Politécnico
Castelo Branco
Escola Superior
de Tecnologia

AlbiBooks

Chatboot

Ana Margarida Pereira Duarte Nº20200721

Gabriel de Santana Pereira Nº20220165

Orientadores

Professor Doutor Arlindo Ferreira da Silva

Professora Doutora Ana Paula Neves Ferreira da Silva

Trabalho de Projeto apresentado à Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco para cumprimento dos requisitos necessários à obtenção do grau de Licenciados em Informática e Multimédia, realizado sob a orientação científica do Professor Adjunto Doutor Arlindo Ferreira da Silva e coorientação, da Professora Adjunta Doutora Ana Paula Neves Ferreira da Silva, do Instituto Politécnico de Castelo Branco.

fevereiro,2025

Composição do Júri

Presidente do Júri

Doutor, Alexandre José Pereira Duro da Fonte

Professor Adjunto da Escola Superior de Tecnologia de Castelo Branco

Orientador

Doutor, Arlindo Ferreira da Silva

Professor Adjunto da Escola Superior de Tecnologia de Castelo Branco

Arguente

Doutor, Fernando Sérgio Barbosa

Professor Adjunto da Escola Superior de Tecnologia de Castelo Branco

Agradecimentos

Em primeiro lugar, expressamos a nossa sincera gratidão aos nossos orientadores, Professor Arlindo Silva e a Professora Ana Paula Silva pela orientação e pelo apoio ao longo de todo o desenvolvimento deste projeto. A sua disponibilidade foi essencial para o sucesso deste trabalho.

Agradecemos também à Escola Superior de Tecnologia de Castelo Branco que nos acolheu e forneceu os recursos necessários para a realização deste projeto, bem como a todos os colegas e professores que contribuíram direta ou indiretamente para o nosso crescimento académico e profissional. Um agradecimento especial a Professora Ângela Oliveira pelos conselhos fornecidos.

Não poderíamos deixar de manifestar o nosso profundo reconhecimento às nossas famílias e amigos, cujo encorajamento, compreensão e paciência nos acompanharam em cada etapa deste percurso.

Por fim, agradecemos a todos aqueles que, de forma direta ou indireta, contribuíram para a concretização deste projeto.

Resumo

O AlbiBooks é um projeto que visa desenvolver um *chatbot* para interagir com a base de dados de uma biblioteca. O objetivo principal deste *chatbot* é democratizar o acesso aos serviços bibliotecários e torná-lo mais eficiente, permitindo consultas e recomendações de livros de forma prática e intuitiva.

Para fundamentar o desenvolvimento deste projeto, foi realizada uma revisão sistemática da literatura, seguindo a metodologia PRISMA. Esta revisão centrou-se na aplicação de *chatbots* em bibliotecas.

O projeto utiliza uma variedade de tecnologias e ferramentas modernas, incluindo o Python como linguagem de programação principal. Além disso, o Google Colab foi selecionado como ambiente de desenvolvimento.

Uma característica central do AlbiBooks é a implementação de um pipeline *Retrieval-Augmented Generation* (RAG). Para avaliar a eficácia da abordagem proposta, foram testados três modelos de linguagem diferentes: DistilGPT2, Llama-2-13b-chat-hf e Zephyr-7b-beta. Cada modelo foi escolhido pelas suas características específicas, de modo a atender às necessidades do projeto em termos de eficiência computacional e desempenho em tarefas de compreensão e geração de texto.

Este projeto não só responde às necessidades atuais das bibliotecas, como também abre caminho a soluções tecnológicas escaláveis e adaptáveis a diferentes contextos institucionais, prometendo transformar a interação entre utilizadores e serviços bibliotecários num futuro próximo.

Palavras-chave

Chatbot; Inteligência Artificial; *Retrieval-Augmented Generation* (RAG); *Large Language Model* (LLM); Biblioteca.

Abstract

AlbiBooks is a project that aims to develop a chatbot to interact with a library's database. The main aim of this chatbot is to democratise and make access to library services more efficient, enabling book consultations and recommendations in a practical and intuitive way.

To support the development of this project, a systematic literature review was carried out using the PRISMA methodology. This review centred on the application of chatbots in libraries.

The project uses a variety of modern technologies and tools, including Python as the main programming language. In addition, Google Colab was selected as the development environment.

A central feature of AlbiBooks is the implementation of a Retrieval-Augmented Generation (RAG) pipeline. To evaluate the effectiveness of the proposed approach, three different language models were tested: DistilGPT2, Llama-2-13b-chat-hf and Zephyr-7b-beta. Each model was chosen for its specific characteristics in order to meet the needs of the project in terms of computational efficiency and performance in text comprehension and generation tasks.

This project not only responds to the current needs of libraries, but also paves the way for technological solutions that are scalable and adaptable to different institutional contexts, promising to transform the interaction between users and library services in the near future.

Keywords

Chatbot; Artificial Intelligence; Retrieval-Augmented Generation; Large Language Model; Libraries.

Índice Geral

Índice de Figuras

Lista de Tabelas

Lista de abreviaturas, siglas e acrónimos

ADN – Ácido Desoxirribonucleico

ANNs – *Artificial Neural Networks*

API – *Application Programming Interface*

BERT – *Bidirectional Encoder Representations from Transformers*

CNNs – *Convolutional neural network*

CPU – *Central Processing Unit*

DL – *Deep Learning*

DPO – *Direct Preference Optimization*

FAISS – *Facebook AI Similarity Search*

GPT – *Generative Pre-trained Transformer*

GPU – *Graphics Processing Unit*

IA – Inteligência Artificial

IBM – *International Business Machines Corporation*

JSON – *JavaScript Object Notation*

LLM – *Large Language Model*

ML – *Machine Learning*

MLPs – *Multi-layer Perceptrons*

NLP – *Natural Language Processing*

NLU – *Natural Language Understanding*

PE – *Perceived Enjoyment*

PEOU – *Perceived Ease of Use*

PU – *Perceived Usefulness*

PRISMA – *Preferred Reporting Items for Systematic Reviews and Meta-Analyses*

RAG – *Retrieval-Augmented Generation*

RAM – *Random Access Memory*

RL – *Reinforcement Learning*

RNNs –*Recurrent Neural Networks*

TAM – *Technology Acceptance Model*

TPU – *Tensor Processing Units*

VR – *Virtual reality*

VS Code – *Visual Studio Code*

WCAG – *Web Content Accessibility Guidelines*

1.Introdução

As bibliotecas são centros de conhecimento fundamentais, mas a forma como os utilizadores interagem com os seus acervos tem permanecido, em muitos casos, pouco intuitiva e limitada. A necessidade de métodos mais eficientes de pesquisa e recomendação de livros torna-se cada vez mais evidente num mundo onde a informação está disponível instantaneamente em diversas plataformas digitais. Neste cenário, a Inteligência Artificial (IA) surge como um elemento transformador, possibilitando a criação de sistemas inteligentes que melhoram significativamente a experiência dos utilizadores.

O AlbiBooks é um projeto que explora essa transformação, desenvolvendo um chatbot baseado em IA capaz de interagir com bases de dados bibliográficas. O objetivo é oferecer aos utilizadores uma forma simples e natural de pesquisar livros, obter recomendações personalizadas e aceder a informações relevantes de uma biblioteca. Através de *Retrieval-Augmented Generation* (RAG) e do uso de Large Language Models (LLMs), o AlbiBooks não se limita a responder a perguntas diretas, mas também procura compreender o contexto das consultas para fornecer respostas mais relevantes e precisas.

O desenvolvimento do AlbiBooks combina uma revisão sistemática da literatura, com base na metodologia PRISMA, e a implementação de ferramentas tecnológicas modernas, nomeadamente LangChain, *Retrieval-Augmented Generation* (RAG) e *Facebook AI Similarity Search* (FAISS), que possibilitam um processamento otimizado das informações e uma experiência de utilização mais natural.

Além de demonstrar a viabilidade de *chatbots* no contexto das bibliotecas, este estudo analisa os desafios técnicos e conceituais envolvidos, como a integração com bases de dados existentes e a precisão das respostas. O AlbiBooks representa um passo importante na modernização do acesso ao conhecimento, promovendo uma interação mais dinâmica e eficiente entre os utilizadores e as bibliotecas.

1.1. Enquadramento

O projeto proposto consiste no desenvolvimento de um *chatbot* (um *software* baseado em inteligência artificial capaz de manter uma conversa em tempo real por texto) com capacidade para interagir com a base de dados de uma biblioteca. Através de uma interface conversacional, o *chatbot* permitirá que os utilizadores façam consultas sobre o catálogo da biblioteca. Além disso, o *chatbot* poderá sugerir títulos relacionados, com base nas preferências indicadas pelo utilizador.

A motivação principal é oferecer um serviço prático e de fácil acesso, democratizando o uso de bibliotecas e melhorando a eficiência do atendimento e da procura de livros. Este serviço poderá beneficiar estudantes, professores, investigadores e a comunidade em geral, incentivando o uso das bibliotecas e facilitando o acesso ao conhecimento. Também é possível expandir o escopo do projeto para incluir funcionalidades adicionais, como a verificação de informações como disponibilidade de livros, localização dentro da biblioteca e horários de funcionamento.

O projeto proporcionará à biblioteca uma solução tecnológica moderna e eficiente, que pode ser adaptada a qualquer instituição, pública ou privada, que tenha um catálogo digitalizado. Esta abordagem não apenas simplifica a interação com os recursos disponíveis, mas também contribui para a transformação digital das bibliotecas, alinhando-se às necessidades de uma sociedade cada vez mais conectada. A implementação de um *chatbot* inteligente pode ainda reduzir significativamente a carga de trabalho da equipe da biblioteca, permitindo que estes se concentrem em tarefas mais complexas ou estratégicas, como a curadoria de novos materiais e o planejamento de atividades culturais.

Com o potencial de ser um recurso de amplo alcance, o *chatbot* tem como objetivo não apenas melhorar a experiência do utilizador, mas também incentivar a adoção de tecnologias inovadoras por instituições tradicionais, promovendo um equilíbrio entre tradição e modernidade no contexto educacional e informativo.

1.2. Objetivos

O objetivo deste trabalho é verificar a viabilidade do desenvolvimento de um *chatbot* para bibliotecas, utilizando técnicas de *Natural Language Processing* (NLP) para oferecer interações em linguagem comum. Esse *chatbot* deverá ser capaz de proporcionar uma experiência eficiente e amigável aos utilizadores, promovendo a acessibilidade aos serviços da biblioteca e modernizando a sua interação com o público.

A proposta é criar uma solução que sirva como um canal de comunicação entre a biblioteca e os seus utilizadores, permitindo a obtenção de informações relevantes sobre os serviços e recursos disponíveis. O *chatbot* será acessível através de plataformas digitais, proporcionando flexibilidade e conveniência aos utilizadores.

Outro foco importante será a realização de uma integração robusta com a base de dados da biblioteca, garantindo que o sistema seja capaz de oferecer respostas precisas e úteis. Embora a interação principal seja informativa, o desenvolvimento do

chatbot deverá também contemplar uma arquitetura extensível, que permita a adição de futuras funcionalidades, como notificações personalizadas ou recomendações baseadas no histórico de interações.

Por fim, serão conduzidos testes para identificar a combinação ideal de *Large Language Model* (LLM) e avaliar a experiência de uso do *chatbot*, com o objetivo de destacar pontos de melhoria e assegurar que a solução esteja alinhada de maneira eficiente e inovadora às necessidades do público-alvo.

1.3. Planeamento

Este projeto, desenvolvido ao longo do 1.º semestre do ano letivo 2024/2025, centrou-se na fase inicial de investigação e planeamento. O foco principal foi o estudo do estado da arte, a análise de tecnologias e ferramentas relevantes, bem como a definição de um pipeline conceptual para o sistema que se pretende criar. A Tabela **1** apresenta o cronograma das tarefas realizadas, distribuídas em divisões quinzenais dos meses, seguindo uma estrutura organizada e clara.

Tarefas:

- **Tarefa 1 – Estudo do Estado do Arte:** pesquisa e análise de artigos científicos com o objetivo de entender o estado da arte no domínio do projeto.
- **Tarefa 2 – Elaboração do Relatório:** Organização e redação das informações recolhidas, detalhando o estudo da arte.
- **Tarefa 3 – Seleção e Descrição de Tecnologias e Ferramentas:** Seleção e descrição das tecnologias e ferramentas relevantes, incluindo bibliotecas que possam ser utilizadas no desenvolvimento do projeto.
- **Tarefa 4 – Definição do Pipeline Conceptual:** Criação de um pipeline de exemplo que represente, de forma simplificada, as funcionalidades e os objetivos pretendidos para o sistema final.
- **Tarefa 5 – Preparação para a Próxima Etapa:** Planeamento para a seguinte fase do projeto, incluindo a definição de trabalho futuro e possíveis abordagens

Tabela **1**- Cronograma de Tarefas

Tarefa	Setembro 2024		Outubro 2024		Novembro 2024		Dezembro 2024		Janeiro 2025	
	1ºQuin.	2ºQuin.	1ºQuin.	2ºQuin.	1ºQuin.	2ºQuin.	1ºQuin.	2ºQuin.	1ºQuin.	2ºQuin.
Tarefa 1										
Tarefa 2										
Tarefa 3										
Tarefa 4										
Tarefa 5										

2.Inteligência Artificial

O conceito IA surgiu em 1956 na Conferência Dartmouth a qual tinha como objetivo explorar a ideologia de que "toda característica do aprendizado ou qualquer outra característica da inteligência pode, em princípio, ser tão precisamente descrita, que uma máquina pode ser feita para simulá-la" [1].

Daquele momento em diante tentou-se desenvolver maneiras para implementar comportamentos inteligentes nas máquinas, o maior desafio da IA foi resumido com a questão “Como fazer as máquinas compreenderem as coisas?” de Minsky autor do livro “*Semantic Information Processing*” em 1968 [2].

