequada dos documentos para compor a base de dados e a implementação eficiente da memória conversacional. Esses desafios reforçam a necessidade de estudos futuros voltados à melhoria da engenharia de prompts, à ampliação e curadoria da base de conhecimento, e ao aprimoramento da gestão de contexto durante o diálogo.

Diante disso, propõem-se as seguintes direções para trabalhos futuros.

- **Ampliar a base de conhecimento**, incorporando documentos operacionais, comunicados institucionais, tutoriais e materiais informativos de outros departamentos e setores da universidade.
- **Desenvolver uma interface mais intuitiva e acessível**, com foco na experiência do usuário e nos princípios de acessibilidade digital.

- **Realizar testes em larga escala com estudantes de diferentes cursos**, a fim de avaliar a eficácia do sistema em contextos mais amplos e identificar oportunidades adicionais de melhoria.
- **Explorar técnicas de aprendizado contínuo**, permitindo que o modelo evolua com base nas interações reais dos usuários, adaptando-se progressivamente às demandas mais frequentes.
- **Aprofundar a engenharia de prompts**, buscando compreender o impacto de variações estruturais no desempenho do agente inteligente.

Conclui-se que o Agente Inteligente é uma ferramenta promissora para o contexto acadêmico, com potencial para fortalecer a autonomia estudantil, melhorar a comunicação institucional e contribuir com a democratização do acesso à informação educacional.

9. Repositório do Projeto

O código-fonte completo deste trabalho, incluindo *scripts* de pré-processamento, engenharia de *prompts*, integração com *LangChain* e testes de avaliação, está disponível publicamente no *GitHub*. O repositório pode ser acessado no seguinte endereço:

`https://github.com/EvyMylena/Chabot-BSI`

O conteúdo do repositório pode ser utilizado para fins educacionais e experimentais.

Referências

[AI 2024] AI, T. (2024). O que é rag (retrieval-augmented generation)?

[Alvarenga et al. 2024] Alvarenga, J., Dalepiane, I., Fell, L., and Bernardino, M. (2024). Uma revisão sistemática da literatura sobre chatbots na educação: Desenvolvimento e aplicações. In *Anais do I Workshop sobre Bots na Engenharia de Software*, pages 30–39, Porto Alegre, RS, Brasil. SBC.

[Ashley 2017] Ashley, K. D. (2017). *Artificial Intelligence and Legal Analytics: New Tools for Law Practice in the Digital Age*. Cambridge University Press.

[AWS 2025] AWS (2025). What is rag (retrieval-augmented generation)?

[Bii 2013] Bii, P. (2013). Chatbot technology: A possible means of unlocking student potential to learn how to learn. *Educational Research*.

[Cloud 2025] Cloud, G. (2025). What is retrieval-augmented generation (rag)?

[da Silva 2018] da Silva, R. D. M. (2018). Chatbot para auxílio no ensino e aprendizagem.

[Dam et al. 2024] Dam, S. K., Hong, C. S., Qiao, Y., and Zhang, C. (2024). A complete survey on llm-based ai chatbots.

[Data 2024] Data, S. (2024). Large language models 101: History, evolution and future.

[Elastic 2025] Elastic (2025). O que é o processamento de linguagem natural (pln)?

[Fayyad et al. 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37.

[Foundation 2025] Foundation, P. S. (2025). General python faq.

[G and K 2024] G, A. and K, D. V. (2024). Rag-based chatbot using llms. *International Journal of Scientific Research in Engineering and Management (IJSREM)*, 08:01–04.

[Gao et al. 2024] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., and Wang, H. (2024). Retrieval-augmented generation for large language models: A survey.

[GitHub 2025] GitHub (2025). Faiss.

[Holmes et al. 2019] Holmes, W., Bialik, M., and Fadel, C. (2019). *Artificial Intelligence in Education. Promise and Implications for Teaching and Learning.*

[IBM 2023] IBM (2023). What is langchain?