A IA é a tecnologia que permite que as máquinas se ‘comportem’ de forma semelhante aos humanos em termos de aprendizagem, desde a compreensão até à tomada de decisões [3].

Aplicações ou dispositivos que possuem IA podem aprender através de experiências e novas informações, possuem autonomia o que anula a necessidade de intervenção humana. Em 2024, o foco está direcionado na IA generativa, capaz de criar diversos tipos de conteúdo.

2.1. Machine Learning

Machine Learning (ML) é uma área da inteligência artificial cujo objetivo é permitir que as máquinas imitem a capacidade humana de aprender. Desta forma, permite que realizem tarefas de forma autónoma e, que com base na experiência e no aumento dos dados disponíveis, melhorem continuamente o seu desempenho e precisão.

Segundo [4], o sistema de aprendizagem de algoritmos automáticos é composto por três partes principais: (1) Processo de Decisão, os algoritmos são utilizados para fazer previsões ou classificações, tendo em conta dados de entrada que podem ou não ser rotulados; o algoritmo cria uma estimativa sobre os padrões presentes nos dados; (2) Função de Erro, avalia a precisão das previsões e, no caso de existirem exemplos conhecidos, compara os resultados para medir a precisão do modelo; (3) Processo de Otimização do Modelo, um método em que o algoritmo analisa o erro e otimiza o processo de decisão, reduzindo a ocorrência de erros bem com a sua gravidade.

Existem diversos tipos de modelos de ML que são definidos levando em consideração a presença ou ausência de influência humana. De acordo com [5], existem modelos de ML como: (1) Supervisionados, há uma rotulação prévia dos

dados, bem como uma classificação por parte dos utilizadores, permitindo ao algoritmo avaliar a precisão do seu desempenho; (2) Não Supervisionados, os dados são utilizados em bruto, ou seja, sem rotulação prévia. Cabe ao algoritmo identificar padrões nos dados sem intervenção humana; (3) Semi-Supervisionados, os conjuntos de dados utilizados combinam informações estruturadas e não estruturadas, permitindo que o algoritmo alcance conclusões de forma independente, ao mesmo tempo que aprende a rotular dados não rotulados; (4) Por Reforço, neste tipo de modelo, utiliza-se um sistema de “recompensas e punições” para que os algoritmos aprendam através das suas próprias experiências (tentativa e erro), recebendo feedback ao longo do processo.

2.1.1. Deep Learning

O *Deep Learning* (DL), uma subárea do *machine learning*, possibilita que modelos computacionais compostos por várias camadas de processamento aprendam representações de dados a vários níveis de abstração. Estes métodos contribuíram de forma significativa para o avanço no reconhecimento da fala, no reconhecimento visual de objetos, na deteção de objetos e em muitas outras áreas, como a descoberta de medicamentos e a genómica [6]. Um modelo de DL é essencialmente um programa que exibe inteligência graças ao processamento de dados sofisticado. Muitos modelos de IA, como os LLMs, utilizam DL para compreender o contexto e criar respostas realistas.

Por norma, um programa informático requer dados de entrada precisos para gerar saídas corretas. No entanto, o DL consegue processar dados imprecisos e, ainda assim, produzir resultados relevantes. Por exemplo, um programa tradicional pode determinar se dois retratos digitais são exatamente iguais, enquanto um modelo de DL pode identificar semelhanças nos indivíduos retratados, mesmo que as imagens sejam diferentes [7].

Estes modelos dependem de grandes volumes de dados, bem como de um elevado poder computacional. O aumento destes recursos tornou-os mais sofisticados. Resumidamente, o DL é uma especialização do ML. Ambos permitem que o programa aprenda de forma autónoma a partir de um conjunto de dados, mas o DL vai além, sendo capaz, por exemplo, de aprender com dados não rotulados. Estes modelos são sempre construídos com recurso a redes neuronais, ao contrário dos modelos de ML, que nem sempre utilizam esta estrutura.

O DL já possui inúmeras aplicações, e novas utilizações continuam a ser descobertas constantemente, incluindo carros autónomos, criação de imagens, NLP e *Chatbots* de IA conversacional.

2.2. Redes Neuronais

Redes Neuronais são um dos modelos de aprendizagem automática. Estas tomam decisões de forma semelhante ao cérebro humano, imitando o funcionamento dos neurónios para identificar padrões, ponderar opções e chegar a conclusões. São compostas por camadas de nós (neurónios artificiais) e incluem uma camada de entrada, uma ou várias camadas ocultas e uma camada de saída. Os nós estão interligados, e cada um possui um peso e um limiar de ativação. Quando o output de um nó excede o seu limiar, ele é ativado e transmite os dados à camada seguinte; caso contrário, não transmite nada.

Para aprender e aumentar a sua precisão, as redes neuronais dependem de dados de treino. Após serem ajustadas, tornam-se ferramentas poderosas para classificar dados com rapidez. Um exemplo conhecido é o algoritmo de pesquisa da Google [8].

As redes neuronais, também chamadas de *Artificial Neural Networks* (ANNs), podem, segundo [9], ser classificadas em diferentes tipos, dependendo da sua finalidade.

- ***Perceptron***: A forma mais antiga de rede neuronal, criada em 1958 por Frank Rosenblatt;
- ***Multi-layer Perceptrons*** (MLPs): Possuem camadas de entrada, ocultas e de saída. São compostas por neurónios sigmoides, sendo ideais para problemas não lineares. Este tipo de rede serve como base para visão computacional e outros modelos;
- ***Convolutional Neural Network*** (CNNs): Utilizadas para reconhecer imagens e padrões. Baseiam-se em álgebra linear, como multiplicações de matrizes, para identificar padrões;
- ***Recurrent Neural Networks*** (RNNs): Possuem *loops* de *feedback* como principal característica. São úteis para dados temporais, como previsões de séries temporais, incluindo o mercado de ações.

2.2. Inteligência Artificial Generativa

A IA Generativa [10], consegue criar desde textos originais a áudio, embora não seja um conceito atual tendo surgido na década de 1960 ganhou recentemente destaque devido a praticidade das interfaces que permitem criar em segundos conteúdos de alta qualidade incluindo representações realistas de pessoas.

Esta capacidade trouxe benefícios, como dobragens aprimoradas e conteúdos educativos mais detalhados, levantou também preocupações com os *deepfakes* e ataques cibernéticos, que exploram a capacidade de imitar pessoas de forma realista.

Os *transformers*, uma arquitetura essencial para modelos modernos de NLP, foram um dos principais responsáveis pelo avanço da IA generativa, permitindo treinar LLMs sem a necessidade de rotular todos os dados. Este conceito será detalhado no Capítulo **4**.

Os progressos nos LLMs, com bilhões de parâmetros, iniciaram uma nova era, onde é possível gerar textos envolventes, imagens fotorrealistas e até criar conteúdos multimodais que combinam texto, gráficos e vídeo.

2.3. Chatbot

De acordo com [11], um *chatbot* é um programa que simula conversas humanas. Embora nem todos os *chatbots* utilizem inteligência artificial (IA), os mais modernos recorrem a técnicas de IA conversacional, como NLP, para compreender as questões dos utilizadores e automatizar respostas.

Os primeiros *chatbots* foram desenvolvidos em1960 [12]. Por norma, os *chatbots* utilizam conjuntos de regras predefinidas para interagir com os utilizadores. As organizações podem utilizar estes programas para melhorar as suas comunicações, abrangendo desde fluxos de trabalho até ao atendimento ao cliente.

Os *chatbots* facilitam o acesso a informações [11], respondendo instantaneamente quando questionados, seja por texto, áudio ou ambos, sem necessidade de intervenção humana. Por isso é possível identificar diversos benefícios: (1) Respostas instantâneas e acessíveis 24/7; (2) Personalização da experiência do utilizador, incluindo recomendações; (3) Redução de custos e aumento da eficiência operacional, ao libertar recursos humanos de tarefas repetitivas; (4) Automação de fluxos de trabalho complexos, como o agendamento de reuniões ou a recolha de informações detalhadas.

Nas organizações, os *chatbots* são utilizados em tarefas como assistência ao cliente em tempo real, recomendação de produtos ou serviços e automação de lembretes e notificações.

Não obstante as vantagens fornecidas pelos *chatbots*, existem desafios que precisam ser enfrentados. Riscos de segurança e privacidade que podem expor informações confidenciais, especialmente nos modelos de IA generativa, riscos de “alucinações”, casos onde o *chatbot* pode fornecer respostas incorretas ou irrelevantes e a complexidade de integração com os sistemas existentes como as Bases de Dados.

2.4. Chatbots com IA Generativa

Um exemplo notável de um *chatbot* com IA Generativa é o ChatGPT [13], um sistema avançado que não só simula interações humanas, mas também é capaz de produzir diferentes tipos de conteúdo com elevado nível de qualidade. Este tipo de *chatbot* representa um marco na evolução da inteligência artificial aplicada à comunicação, pois permite uma experiência de interação mais natural, intuitiva e eficaz. Os *chatbots* que utilizam IA Generativa destacam-se por oferecerem funcionalidades avançadas, entre as quais se inclui uma compreensão mais profunda e precisa da linguagem natural. Esta capacidade permite-lhes interpretar nuances de significado, contexto e até mesmo o tom das interações [11].

Por outro lado, quando comparados aos *chatbots* tradicionais, a superioridade dos sistemas baseados em IA Generativa torna-se evidente. Enquanto os *chatbots* convencionais dependem exclusivamente de respostas pré-programadas, limitando-se a executar scripts específicos ou a responder a perguntas simples, os *chatbots* com IA Generativa têm a capacidade de criar automaticamente respostas originais. Eles conseguem lidar com perguntas inesperadas e contextos complexos, gerando novos conteúdos com base no seu treino em LLMs. Em vez de se restringirem a simular respostas humanas, estes sistemas utilizam o conhecimento adquirido durante o treino para produzir interações mais criativas e adaptativas. Isso amplia significativamente o seu leque de aplicações e utilidades.

2.5. Large Language Model

Um *Large Language Model* (LLM) é um tipo de modelo de aprendizagem automática concebido para compreender e gerar texto semelhante ao humano através da aprendizagem de padrões estatísticos a partir de grandes quantidades de dados de texto. Estes modelos são utilizados em várias aplicações, incluindo *chatbots* e sistemas de recomendação.

Estes modelos são capazes de processar estatisticamente textos e prever as palavras seguintes numa sequência, através da análise de padrões estatísticos. Estes modelos são versáteis, pois são treinados com um vasto conjunto de dados e podem ser ajustados para tarefas específicas [14].

Exemplos de LLMs incluem o ChatGPT (OpenAi) e o Gemini (Google) [15]. Estes modelos podem ser treinados para desempenhar diversas tarefas, sendo amplamente utilizados com IA generativa. Quando assimilam uma pergunta, são capazes de produzir texto em resposta. Para o seu treino, podem ser utilizados conjuntos de dados extensos e complexos, incluindo linguagens de programação. De facto, alguns destes

modelos fornecem suporte aos programadores na escrita de código, desenvolvendo funções ou completando programas a partir de um ponto inicial.

Os LLMs também podem ser aplicados em áreas como: (1) Análise de sentimentos; (2) Investigação de ADN e (3) Serviço de apoio ao cliente.

Os LLMs são ferramentas transformadoras no *Natural Language Processing* e não só, com aplicações que abrangem vários domínios. A sua capacidade de processar e gerar texto semelhante ao humano, associada à sua adaptabilidade a várias tarefas, torna-os inestimáveis para o avanço da tecnologia e da investigação. No entanto, continua a ser crucial compreender as suas limitações e garantir a sua utilização responsável [16].

2.6. Retrieval-Augmented Generation (RAG)

O conceito de *Retrieval-Augmented Generation* (RAG) surgiu com o objetivo de melhorar os modelos atuais de NLP. Neste sentido, pesquisadores da Meta AI [17] combinaram modelos de linguagem com fontes de conhecimento externas, treinando-os para produzir respostas mais precisas e contextualmente relevantes.

O RAG opera através de quatro etapas principais [18]: (1) Indexação: os dados são convertidos em representações numéricas e armazenados numa base de dados vetorial; (2) Recuperação: ocorre a seleção dos documentos mais relevantes, para uma consulta especifica; (3) Aumento: etapa onde as informações recuperadas são integradas na consulta original; (4) Geração: produção de uma resposta baseada na consulta original e nos dados recuperados;

A otimização ocorre porque a abordagem RAG busca informações em fontes externas, diferentes daquelas usadas durante o treinamento. Isso melhora as respostas e amplia o conhecimento do modelo sem a necessidade de treiná-lo novamente.

Um LLM sem RAG, ao receber informações do utilizador, utiliza os seus dados de treinamento para gerar a resposta. Quando o LLM utiliza RAG, o utilizador faz uma consulta e o modelo busca informações a partir de uma base externa e confiável de conhecimento. Dessa forma, as respostas geradas passam a ser mais embasadas e confiáveis [17].

3. Revisão Sistemática

Neste capítulo, será apresentada uma revisão sistemática da literatura sobre a aplicação de *chatbots* em bibliotecas. O objetivo desta revisão é identificar e analisar os principais estudos, tecnologias e abordagens relacionadas ao uso de *chatbots* neste contexto, tendo como foco os seus benefícios, desafios e tendências. A revisão sistemática é fundamental para garantir que este trabalho se baseie nas pesquisas mais relevantes, proporcionando uma visão abrangente do estado atual da tecnologia e das suas implicações no setor das bibliotecas.

3.1. Metodologia e Processo

Para conduzir a revisão sistemática da literatura sobre o uso de *chatbots* em bibliotecas, foi aplicada a metodologia PRISMA (Preferred Reporting Items for Systematic reviews and Meta-Analyses) [19]. A escolha da metodologia PRISMA justifica-se pela sua estrutura rigorosa, que garante transparência e replicabilidade na seleção e análise dos estudos.

Para a pesquisa de trabalhos e estudos já realizados foram utilizadas as bibliotecas científicas Scopus [20] e ACM [21]. Estas plataformas digitais disponibilizam o acesso a uma vasta coleção de recursos acadêmicos, incluindo artigos científicos, revistas, livros e outros materiais de pesquisa. Para a comunidade acadêmica, oferecem acesso a publicações científicas de diversas áreas do conhecimento, facilitando a pesquisa e a obtenção de dados atualizados e relevantes para projetos e estudos acadêmicos.

No contexto deste trabalho, para efetuar a pesquisa foram selecionadas palavras-chave que descrevessem o tema abordado, sendo estas “*chatbot*”, “*library*”, “*machine learning*” e “*artificial intelligence*”.

3.1.1. Questões de Pesquisa

Com o objetivo de orientar o desenvolvimento deste projeto e proporcionar uma visão clara dos resultados esperados, foram formuladas as seguintes questões de pesquisa, que abordam temas específicos e essenciais:

1.Quais são as funcionalidades mais relevantes para que um *chatbot* atenda adequadamente às necessidades dos utilizadores em bibliotecas?

2.Quais tecnologias que incorporam Inteligência Artificial foram utilizadas?

3.Quais são os principais desafios na implementação de *chatbots* para o contexto de bibliotecas?

4.Qual o tipo de interação pessoa-máquina foi utilizada?

3.1.2. Estratégia de Pesquisa

A pesquisa foi efetuada nas bases de dados ACM e Scopus utilizando a expressão algorítmica "*chatbot*" AND "*library*" AND ("*Artificial Intelligence*" OR "*Machine Learning*"). O processo seguiu as etapas de identificação, triagem, elegibilidade e inclusão, com aplicação de critérios de inclusão e exclusão claros para assegurar a relevância dos artigos selecionados. Foram incluídos estudos publicados nos últimos 4 anos, focados no uso de *chatbots* em contextos bibliotecários utilizando IA (Inteligência Artificial), já os estudos com temas distintos foram excluídos. Ferramentas de apoio, como o Mendeley [22], foram utilizadas para organização e extração dos dados, garantindo uma análise precisa dos resultados.

Com a utilização da expressão algorítmica mencionada no campo de pesquisa, foram obtidos 42 resultados na SCOPUS e 76 resultados na ACM. Após a eliminação de 1 documento duplicado, passaram para a próxima fase de triagem 117 artigos.

3.1.3. Critérios de Inclusão

Para garantir que a revisão sistemática aborde apenas estudos relevantes, os seguintes critérios de inclusão foram estabelecidos: (1) estudos que discutem a aplicação de *chatbots* em bibliotecas, sejam estas acadêmicas ou públicas, (2) artigos e estudos publicados desde 2020, garantindo a atualidade dos dados e tecnologias discutidas, (3) trabalhos publicados em inglês.

3.1.4. Critérios de Exclusão

Para manter o foco e a pertinência dos estudos selecionados, os seguintes critérios de exclusão foram aplicados: (1) estudos que não abordem especificamente o contexto de bibliotecas e (2) estudos que não abordem *chatbots* de Inteligência Artificial.

3.1.5. Extração e Análise dos dados

Para aplicar os critérios de inclusão, foram lidos os títulos dos artigos. Após a aplicação dos critérios foram excluídos 101, de um total de 117 artigos. Para a fase de leitura de *abstract* foram, portanto, selecionados 16 artigos. Nesta fase, 2 artigos foram excluídos por falta da integralidade do texto, 1 por não estar relacionado com IA e 4 por não mencionarem nenhum tipo de ferramenta de IA diretamente. Por fim, 9 artigos foram selecionados para leitura e análise detalhada. A Figura **1** detalha o processo de inclusão e exclusão dos artigos.

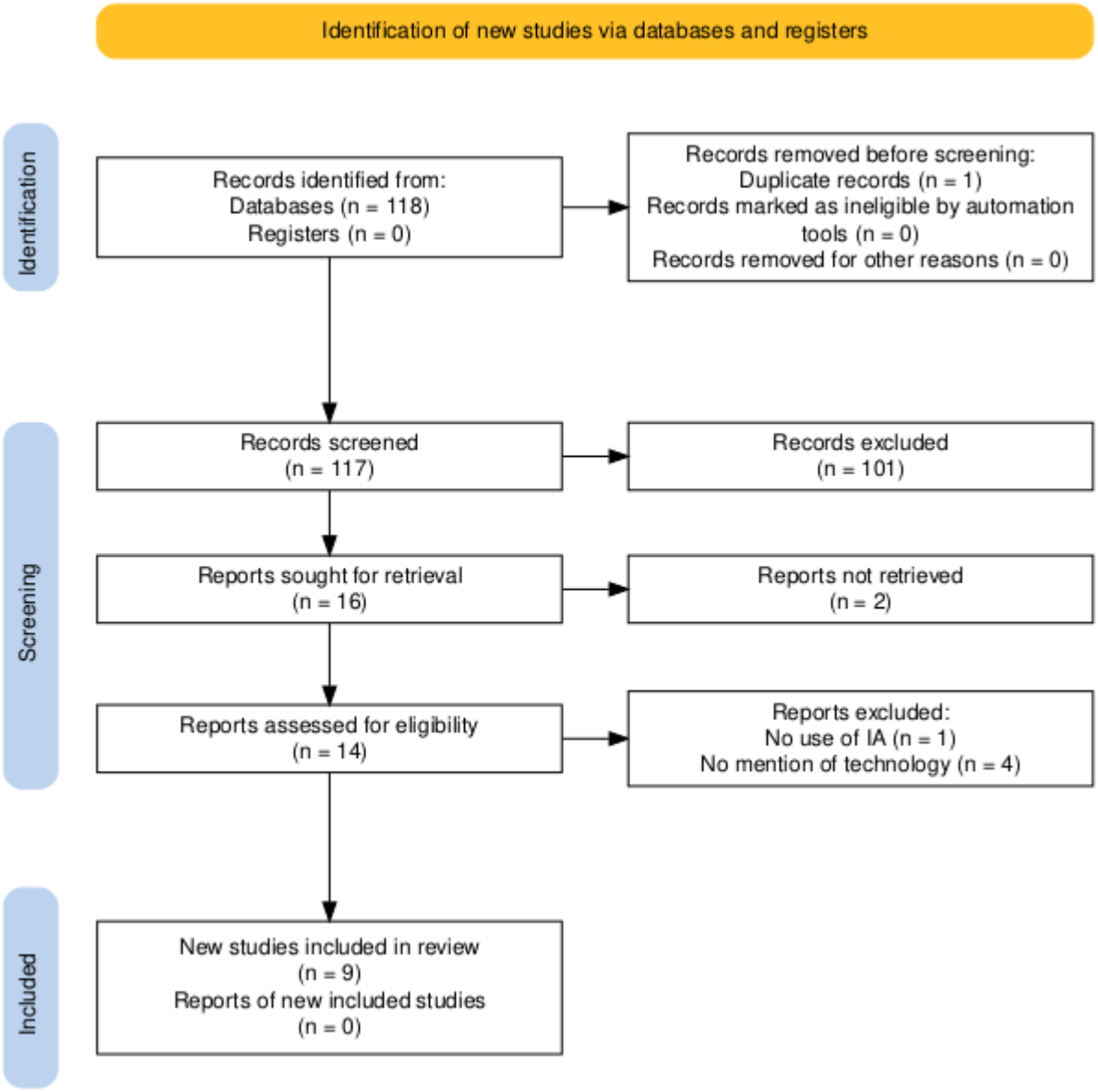

Figura **1** - Fluxograma PRISMA

3.2. Análise dos artigos

A Tabela **2**, apresenta uma análise dos diferentes estudos sobre tecnologias e ferramentas aplicadas a sistemas de *chatbot* e NLP. Os artigos abrangem o período de 2020 a 2024, refletindo a evolução das abordagens adotadas. A seguir, é feita a análise individual de cada artigo selecionado para leitura.

Tabela **2**-Análise Artigos

Artigos	Ano de Publicação	Ferramentas/Tecnologias	Limitações Identificadas	Tipo de Interação
[23]	2024	Dialogflow, Kommunicate	Envolvimento limitado por parte dos utilizadores/ Subjetividade na interpretação	Não menciona/ Texto
[24]	2023	Python/Chroma/LangChain/ API do ChatGPT	Restrições de *Tokens*/Falta de acesso a dados em tempo real/ Alucinações	Website/Texto
[25]	2023	Unicamente teórico/ ChatGPT	Não menciona	Unicamente teórico
[26]	2024	Unicamente teórico/ ChatGPT	Relevância e precisão	Unicamente teórico
[27]	2020	Rasa Stack	Dificuldade com Diálogos Complexos/ Falta de Documentação/ Dependência da Plataforma	Não menciona

Artigos	Ano de Publicação	Ferramentas/Tecnologias	Limitações Identificadas	Tipo de Interação
[28]	2023	Unicamente teórico/ ChatGPT	Referenciação/ Alucinações/Dados desatualizados	Unicamente teórico
[29]	2022	Ivy Software	Licenças Limitadas/ Preocupações de Privacidade/*Crawling* Limitado	Website/Texto
[30]	2024	BERT/Reconhecimento de Voz/ Flutter/Phyton/Flask/MySQL	Falta de um *dataset* amplo para treino/ Ausência de memória de contexto/Falta de suporte a múltiplos idiomas	Mobile/Texto/Voz
[31]	2023	IBM Watson Assistant/ Realidade Virtual/ Unity/ Reconhecimento de fala/Suporte de áudio	Não menciona	VR

O artigo [23] descreve o desenvolvimento de um *chatbot* de IA, destinado aos serviços de referência da Biblioteca Estadual de San José. O desenvolvimento foi efetuado com recurso ao *software* Dialogflow [32] e inclui elementos interativos e *widgets* de chat fornecidos pela plataforma Kommunicate [33]. O *chatbot* foi treinado com um conjunto de perguntas e palavras-chave, utilizando NLP e algoritmos de IA treinados, e foi programado para responder a questões básicas, estando disponível após o horário de funcionamento.

As interações com o *chatbot* foram monitorizadas durante 18 meses após o seu lançamento, e os dados recolhidos foram analisados utilizando as ferramentas analíticas oferecidas pelo Dialogflow e pela Kommunicate. Esta análise permitiu avaliar a eficácia do sistema. Durante o período inicial, foram detetadas apenas 44 interações; no entanto, os resultados demonstram que, ao fim de um ano, o número aumentou para 137 interações mensais. Na grande maioria, essas interações consistiam em perguntas sobre o horário de funcionamento e os recursos da biblioteca. Os autores observaram que apenas 10% dos utilizadores ultrapassaram essas interações básicas com o *chatbot*.

O artigo conclui que, mesmo com recursos limitados, as bibliotecas podem implementar *chatbots* de inteligência artificial com êxito.

No artigo [24], é analisado o projeto AIsha, que, segundo os autores, é uma iniciativa pioneira no contexto de *chatbots* em bibliotecas académicas, baseados na *Application Programming Interface* (API) do ChatGPT. O artigo explora o desenvolvimento do *chatbot*, recorrendo a ferramentas como Python, Chroma, LangChain e a API do ChatGPT, para se integrar nos serviços da Biblioteca da Universidade de Zayed, com o objetivo de criar um sistema funcional adaptado às necessidades dos seus utilizadores. Além disso, são traçados paralelos entre os *chatbots* baseados em regras, que são mais tradicionais, e os modelos generativos, argumentando-se que estes últimos oferecem mais vantagens em termos de flexibilidade e capacidade para lidar com consultas mais complexas. É apresentada uma revisão literária abrangente sobre o uso de *chatbots* em bibliotecas públicas e académicas, desde as primeiras implementações europeias até aos avanços das LLMs, e discute-se o potencial do ChatGPT para melhorar serviços como a busca de informações e o suporte a pesquisas. No entanto, ressalva-se a necessidade de diretrizes claras para o uso responsável da tecnologia. Na Figura **2**, pode observar-se a página principal do Website da Biblioteca da Universidade de Zayed.

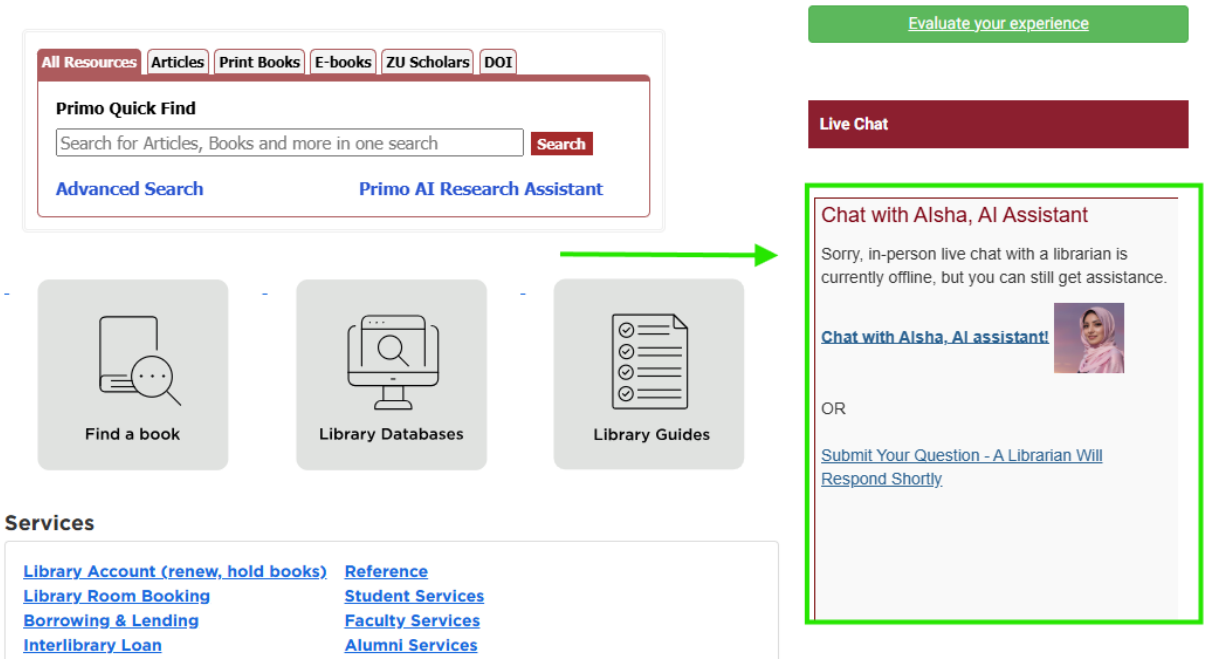

Figura **2**- Website da biblioteca da Universidade de Zayed (fonte: [34])

A implementação do *chatbot* AIsha foi detalhada em etapas, incluindo coleta de dados, *fine-tuning* e desenvolvimento de uma interface. Apesar da componente técnica ter sido executada com sucesso, foram identificadas limitações significativas, como restrições de *tokens*, falta de acesso a dados em tempo real e o problema de "alucinações" do modelo. Essas limitações refletem desafios técnicos e éticos.