[IBM 2024] IBM (2024). O que é o pln (processamento de linguagem natural)?

[Inc. 2025] Inc., S. (2025). Basic concepts of streamlit.

[ISO 2025] ISO (2025). Unravelling the secrets of natural language processing.

[Jiang and Zhang 2023] Jiang, J. and Zhang, M. (2023). Overspinning a rotating black hole in semiclassical gravity with type-a trace anomaly. *The European Physical Journal C*, 83(8).

[Kerner 2025] Kerner, S. M. (2025). Deepseek explained: Everything you need to know.

[LangChain 2025a] LangChain (2025a). Build a retrieval augmented generation (rag) app.

[LangChain 2025b] LangChain (2025b). Build a retrieval augmented generation (rag) app: Part 1.

[LangChain 2025c] LangChain (2025c). Introduction.

[LangChain 2025d] LangChain (2025d). Langchain.

[LangChain 2025e] LangChain (2025e). Retrieval augmented generation (rag).

[Larisane et al. 2018] Larisane, N., Vanzin, V., Krassmann, A., Antunes, C., Silva, J. L., and Tarouco, L. (2018). Chatbots na educação: uma revisão sistemática da literatura. *RENOTE*, 16.

[Leung 2023] Leung, V. (2023). Langchain: A framework for llm-powered applications.

[Lima et al. 2024] Lima, F. A. M. d., Boente, A. N. P., Santos, R. M. d., Ferreira, V. M. d. S., Boente, K. P., and Boente, R. M. P. (2024). Modelo de inteligência artificial aplicado à análise de dados de pessoas com deficiência: utilização de langchain. *Revista de Gestão e Secretariado*, page e4281.

[Meta 2025] Meta (2025). Faiss.

[Naveed et al. 2024] Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N., and Mian, A. (2024). A comprehensive overview of large language models.

[Oracle 2025] Oracle (2025). What is python?

[Parthasarathy et al. 2024] Parthasarathy, V. B., Zafar, A., Khan, A., and Shahid, A. (2024). The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities.

[Rae et al. 2022] Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P.-S., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kuncoro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J.-B., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M., Hechtman, B., Weidinger, L., Gabriel, I., Isaac, W., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L., Hassabis, D., Kavukcuoglu, K., and Irving, G. (2022). Scaling language models: Methods, analysis insights from training gopher.

[Shanahan 2024] Shanahan, M. (2024). Talking about large language models. *Commun. ACM*, 67(2):68–79.

[Shazeer et al. 2017] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

[Smutný and Schreiberova 2020] Smutný, P. and Schreiberova, P. (2020). Chatbots for learning: A review of educational chatbots for the facebook messenger. *Computers Education*, 151:103862.

[Stryker 2024] Stryker, C. (2024). O que é ia multimodal?

[Uren and Walsh 2025] Uren, C. and Walsh, D. (2025). What is deepseek, the ai chatbot from china that is sending shockwaves through the tech world?

[Wei et al. 2024] Wei, C., Wang, Y.-C., Wang, B., and Kuo, C.-C. J. (2024). An overview of language models: Recent developments and outlook. *APSIPA Transactions on Signal and Information Processing*, 13(2).

[Wicks 2023] Wicks, G. H. (2023). Chatbot-poli: uma proposta de mvp para automatização de respostas para dúvidas acadêmicas com openai.

[Winkler and Söllner 2018] Winkler, R. and Söllner, M. (2018). Unleashing the potential of chatbots in education: A state-of-the-art analysis. *Academy of Management Proceedings*, 2018:15903.

[Wu et al. 2023] Wu, J., Gan, W., Chen, Z., Wan, S., and Yu, P. S. (2023). Multimodal large language models: A survey.

[Yigci et al. 2024] Yigci, D., Eryilmaz, M., Yetisen, A. K., Tasoglu, S., and Ozcan, A. (2024). Large language model-based chatbots in higher education. *Advanced Intelligent Systems*, n/a(n/a):2400429.