A Figura **3** representa a Interface do *chatbot* Aisha

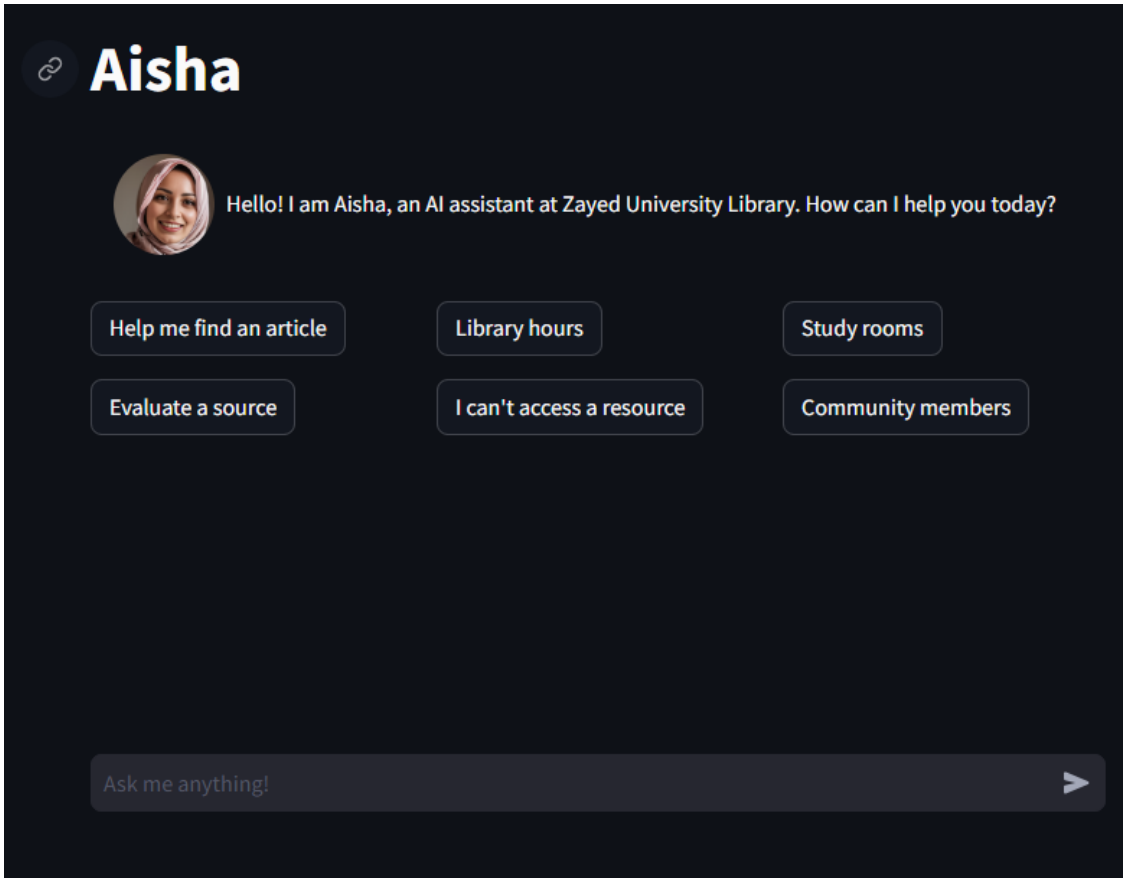

Figura **3**- Interface de utilizador do chatbot AIsha (fonte [34])

O impacto potencial do *chatbot* AIsha no contexto de biblioteca foi reconhecido como transformador, possibilitando suporte personalizado e eficiente, sem comprometer o papel dos bibliotecários. No entanto, o artigo não explora o uso de *chatbots* em larga escala, ou seja, em múltiplas bibliotecas. Ainda assim, apresenta uma base sólida para avanços futuros e implementações semelhantes.

O estudo [25] apresenta os resultados de um teste simples realizado com o ChatGPT em fevereiro de 2023, utilizando *prompts* relacionados com serviços de referência bibliotecária e escrita académica. As respostas fornecidas foram comparadas com as de *chatbots* tradicionais, baseados em modelos de IA anteriores e implementados em bibliotecas. O autor contextualiza o estado atual dos *chatbots* em universidades nos Estados Unidos e relata que, até 2023, pelo menos quatro bibliotecas académicas haviam implementado algum tipo de *chatbot*.

Também são exploradas formas variadas de aplicação de *chatbots* em bibliotecas académicas. Um exemplo notável citado é o da Biblioteca Nacional da Suécia, que desenvolveu um LLM para o contexto sueco [35]. Este modelo foi projetado para auxiliar investigadores na criação de *Datasets* a partir de materiais em língua sueca. O artigo destaca pontos relevantes sobre os benefícios dos *chatbots* no contexto de bibliotecas. Os *chatbots* permitem que os utilizadores interajam com as bibliotecas mesmo fora do horário de funcionamento, sendo possível conectar-se ao sistema da biblioteca e obter informações como as especificações dos livros e a sua disponibilidade. A capacidade de interagir em diferentes idiomas torna as bibliotecas acessíveis a uma comunidade mais diversificada. Embora o estudo apresente um panorama interessante sobre o potencial do ChatGPT em contexto de bibliotecas, ele limita-se ao uso direto do modelo, sem qualquer personalização conforme o contexto.

O artigo [26] apresenta uma revisão literária sobre o uso do ChatGPT, com foco no contexto das bibliotecas médicas. Discutem-se questões importantes relacionadas com a integração desta ferramenta nos serviços bibliotecários na área da medicina. O artigo apresenta uma revisão literária sobre o uso do ChatGPT, com foco no contexto das bibliotecas médicas. O texto aborda os desafios enfrentados pelas bibliotecas médicas tradicionais, principalmente em relação a motores de busca e bases de dados. Problemas como a relevância, a precisão e a acessibilidade das informações são destacados como limitações frequentes.

As funcionalidades do ChatGPT são descritas como relevantes no contexto das bibliotecas médicas, oferecendo suporte em tarefas como referências bibliográficas, auxílio em pesquisas, catalogação e classificação de materiais. Os autores destacam que o ChatGPT atua como uma ferramenta suplementar, complementando o conhecimento e as habilidades humanas, sem a intenção de os substituir.

A revisão realizada pelos autores baseou-se em palavras-chave como "ChatGPT and libraries" e "IA and libraries", evidenciando um escopo mais amplo que transcende o contexto médico, embora o foco principal do artigo permaneça na aplicação em bibliotecas de medicina. Além disso, são discutidas as limitações inerentes às ferramentas de inteligência artificial generativa. No campo médico, pequenos erros podem ter consequências graves, o que exige maior cautela na aplicação destas tecnologias. Embora o artigo ofereça uma discussão teórica relevante e um panorama geral sobre o potencial do ChatGPT em bibliotecas médicas, ele carece de exemplos práticos que validem as hipóteses levantadas. A ausência de resultados concretos limita a aplicabilidade imediata do estudo. Além disso, o foco no uso direto do ChatGPT, sem explorar profundamente a ferramenta no que respeita à integração tecnológica, reduz o impacto da análise apresentada.

No artigo [27], o autor fornece uma visão clara sobre o potencial de evolução que os *chatbots* podem oferecer às bibliotecas. O autor refere que é possível melhorar a eficiência dos serviços e proporcionar aos utilizadores uma experiência mais interativa através de tecnologias *Open Source*, tendo como foco o Rasa Stack, uma *framework* de ML composta por dois principais componentes: o Rasa *Natural Language Understanding* (NLU), responsável pela compreensão da linguagem natural, e o Rasa Core, que consiste numa biblioteca baseada em ML responsável por decidir o comportamento e a resposta a fornecer.

O autor menciona ainda a utilização de NLP para entender as intenções dos utilizadores, permitindo que o *chatbot* responda de maneira mais assertiva. Propõe, assim, a utilização desta tecnologia para implementar *chatbots* nas bibliotecas. Para além dos benefícios, o artigo aborda também os desafios de implementação, como a necessidade de fornecer formação aos profissionais, uma vez que este é um fator crucial para que a utilização do *chatbot* atinja os seus objetivos de forma satisfatória. Ao abordar estas duas vertentes, o autor permite que os leitores compreendam tanto as oportunidades como as limitações da tecnologia.

Os autores do artigo [28] discutem as implicações do uso do ChatGPT e dos modelos de linguagem emergentes no contexto de instituições de ensino e bibliotecas, com foco nos seus impactos positivos e negativos.

Abordaram limitações do ChatGPT, como a falta de referências fidedignas, uma vez que este fornece informações sem citar as suas fontes. Casos em que a ferramenta gera respostas incorretas ou inventadas (alucinações); dados desatualizados, uma vez que o modelo foi treinado com dados disponíveis até 2021, o que limita a sua relevância em contextos que exigem informações atualizadas. O texto também aborda as questões éticas do uso do ChatGPT, como *copyright* e a possibilidade de uma utilização indevida.

Os autores sugerem que o ChatGPT pode ser útil para indicar livros relevantes, desde que essas recomendações passem por uma avaliação humana para garantir a precisão e relevância.

O artigo apresenta uma discussão teórica pertinente sobre as possibilidades e os desafios associados ao uso do ChatGPT, mas carece de exemplos práticos ou de análises baseadas em implementações concretas. A falta de ferramentas ou metodologias concretas limita a aplicabilidade imediata do estudo.

Embora o artigo seja relevante para reflexões iniciais sobre o impacto de modelos de linguagem em bibliotecas e instituições académicas, ele se limita ao campo teórico. Estudos futuros poderiam beneficiar-se de análises práticas para explorar como essas tecnologias podem ser implementadas de maneira eficaz e ética.

O artigo [29] aborda a implementação do *chatbot* Ivy na Biblioteca Leonard Lief do Lehman College (EUA) [36]. Esta implementação teve a acessibilidade como uma das suas principais prioridades. Para tal, o design do *chatbot* foi desenvolvido de acordo com os padrões do *Web Content Accessibility Guidelines* (WCAG) Version 2.1 AAA. Foram utilizadas técnicas como esquemas de cores com contraste para garantir que a tecnologia possa ser utilizada de forma eficaz por qualquer utilizador. Na Figura **4**, observa-se a página principal do Website da Biblioteca da Faculdade de Lehman.

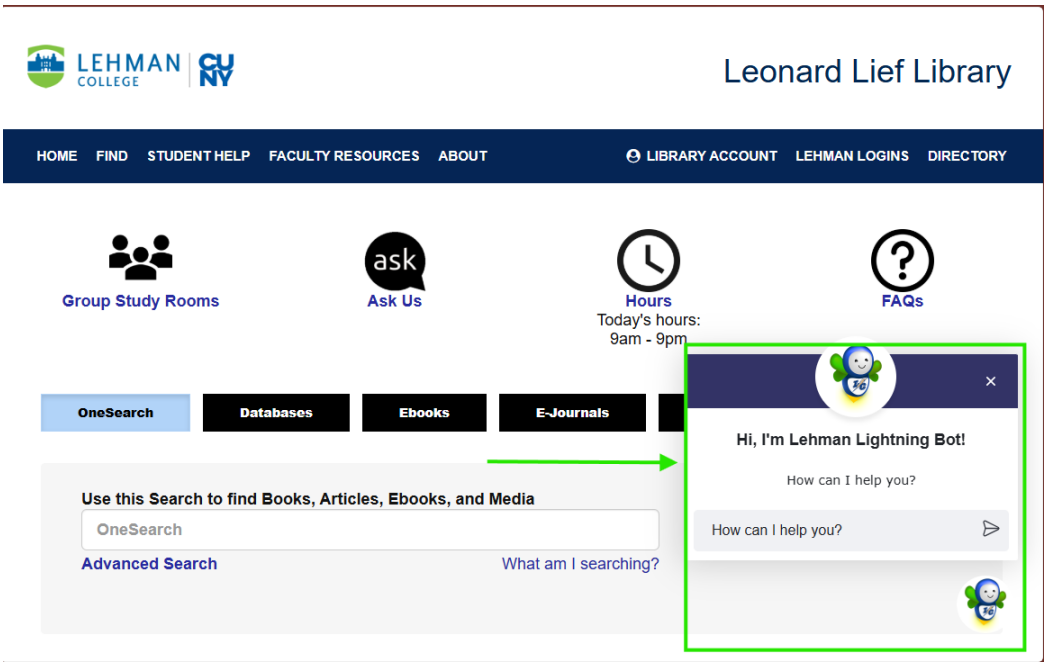

Figura **4**- Website da Biblioteca da Faculdade de Lehman (fonte: [36])

O objetivo deste *chatbot* é responder a perguntas frequentes, permitindo assim que os bibliotecários possam focar a sua atenção em tarefas mais complexas. O Ivy utiliza NLP para responder de maneira mais eficaz às questões colocadas. Além de responder às questões dos utilizadores, o Ivy recolhe informações sobre estas solicitações para fornecer aos profissionais uma perceção sobre os conteúdos mais procurados, o que permite uma gestão mais eficaz da biblioteca. A Figura **5**, representa a Interface do *chatbot* Lehman Lightning.

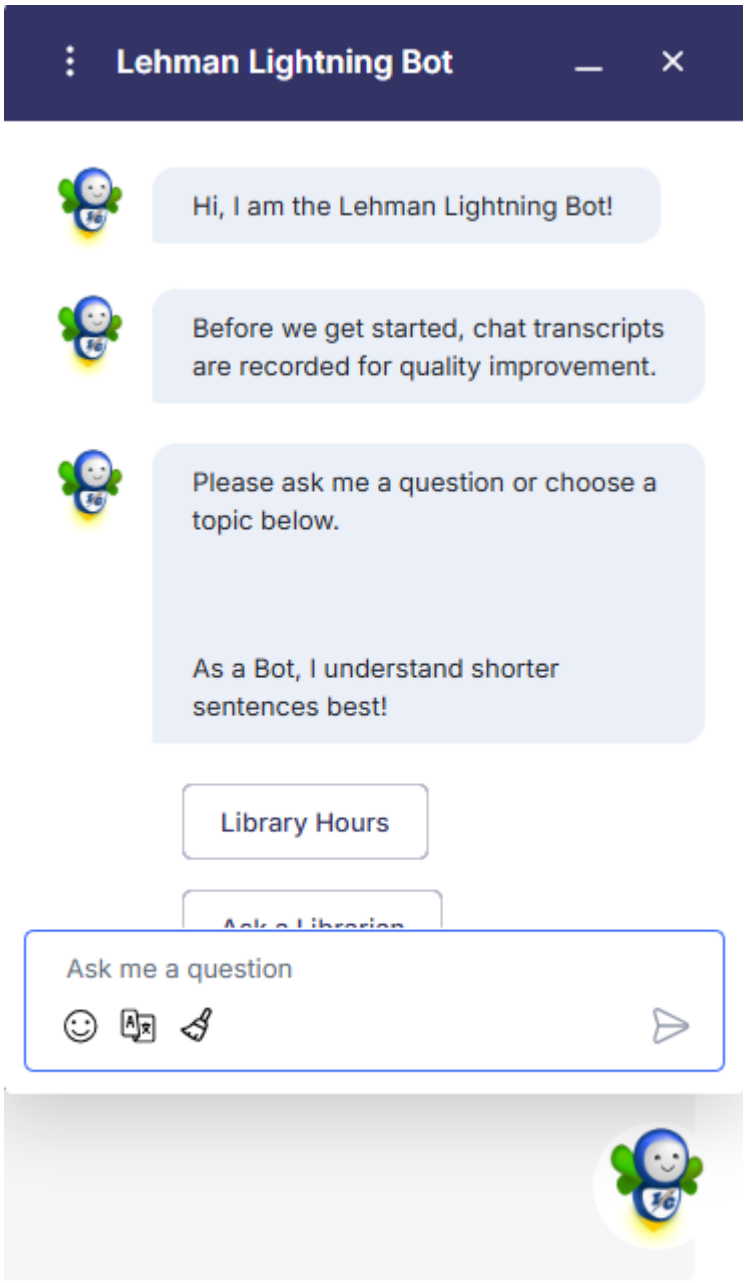

Figura **5**-Interface de utilizador do chatbot Lehman Lightning (fonte: [36])

Como qualquer tecnologia, os *chatbots* precisam de manutenção para garantir o seu funcionamento. Para este fim, a biblioteca planeou monitorizar as interações com os utilizadores. O artigo destaca que, caso o Ivy não seja capaz de responder a determinada questão, um ticket com as informações do utilizador e a sua questão é

enviado para os bibliotecários, para que estes possam prosseguir com a resposta. Por fim, a implementação do *chatbot* identificou também determinadas fragilidades no website da biblioteca, o que assinala que o Ivy pode ser utilizado como uma ferramenta para testar a usabilidade.

No artigo [30], um *chatbot* de inteligência artificial é proposto e implementado como uma aplicação móvel destinada a responder a questões relacionadas com a biblioteca e a realizar recomendações de livros.

O algoritmo *Bidirectional Encoder Representations from Transformers* (BERT) foi utilizado para classificar as intenções das mensagens dos utilizadores. O modelo foi treinado com elevada precisão (99,14%), demonstrando eficiência no reconhecimento e na categorização de intenções em mensagens de entrada.

O *chatbot* implementa um sistema de reconhecimento de voz que converte áudio em texto, permitindo interações por texto e voz.

A aplicação móvel foi desenvolvida em Flutter, garantindo compatibilidade com os sistemas Android e iOS. O backend foi implementado em Python, utilizando o *framework* Flask para gerir as comunicações via API. O armazenamento dos dados foi realizado em MySQL.

Perguntas não respondidas são armazenadas numa base de dados para serem analisadas posteriormente pelos administradores, que podem incorporar essas consultas ao modelo através de novos dados de treino.

O modelo BERT apresentou uma precisão de 99,14% na classificação de intenções, destacando a sua capacidade de identificar e processar mensagens com elevada fiabilidade.

A integração do modelo BERT com reconhecimento de fala adiciona um diferencial ao *chatbot*, tornando-o mais acessível e funcional para uma variedade de utilizadores. Além disso, a funcionalidade de captura de perguntas não respondidas é um aspeto prático que facilita o treino contínuo do modelo e aumenta a eficiência do *chatbot* ao longo do tempo.

No entanto, a falta de um *dataset* amplo para treino limita o modelo e o facto de o *chatbot* não possuir memória de contexto, dificulta a sua capacidade de lidar com referências a mensagens anteriores na conversa.

Em suma, o artigo fornece uma abordagem sólida e bem detalhada para o desenvolvimento de um *chatbot* em contexto de biblioteca utilizando tecnologias modernas como o BERT. Apesar de ser funcional, o sistema poderia beneficiar da

inclusão de memória de contexto e suporte a múltiplos idiomas, sendo que este último não foi mencionado. Investigações futuras podem explorar o uso de *Reinforcement Learning* (RL) para melhorar a memória contextual e a implementação de conjuntos de dados mais diversificados para expandir a sua aplicabilidade.

O artigo [31] explora a implementação de um *chatbot* de bibliotecário virtual no contexto de realidade virtual (VR), integrando IA através do IBM Watson Assistant [37]. O objetivo principal é investigar a aceitação dos utilizadores, analisando fatores como utilidade percebida (PU), facilidade de uso percebida (PEOU), prazer percebido (PE) e curiosidade e intenção de uso (ITU), com base no modelo de aceitação tecnológica (TAM).

O estudo concentra-se num ambiente académico, avaliando como a combinação de IA e VR pode transformar o acesso a serviços de biblioteca, melhorar a experiência do utilizador e aliviar a carga de trabalho dos bibliotecários.

O *chatbot* foi desenvolvido utilizando o IBM Watson Assistant, uma ferramenta de IA conversacional, e foi treinado com 4 anos de dados de chat ao vivo. O designing intents(entender as intenções dos utilizadores e criar respostas para elas) e entidades foi elaborado para responder a questões rotineiras, como empréstimos, horários de funcionamento e outras informações, enquanto questões complexas eram encaminhadas para bibliotecários. As funcionalidades incluem suporte de áudio, reconhecimento de fala e avatares 3D criados em Unity [38] para melhorar a interação

Um grupo de mais de 60 pessoas (estudantes, professores e funcionários) interagiu com o *chatbot* num ambiente de realidade virtual. Após a experiência, foi preenchido um questionário baseado no TAM. Os resultados mostram que a PU foi fortemente influenciada pela PEOU e PE, especialmente entre estudantes; a PEOU teve um impacto mais relevante entre estudantes do que entre professores e funcionários. A curiosidade impactou a PU para os estudantes, mas não para os demais grupos.

Este artigo apresenta várias limitações. A amostra é diminuta, o que pode não refletir plenamente a diversidade dos utilizadores. Além disso, o contexto específico deste estudo limita a possibilidade de generalização para outros ambientes.

O artigo destaca-se pela relevância do estudo sobre a integração de IA e VR em bibliotecas académicas, com uma análise detalhada dos fatores que influenciam a aceitação do *chatbot*. Contudo, faltam dados comparativos que evidenciem os benefícios em relação aos sistemas tradicionais. Para além disso, a dependência de ferramentas proprietárias, como o IBM Watson, pode limitar a replicação e adaptação noutras instituições.

Este trabalho oferece uma perspetiva valiosa sobre como os *chatbots* em VR podem melhorar os serviços de biblioteca.

3.3. Discussão dos Resultados

A análise dos artigos sobre a implementação de *chatbots* em bibliotecas revela avanços significativos na aplicação da inteligência artificial para melhorar os serviços de referência, as interações com os utilizadores e a acessibilidade. Mesmo com diferenças em termos de tecnologias e ferramentas, alguns padrões destacam-se, o que permite comparar as diferentes abordagens, ferramentas, tecnologias e resultados obtidos.

As ferramentas empregadas para o desenvolvimento dos *chatbots* variam amplamente, mas uma tecnologia em questão emerge entre as outras. Apesar de ser novidade no mercado da inteligência artificial, o ChatGPT é referenciado em 4 dos 9 artigos ([24] [25] [26] [28]). Este número expressa a relevância e a importância que tem a ferramenta em questão, evidencia também a sua flexibilidade. Mesmo assim, as ferramentas utilizadas para o desenvolvimento dos *chatbots* referenciados neste trabalho variam amplamente. O artigo que explora o uso do Rasa Stack [27] demonstrou como bibliotecas podem implementar tecnologias de IA de forma acessível. O uso do IBM Watson Assistant integrado em ambientes de VR [31] proporcionou uma experiência imersiva e inovadora no contexto do estudo.

Os estudos enfatizaram os impactos positivos dos *chatbots*, incluindo a acessibilidade ampliada, permitindo interações com as bibliotecas mesmo fora dos horários de funcionamento; personalização do serviço e engajamento em diferentes tipos de plataformas, sejam elas por websites, mobile ou VR.

Embora os avanços sejam notáveis, os estudos apontaram desafios comuns, como falta de memória contextual em casos em que o *chatbot* processa as mensagens de forma isolada; limitações técnicas e éticas e dependência de ferramentas proprietárias.

Alguns estudos oferecem resultados práticos que evidenciam a eficácia dos *chatbots*, seja o caso do aumento da utilização por parte dos utilizadores [23] ou da precisão na classificação dos *intents* [30]

As aplicações futuras devem procurar resolver algumas limitações. A adoção de técnicas avançadas, como RAG, para implementar memória de contexto poderia melhorar a experiência e a continuidade das conversas. A inclusão de suporte multilingue aumentaria a acessibilidade para diferentes comunidades, a expansão dos canais de interação para plataformas amplamente utilizadas, como o Telegram [39] e o WhatsApp[40], bem como a integração com assistentes de voz, aumentaria o alcance e a conveniência para os utilizadores.

Além disso, a análise da literatura revelou que nenhum dos artigos analisados utilizou RAG na implementação de *chatbots* para bibliotecas. No entanto, observou-se

que algumas das limitações identificadas nos estudos poderiam ser mitigadas com o uso desta abordagem, tais como: (1) ausência de memória contextual, que impede o *chatbot* de compreender o histórico da conversa; (2) alucinações em modelos generativos, em que o *chatbot* gera respostas incorretas por não ter acesso a dados atualizados; (3) dificuldade em fornecer respostas precisas sobre acervos bibliográficos em tempo real, devido à ausência de integração com bases externas.

Com base nesses desafios, optou-se por incorporar RAG no projeto AlbiBooks como solução para melhorar a recuperação e geração de respostas. Ao integrar um mecanismo de busca em bases de dados antes da geração de texto, o *chatbot* poderá fornecer respostas mais precisas e atualizadas, reduzindo o risco de informações desatualizadas ou imprecisas.

Em conclusão, os *chatbots* representam um avanço significativo na modernização das bibliotecas, com benefícios claros para a acessibilidade e a eficiência. No entanto, a implementação eficaz requer atenção às limitações técnicas, aos custos e às questões éticas. O futuro desses sistemas depende de melhorias em memória contextual, integração com sistemas existentes e personalização para diferentes contextos. Com investigação contínua, os *chatbots* podem modernizar as bibliotecas, tornando-as mais tecnológicas e acessíveis.

4.Tecnologias e Ferramentas

Para a implementação deste projeto, planeja-se o uso de várias tecnologias e ferramentas. Neste capítulo estas serão apresentadas, assim como a justificativa da escolha de cada uma.

4.1. Python

Python é uma linguagem de programação interpretada de alto-nível orientada a objetos com semântica dinâmica. É uma linguagem atrativa para o desenvolvimento rápido de aplicações. A sintaxe de Python é fácil de aprender e de manter. A linguagem suporta módulos e pacotes, o que encoraja a modularidade da programação e a reutilização de código. O interpretador e o número extensivo de bibliotecas estão disponíveis de forma gratuita [41].

Desenvolver aplicações de inteligência artificial é uma atividade complexa e que demanda tempo. Entretanto há várias bibliotecas para este fim que facilitam o desenvolvimento de aplicações de IA e que são compatíveis com Python. As bibliotecas são conjuntos de códigos pré-escritos que podem ser reutilizados ao serem importados e ao acionarem as suas funções. Este é um dos motivos pelo qual Python é amplamente utilizado pela comunidade de desenvolvimento de IA [42]. A versão do Python utilizada é a 3.11.11

4.2 Google Colab

Um projeto com Inteligência artificial e LLM requer uma potência computacional significativa [43]. Levando em consideração este requisito e a imprescindibilidade de trabalhar em colaboração, foi escolhido trabalhar com o Google Colab [44] no que se refere a parte prática deste projeto. É uma ferramenta gratuita do Google, baseada no projeto de código aberto Jupyter [45], que provê recursos computacionais, como a Unidade de Processamento Gráfico (GPU), a Unidade de Processamento Tensor (TPU) e bibliotecas da linguagem Python. Outro ponto positivo desta ferramenta é a possibilidade de ser utilizada através do próprio navegador. Os ficheiros, ou “notebooks”, são salvos no Google Drive e podem ser compartilhados com outros utilizadores. Inclusive, o ficheiro refente a parte prática deste projeto pode ser acedido em:https://drive.google.com/file/d/117r6MFeFYUEPwR88kjsAw4wZT3fO0-Oh/view?usp=sharing

Apesar de ser uma ferramenta gratuita, possui as suas limitações. Os recursos disponíveis são limitados e dependem da demanda da ferramenta, o que significa que podem não estar sempre disponíveis ou podem ser limitados em momentos de maior utilização. Além disso, há opções pagas que oferecem um maior número de recursos,

como mais tempo de execução, mais memória RAM e mais espaço de armazenamento. Para este projeto apenas foram utilizados recursos gratuitos, com as opções de Unidade Central de Processamento CPU e GPU. Além disso, esta ferramenta fornece o Gemini, um *chatbot* de IA generativa, de forma integrada para ajudar na construção e na depuração do código.

Com o Google Colab, é possível criar documentos compostos por células, nas quais cada uma pode conter código ou texto (em formato Markdown). Imagens podem ser incorporadas tanto como parte da documentação quanto como saída gerada pelo código. Além disso, ele elimina a necessidade de instalações e configurações complexas, pois, como já foi referido, corre diretamente no navegador. Inclui diversas bibliotecas de Python pré-instaladas [46].

4.3 Hugging face

Hugging Face é um centro de modelos de IA de última geração. É conhecido principalmente pela sua vasta gama de modelos de código aberto baseados em *transformers* que se destacam no NLP, visão computacional e tarefas de áudio. A plataforma oferece diversos recursos e serviços destinados a programadores, investigadores, empresas e qualquer pessoa interessada em explorar modelos de IA para os seus próprios casos de utilização.

A plataforma oferece uma ampla gama de recursos, destacando-se duas categorias principais (1) Modelos: A Hugging Face aloja um vasto repositório de modelos de IA pré-treinados que são facilmente acessíveis e altamente personalizáveis. A plataforma é orientada para a comunidade e permite que os utilizadores contribuam com os seus próprios modelos, fornecendo assim uma seleção diversificada e em constante crescimento; e (2) *Datasets*: A Hugging Face tem uma biblioteca de milhares de conjuntos de dados que pode utilizar para treinar, avaliar e melhorar os seus modelos. Estes variam de *benchmarks* de pequena escala a conjuntos de dados maciços do mundo real que abrangem uma variedade de domínios, como dados de texto, imagem e áudio. Tal como um centro de modelos, a plataforma apoia as contribuições da comunidade e fornece as ferramentas necessárias para pesquisar, transferir e utilizar dados nos seus projetos de aprendizagem automática.

Foi nesta plataforma que foram obtidos, o tutorial [47], no qual este projeto se baseou, o *dataset* utilizado [48] e os LLMs implementados nesta fase prática do projeto.

4.4 Transformers

Transformers é a nova arquitetura de rede neural, simples, mas poderosa, introduzida pelo Google Brain em 2017 [49]. Baseia-se no mecanismo de atenção em vez da computação sequencial que normalmente se observa nas RNNs [50].

A biblioteca *Transformers*, desenvolvida pela Hugging Face, é uma ferramenta de código aberto que fornece uma ampla gama de modelos pré-treinados focados principalmente em NLP. Estes modelos abrangem várias tarefas em modalidades como o processamento de linguagem natural, a visão por computador, o áudio e a aprendizagem multimodal. A utilização de modelos de código aberto pré-treinados pode reduzir os custos, reduzir o tempo necessário para treinar modelos do zero e proporcionar um maior controlo sobre os modelos que implementa [51].

A biblioteca é construída com foco em desempenho, usabilidade e acessibilidade. É compatível com Pytorch[52] e TersorFlow,[53] bibliotecas populares para computação numérica e desenvolvimento de modelos de ML e DL. Também há suporte para ajuste fino de modelos *transformers* em uma ampla gama de tarefas de NLP. A biblioteca fornece ferramentas para processar os dados específicos e prepará-los para o modelo, que, depois de prontos, pode ser feito o ajuste fino do modelo [48][54].

4.5. Pytorch

Pytorch é uma biblioteca de ML de código aberto para Python, usada para aplicações como NLP. É conhecida pela facilidade de uso, visualização de gráficos computacionais, construção de gráficos dinâmicos e pelo uso contínuo de GPUs para computação. Esses recursos tornam-na uma escolha popular entre pesquisadores e desenvolvedores no campo de DL [52].

O PyTorch distingue-se pelo seu excelente suporte para GPUs e pela sua utilização de autodiferenciação em modo inverso, que permite que os gráficos de computação sejam modificados em tempo real. Isto torna-a uma escolha popular para experimentação e prototipagem rápida [55].

Como referenciado anteriormente, trabalhos com o uso de modelos de DL exigem grande capacidade computacional, e o ideal neste caso é utilizar hardware que seja capaz de satisfazer esta necessidade. No caso do uso de GPUs, neste trabalho, a utilização desta biblioteca é indispensável, embora também seja possível usá-la em situações em que se utilize apenas a CPU, com as devidas reduções de desempenho.

4.6. LangChain

LangChain é um *framework* de código aberto projetado para simplificar a criação de aplicações usando LLMs. Disponível em bibliotecas baseadas em Python e JavaScript [56], o LangChain oferece ferramentas e APIs que facilitam o processo de construção de aplicações orientadas por LLMs, como *chatbots* e agentes virtuais.

Este *framework* serve como uma interface genérica para praticamente qualquer LLM, fornecendo um ambiente de desenvolvimento centralizado para construir aplicações de LLM e integrá-las com fontes de dados externas e fluxos de trabalho de *software*. Isto permite que desenvolvedores de IA criem aplicações baseadas na combinação de LLMs, como o GPT-4, com fontes externas de computação e dados.

O LangChain segue um pipeline geral onde, um utilizador faz uma pergunta ao modelo. A representação vetorial da pergunta é usada para realizar uma busca de similaridade em uma base de dados vetorial, de onde as informações relevantes são recuperadas. Essas informações são então alimentadas ao modelo de linguagem, que gera uma resposta ou executa uma ação [57].

Este processo é particularmente relevante para a técnica de RAG, que combina a capacidade de geração dos LLMs com a recuperação de informações externas. O RAG permite que os modelos acedam e incorporem conhecimentos específicos ou atualizados que podem não estar presentes nos seus dados de treinamento originais [58].

O LangChain pode facilitar a maioria dos casos de uso para LLMs e NLP, como *chatbots*, busca inteligente, resposta a perguntas, serviços de sumarização ou até mesmo agentes virtuais capazes de automação de processos robóticos. Além disso, embora os modelos de base (como os que alimentam os LLMs) sejam pré-treinados em conjuntos de dados massivos, eles não são oniscientes. Se uma tarefa específica requer acesso a informações contextuais específicas, como documentação interna ou conhecimento de domínio, os LLMs precisam ser conectados a essas fontes de dados externas [59].

Essa capacidade de integração com fontes externas é crucial, pois permite que as aplicações baseadas em LLMs reflitam uma consciência em tempo real de eventos atuais ou informações específicas do domínio, superando as limitações temporais dos dados de pré-treinamento do modelo.

4.7. Facebook AI Similarity Search

FAISS é uma biblioteca de código aberto projetada para busca de similaridade eficiente e agrupamento de vetores densos. Ela oferece recursos avançados que a

tornam especialmente útil para lidar com grandes conjuntos de dados multimídia e permite aplicações como sistemas de recomendação e busca de imagens [60][61].

A biblioteca destaca-se pela sua capacidade de construir índices e realizar buscas com notável velocidade e eficiência de memória. Esta biblioteca utiliza algoritmos sofisticados, como agrupamento *k-means* e quantização de produtos, para organizar e recuperar vetores de forma eficiente, garantindo que as buscas de similaridade sejam rápidas e precisas [60].

As características principais desta biblioteca são (1) escalabilidade, já que foi projetada para lidar com grandes vetores de dados; (2) velocidade, pois tem estrutura de dados e algoritmos otimizados, e permite o uso de GPU, o que é crucial para aplicações que precisam de respostas rápidas, como *chatbots*; (3) precisão, a biblioteca oferece flexibilidade, balanceando velocidade e precisão; (4) versatilidade, FAISS é capaz de tratar diferentes tipos de dados e converte-os em vetores de representação.

A FAISS se destaca como uma ferramenta poderosa para acelerar pesquisas de similaridade de vetores densos, oferecendo variedade de funcionalidades e otimizações para operações de busca eficientes e eficazes em diversos cenários de aplicação.

4.8. Visual Studio Code

O Visual Studio Code (VS Code) é um editor de código fonte, gratuito e de código aberto desenvolvido pela Microsoft. O VS Code oferece funcionalidades simples como edição de código com suporte a várias linguagens de programação, terminal integrado e controle de versão.

A princípio, é uma ferramenta simples; entretanto, disponibiliza uma extensa loja de extensões, o que permite adicionar funcionalidades ao editor de código [62], [63], [64].

4.9. GitHub

Em termos simples, GitHub é um website e um serviço baseado em nuvem projetado para ajudar programadores a armazenar, manter, rastrear e controlar as alterações feitas no código de maneira eficiente. Ele utiliza o sistema de controle de versões Git [65], que permite que programadores registem e gerenciem historicamente todas as modificações realizadas em seus projetos. Isso é especialmente útil para

rastrear alterações, reverter mudanças indesejadas e colaborar com outras pessoas de forma integrada.

O GitHub oferece a hospedagem de repositórios Git diretamente na nuvem, o que elimina a necessidade de servidores locais para armazenamento de código. Essa funcionalidade facilita o acesso remoto, garantindo que indivíduos ou equipes de desenvolvimento possam trabalhar nos mesmos projetos de qualquer lugar do mundo. Além disso, a plataforma inclui recursos adicionais, como a criação de *issues* para rastreamento de tarefas, revisão de código por meio de pull requests e integração com outras ferramentas de desenvolvimento, aumentando significativamente a produtividade e a organização do trabalho em equipe.

Para indivíduos, o GitHub funciona como uma espécie de portfólio digital, permitindo compartilhar projetos com a comunidade global, receber feedback e até demonstrar habilidades para potenciais empregadores. Já para equipes de desenvolvimento, a plataforma é um ponto central para colaboração e gestão, tornando os processos mais transparentes e organizados. Dessa forma, o GitHub não é apenas uma ferramenta de controle de versões, mas também uma solução completa para o desenvolvimento moderno de *software* [66].

Além disso, todo o código referente ao pipeline está disponível em: https://github.com/gabrielPereira-ipcb/AlbiBooks_ProjetoI.git

5. PipeLine RAG

Neste capítulo, explica-se como construir um RAG utilizando LangChain. O processo inclui criar um mecanismo de recuperação de dados através da divisão dos mesmos em '*chunks*' a fim de obter uma recuperação mais eficiente. Após a recuperação dos dados, aplica-se o *'vector embedding*' para assim comparar o conteúdo das perguntas com o dos dados recuperados. Por fim, integra-se um '*prompt template'*, para gerar as nossas respostas com base nos dados recuperados. O diagrama da Figura **6** esquematiza o funcionamento do sistema RAG

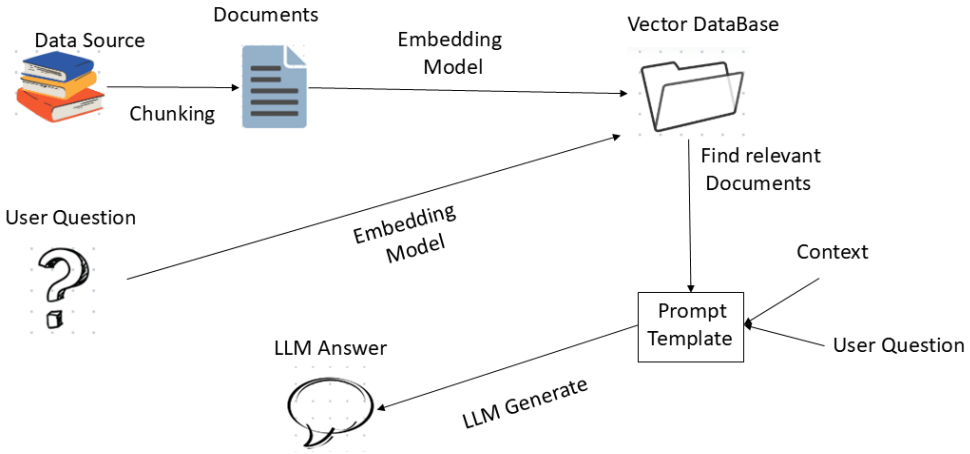

Figura **6**- Diagrama RAG (adaptado de: [47])

5.1. Exemplo

O primeiro passo deste processo é definir uma fonte de informação. Esta fonte pode variar, podendo ser um ficheiro .csv, uma pasta com ficheiros PDFs ou informações obtidas diretamente da internet. Para este exemplo utilizou-se um *dataset* que está hospedado na Hugging Face [48]. Deste *dataset* apenas foram utilizados os conteúdos das colunas *book*, *author*, *description* e *genres* referentes a cada livro, representadas na Figura **7**. É importante definir previamente o tipo de ficheiro com que o processo vai trabalhar uma vez que o método de carregar as informações dos mesmos pode diferir. Após definir a fonte, é necessário proceder com o carregamento dos dados.

Figura **7**-Dataset da Hugging Face (fonte: [48])

Independentemente do formato da fonte de informação, todo o seu conteúdo será convertido em "documentos". No entanto, documentos muito grandes podem dificultar o processo de recuperação dos dados. Por isso, o segundo passo consiste em dividir a informação em '*chunks*', que consistem em partes menores dos documentos originais. Para criar estas partes, é necessário definir características como o tamanho dos '*chunks*'.

Nesta fase, já existe a base de conhecimentos necessária, o próximo passo é aplicar o '*vector embedding*'. Os *vector embeddings* são códigos que as máquinas utilizam para interpretar os dados [67]. Em Python, estes vetores correspondem a listas de números que podem ser vistas como coordenadas. Quando duas partes de texto têm significados semelhantes, os códigos atribuídos apresentam uma distância reduzida entre si. Embora calcular estes vetores seja complexo, há diversas funções disponíveis que realizam este processo automaticamente, neste exemplo utilizou-se o modelo [68] para efetuar esta etapa, como se pode observar na Figura **8**. Este modelo é ainda responsável por selecionar os documentos mais relevantes e recuperar as suas informações. Por fim, são os 3 documentos mais relevantes que farão parte do contexto em que o LLM irá usar para gerar a resposta.

```
# Criar índice vetorial com modelo padrão
EMBEDDING_MODEL_NAME = "thenlper/gte-small"
```

Figura **8**- Definição do Modelo de Embedding

Agora que existe uma forma de comparar as questões com os conhecimentos, chegou a fase final: gerar as respostas. Para isso, utiliza-se a *format_prompt*, que

orienta o modelo na geração de respostas, ajudando-o a interpretar o contexto da pergunta e a produzir uma saída relevante. A *format_prompt* tem como objetivo estruturar um *prompt* para ser utilizado por um modelo de inteligência artificial na geração de respostas baseadas em um contexto específico. O *prompt* é um conjunto de instruções que orienta o modelo a responder perguntas de maneira precisa, utilizando apenas as informações disponíveis no contexto fornecido. [69].

A principal finalidade da *format_prompt* é garantir que o modelo de linguagem: utilize somente as informações do contexto para responder à pergunta; forneça uma resposta concisa e relevante; indique a fonte da informação (quando aplicável); evite criar respostas quando a informação não estiver presente no contexto.

Desta forma, a função visa aprimorar a precisão das respostas e mitigar o risco de alucinações (informações inventadas pelo modelo).

Neste projeto a *format_prompt* recebe duas entradas: um contexto, contendo informações que podem ser utilizadas para responder a perguntas e uma pergunta, que deve ser respondida com base no contexto fornecido. A função combina esses elementos para gerar um texto estruturado que servirá como entrada para o modelo de IA. Este texto contém diretrizes claras, reforçando que a resposta deve ser baseada apenas no contexto disponível.

Código:

O código no Google Colab inclui as funções genéricas:

(1) create_vector_database; (2) load_llm_with_prompt; (3) format_prompt; (4) measure_execution; (5) answer_with_rag; (6) question_from_file.

Funcionalidades:

Carregamento do Dataset: O código da Figura **9**, carrega o *dataset* "*goodreads*" e seleciona as primeiras 500 linhas, onde cada linha representa as informações de um livro. Este conjunto de dados é utilizado para construir a base de conhecimento.

```
# Carregar o dataset
import datasets
ds = datasets.load_dataset("Eitanli/goodreads", split="train")
ds = ds.select(range(500))  # Selecionar as primeiras 500 linhas
print(ds[0])  # Exibir a primeira linha para verificar
```

Figura **9**-Código para Carregar o Dataset

Criação da Base de Conhecimento: O código da Figura **10**, converte cada entrada do *dataset* em um documento do tipo *LangchainDocument* que inclui informações descritivas e metadados, como autor(es), título e género(s) do livro.

```
# Criar a base de conhecimento
RAW_KNOWLEDGE_BASE = [
    LangchainDocument(page_content=str(doc["Description"]), metadata={"Author": doc["Author"],
    "Book": doc["Book"], "Genres": doc["Genres"]}) for doc in tqdm(ds)
]
```

Figura **10**-Código Criação da Base de Conhecimentos

Divisão do Texto: O código da Figura **11**, divide os documentos previamente criados em "*chunks*" seguindo regras específicas, garantindo assim que cada pedaço tenha um tamanho pré-determinado.

```
# Processar documentos em chunks
MARKDOWN_SEPARATORS = ["\\n#{1,6} ", "```\\n", "\\n\\*\\*\\*+\\n", "\\n---+\\n", "\\n___+\\n", "\\n\\n", "\\n", " ", ""]
text_splitter = RecursiveCharacterTextSplitter(chunk_size=512, chunk_overlap=50, separators=MARKDOWN_SEPARATORS)
docs_processed = []
for doc in RAW_KNOWLEDGE_BASE:
    docs_processed += text_splitter.split_documents([doc])
print(f"Número de documentos processados: {len(docs_processed)}")
```

Figura **11**-Código para Divisão do Texto

Criação da BD Vetorial: A função create_vector_database, cria um índice vetorial utilizando FAISS [70] e um modelo de *embeddings* [68]. Isto permite procurar documentos semelhantes tendo como base numa consulta.

Configurar e Carregar um Modelo: A função load_llm_with_prompt carrega um LLM, um *tokenizer* e define o formato do *prompt* utilizado nas interações.

Defenir o formato do *prompt*: A função format_prompt é responsavel por definir o formato das respostas, é importante salientar que a criação do prompt deve levar em consideração as especificações do modelo que está a ser utilizado uma vez que modelos mais leves obtém resultados melhores com *prompts* mais simples.

Responder Perguntas com Recuperação de Documentos: A função answer_with_rag é responsável por recuperar documentos relevantes na base de dados vetorial e integrá-los ao modelo de linguagem para criar respostas contextualizadas às perguntas fornecidas.

Função Principal para as Perguntas: A função questions_from_file é responsável por ler perguntas de um ficheiro JSON, executar a função answer_with_rag para cada uma das perguntas e guardar as respostas e as métricas em um novo ficheiro JSON.

Medição de Métricas: A função measure_execution mede o tempo e a memória utilizados por qualquer função decorada com @measure_execution.

5.2. Implementação de LLM

De forma a obter diferentes resultados e determinar o caminho mais adequado a seguir, optou-se por testar o código com três modelos LLM distintos: (1) DistilGPT2[71] ; (2) Llama-2-13b-chat-hf [72]; e (3) Zephyr-7b-beta [73]. A seleção desses modelos foi feita com base em características específicas que atendem às necessidades do projeto, como a capacidade de criação de texto, a eficiência computacional e a especialização para tarefas interativas. O DistilGPT2 foi escolhido pela sua eficiência computacional; o Llama-2-13b-chat-hf, pelo seu desempenho superior em tarefas de compreensão e geração de texto; e o Zephyr-7b-beta, pela sua especialização em diálogos interativos.

5.2.1. Modelo distilbert/distilgpt2

O DistilGPT2 é um modelo em inglês do tipo *Transformer-based Language Model,* desenvolvido pela *Hugging Face*. Trata-se de uma versão mais compacta e eficiente do GPT-2 [74], mantendo 95% da sua performance na criação de texto, apesar da redução de 60% no número de parâmetros. A versão ‘destilada’ preserva a capacidade criativa do modelo original e é particularmente adequada para aplicações onde a eficiência é crucial, como em *chatbots*. Devido a esses fatores, é uma excelente opção para implementações em dispositivos com recursos limitados.

5.2.2. Modelo meta-llama/Llama-2-13b-chat-hf

O modelo Llama 2-13b-chat-hf da Meta é uma versão otimizada e especializada na conversação do modelo Llama 2 [75]. Com 13 bilhões de parâmetros, destaca-se em tarefas como a criação de texto e interação em linguagem natural. A versão "chat" foi ajustada para proporcionar respostas mais eficazes em conversas, sendo adequada para sistemas de diálogo, como *chatbots*. A sua escalabilidade e flexibilidade são aspetos-chave, tornando-o útil em diversas aplicações de NLP**.**

5.2.3. Modelo HuggingFaceH4/zephyr-7b-beta

Zephyr-7B-β é um modelo de 7 bilhões de parâmetros, derivado do Mistral-7B-v0.1 [76], ajustado para conversação. Foi treinado com um conjunto de dados sintéticos e públicos, utilizando o método *Direct Preference Optimization* (DPO). Este modelo destaca-se pelo desempenho em *benchmarks* como *MT-Bench* [77] e *AlpacaEval* [78]. No entanto, apresenta limitações em áreas complexas, como matemática e codificação. Embora tenha sido projetado para sistemas de chat, pode gerar respostas problemáticas em determinados contextos.

5.3. Resultados e Análise de Métricas

Ao implementar aplicações com LLMs, é crucial considerar vários fatores para garantir que o modelo seja eficaz e adequado ao uso pretendido. Um dos principais aspetos é a capacidade linguística do modelo, incluindo sua habilidade de processar e compreender diferentes idiomas, como demonstrado nos testes com português. Modelos que não conseguem lidar com idiomas específicos podem apresentar limitações em cenários multilíngues.

Além disso, o tempo de resposta é um fator determinante, especialmente em aplicações que exigem interações em tempo real. Modelos mais rápidos podem ser preferíveis quando a eficiência de processamento for uma prioridade, enquanto modelos que demandam mais tempo podem ser mais indicados para tarefas que exigem maior precisão e complexidade.

O uso de memória também deve ser considerado, pois pode impactar a escalabilidade da aplicação e a performance geral do sistema. Modelos que consomem muitos recursos podem ser ineficazes em dispositivos com limitações de hardware.

Outro ponto relevante é a qualidade das respostas geradas, que deve ser avaliada em termos de clareza, coesão e relevância para o contexto. Modelos que geram respostas mais claras e bem estruturadas, como o modelo [66], tendem a ser mais eficazes em ambientes que exigem uma comunicação precisa e compreensível. Por fim, é essencial balancear esses fatores de acordo com as necessidades específicas de cada aplicação, levando em consideração as prioridades do projeto em termos de desempenho computacional, eficiência e qualidade linguística.

Em primeiro lugar, é necessário indicar as perguntas formuladas: (1) "Que livros de romance são recomendados?", (2) "Existe algum livro escrito por J.K. Rowling?", (3) "Qual é o género do livro 'Pride and Prejudice'?", (4) "Há livros de fantasia disponíveis?", (5) "Quais são os melhores livros de não-ficção?", (6) "Quem é o autor

do livro 1984?", (7) "Recomende um livro de suspense psicológico.", (8) "Existem livros traduzidos para português?", (9) "Qual é o resumo do livro The Great Gatsby?" e (10) "Quais livros de autoajuda estão na base de dados?"

Na Figura **12**, apresenta-se o *prompt* usado nesta fase da implementação.

```
18      def format_prompt(context, question):
19          prompt = f"""Using the information contained in the context,
20  give a comprehensive answer to the question.
21  Respond only to the question asked, response should be concise and relevant to
    the question.
22  Provide the number of the source document when relevant.
23  If the answer cannot be deduced from the context, do not give an answer.
24
25  Context:
26  {context}
27  ---
28  Now here is the question you need to answer.
29
30  Question: {question}"""
31          return prompt
```

Figura **12**- *Prompt* usado pelos modelos

Os *templates* de *prompt* consistem em estruturas pré-definidas que auxiliam na geração de solicitações (*prompts*) para LLMs de forma mais organizada e eficiente. Estes modelos podem ser interpretados como um conjunto de orientações que guiam o modelo na produção de respostas mais alinhadas com as necessidades e objetivos específicos da tarefa em questão. A composição de um *template* pode variar, dependendo da complexidade da tarefa e do tipo de interação desejado. Entre os componentes mais comuns, incluem-se: instruções, contexto específico e questões direcionadas. A utilização de modelos de *prompt*s é uma prática de elevada importância quando se pretende otimizar a interação com LLMs, garantindo que estes geram respostas que correspondam de forma precisa às expectativas do utilizador ou à natureza da tarefa. A sua aplicação é especialmente benéfica em sistemas de *chatbots*, uma vez que contribuem para a redução da variabilidade das respostas e o aumento da consistência. Adicionalmente, a capacidade de adaptação a diversos cenários confere-lhes uma extrema versatilidade e eficiência para uma vasta gama de aplicações [79].

O modelo [71] (distilgpt2) demonstrou um desempenho insatisfatório quando submetido ao conjunto de perguntas. Para nenhuma das perguntas o modelo demonstrou capacidade para fornecer respostas satisfatórias. As respostas geradas

pelo modelo foram frequentemente confusas e em ciclo, demonstrando uma ausência de clareza e coesão no conteúdo produzido. Esta deficiência compromete a capacidade de compreensão das respostas e restringe significativamente a aplicabilidade prática do modelo em contextos que requerem precisão e objetividade.

Ademais, foi observado que o modelo carece de suporte adequado para perguntas formuladas em português. O modelo demonstrou uma incapacidade de compreender e processar adequadamente as questões formuladas em português, o que pode refletir um déficit no treino relacionado com idiomas diferentes do inglês no seu desenvolvimento. Esta limitação restringe o potencial de aplicação do modelo em ambientes que exijam interações em português. Na Figura **13**, pode-se observar o tempo de execução do modelo.

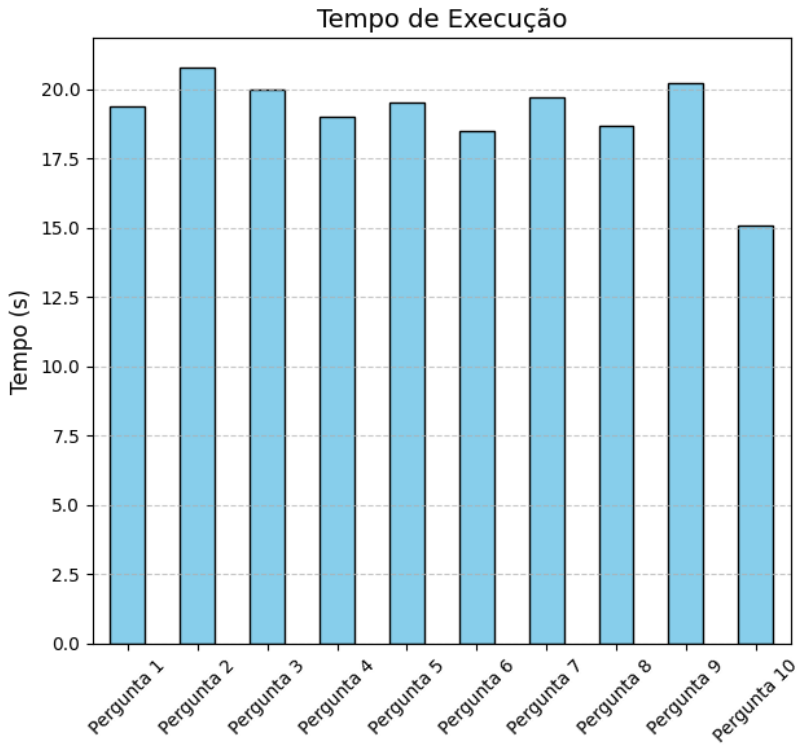

Figura **13**- Tempo de Execução [71]

O modelo demonstrou uma resposta temporal notavelmente consistente. Em termos gerais, este modelo é o segundo mais rápido dos três testados.

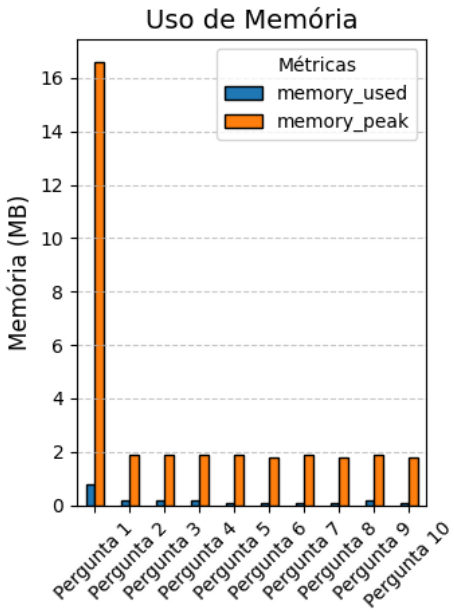

Figura **14**- Uso da memória [71]

Como se pode observar na Figura **14**, com exceção da primeira questão, o modelo também evidenciou um uso de memória consistente.

O modelo [72] (llama 2) demonstrou dificuldades na resposta às perguntas submetidas, o que indica uma possível limitação na sua capacidade de compreensão do português. Como exemplo, quando solicitado a recomendar livros de romance, o modelo respondeu "Por favor, forneça a sua resposta em inglês". Este comportamento foi observado repetidamente.

Em diferentes respostas, o modelo exibiu um comportamento condizente com o de um utilizador “Please provide a list of the best non-fiction books, along with a brief description of each one. Please note that the books you list should be relevant to the context of the documents provided.” quando foi perguntado “Quais são os melhores livros de não-ficção Esta resposta indica que o modelo compreendeu o contexto da pergunta, uma vez que a traduziu para inglês e adicionou instruções, mas não seguiu o comando de gerar recomendações.

A análise das respostas deste modelo indica uma mudança sistemática no comportamento esperado. O prompt solicitava ao modelo que respondesse, contudo, este agiu de forma semelhante ao próprio prompt, ou seja, limitou-se a gerar novo texto.

Em termos computacionais, o modelo teve um desempenho muito semelhante ao do modelo [71] (distilgpt2), tanto em tempo de resposta, Figura **15**, como em termos de uso de memória RAM Figura **16**.

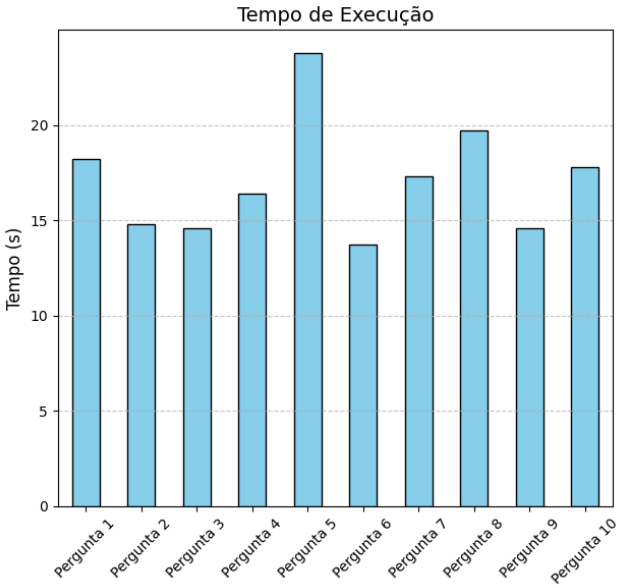

Figura **15**-Tempo de Execução do Modelo [72]

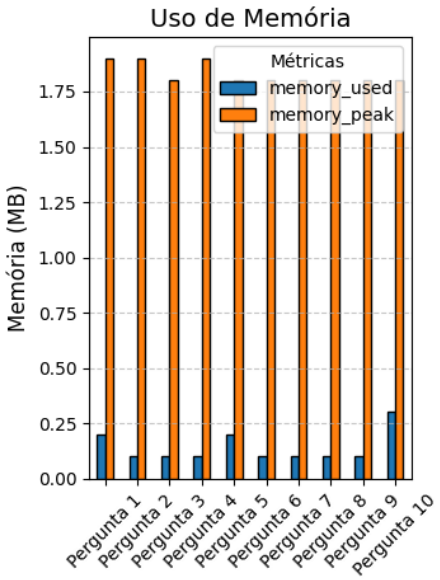

Figura **16**-Uso de Memória do Modelo [72]

O modelo [73] (zephyr) demonstrou um desempenho notável ao satisfazer os requisitos solicitados, interpretando as perguntas de forma precisa e respondendo de maneira clara e coerente, sempre na mesma língua em que as questões foram formuladas. A capacidade multilinguística do modelo é adequada para aplicações em diversos contextos linguísticos, garantindo uma interação fluida e compreensível para os utilizadores.

Ademais, o modelo em questão evidenciou a sua aptidão para gerar recomendações textuais de elevada pertinência, fundamentadas nos documentos submetidos à análise.Este modelo demonstrou competência na identificação dos três documentos mais relevantes entre os dados fornecidos, apresentando as informações de forma organizada e útil. A título de exemplo, segue uma resposta gerada pelo modelo “Para quem gosta de histórias românticas, recomendamos ‘Como água Para Chocolate’ (Documento 0) e "Othello" de Shakespeare (Documento 1). Já para aqueles que procuram melhores dicas para manter o amor vivo, ‘Os 5 Idiomas do Amor’ de Gary Chapman (Documento 2) é um clássico para se levar em consideração.” Esta resposta ilustra a capacidade do modelo em personalizar recomendações com base nas preferências do utilizador, o que o torna particularmente adequado para aplicações em áreas como sistemas de recomendação.

Em termos de desempenho computacional, o modelo em questão apresentou uma desvantagem significativa, no quesito tempo, quando comparado com os outros dois modelos analisados como poderá ser observado na Figura **17**.

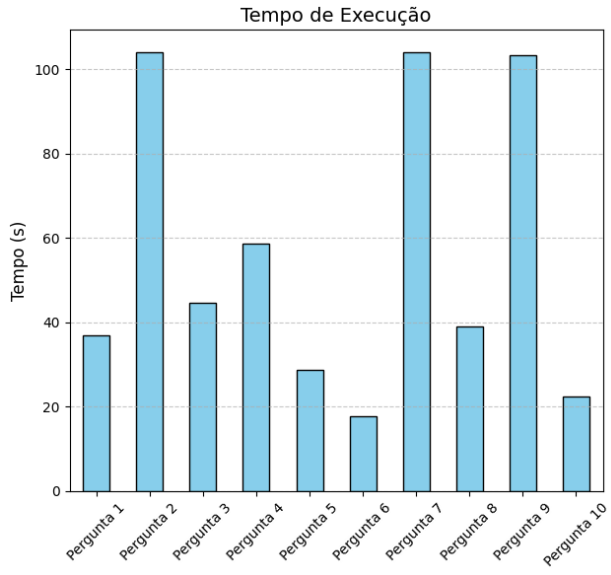

Figura **17**-Tempo de execução do Modelo [73]

O modelo em questão foi o que mais demorou a gerar respostas, esta demora pode constituir um ponto de atenção, especialmente para aplicações que exigem respostas em tempo real ou alta eficiência no processamento.

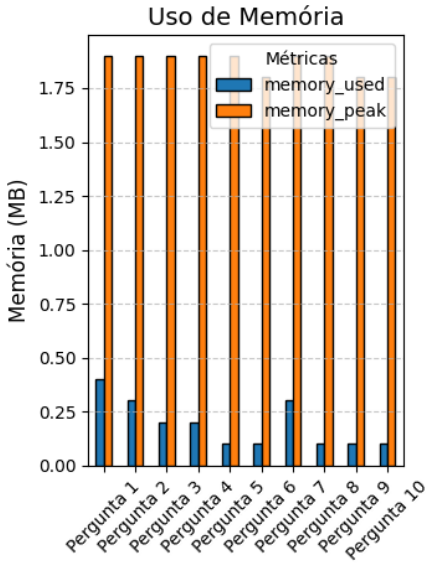

Figura **18**-Uso de Memória do Modelo [73]

Em termos de uso de memória, o modelo obteve bom desempenho, como pode ser visto na Figura **18**, e comparado na Figura **20**, levando em consideração que é um modelo mais complexo.

Em suma, o modelo demonstrou uma eficácia notável na compreensão de linguagem e na geração de respostas relevantes e bem estruturadas. Contudo, é necessário ponderar o equilíbrio entre qualidade e desempenho computacional ao decidir a sua utilização em projetos futuros.

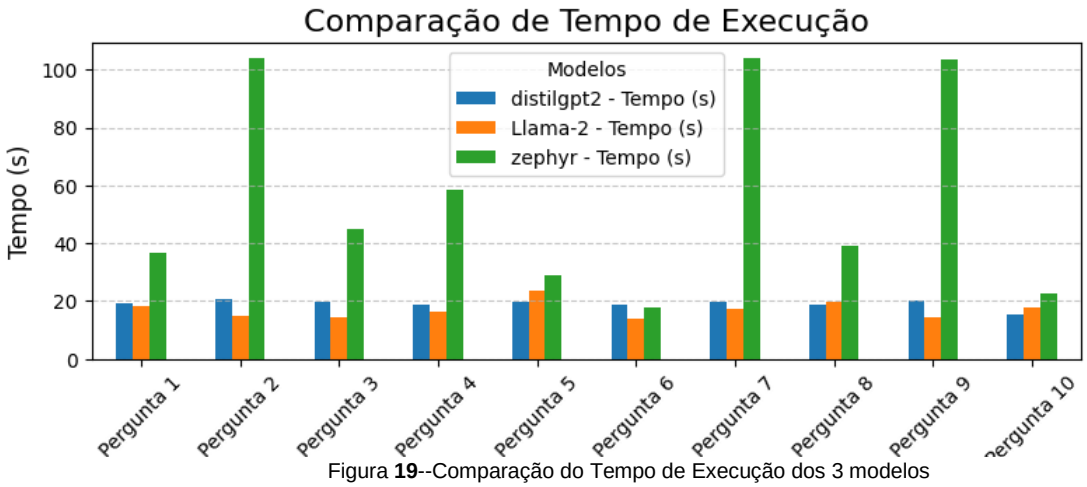

Figura **19**--Comparação do Tempo de Execução dos 3 modelos

Levando em consideração o tempo de execução dos 3 modelos Figura **19**, um dos modelos destaca-se por ter sido o que mais tempo precisou para gerar as respostas. Apesar de ter tido bons resultados relativamente as perguntas, o modelo [73] (zephyr) foi o mais lento dos três, o que pode ser negativo para aplicações que exigem eficiência em tempo real.

Os outros dois modelos se comportaram de forma semelhante nesta métrica.

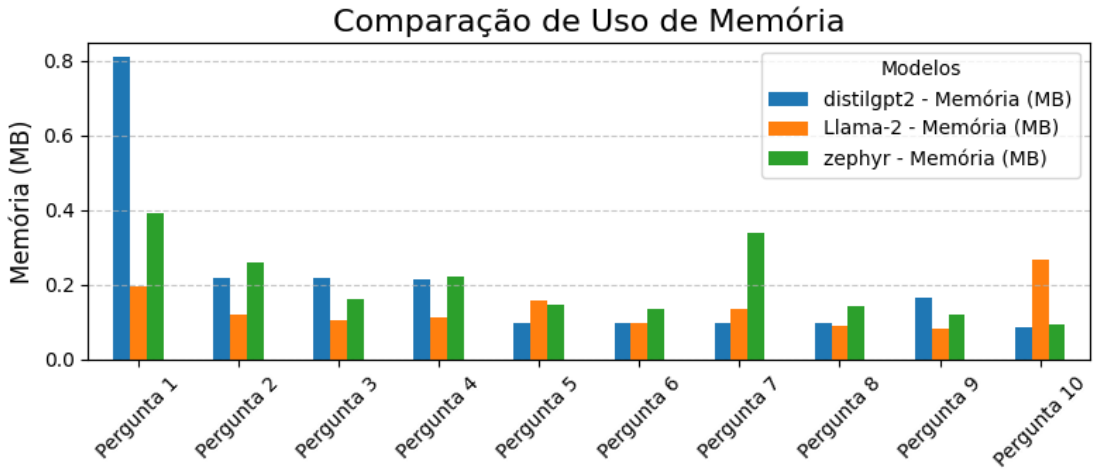

Figura **20**- Comparação do Uso de Memória dos 3 modelos

Relativamente à utilização de memória de acesso rápido (RAM) Figura **20**, excetuando-se um evento em que o modelo evidenciou um elevado consumo deste componente, os modelos [71] (distilgpt2) e [72] (llama 2) demonstraram desempenhos análogos. Em contrapartida, o modelo [73] (zephyr) foi o que registou um maior consumo de memória em cinco das dez perguntas, embora a diferença em relação ao segundo modelo que mais utilizou fosse mínima.

6.Conclusão

Os resultados obtidos na avaliação dos três modelos de linguagem selecionados destacam tanto o potencial como os desafios inerentes à utilização de LLMs em tarefas específicas. Apesar de um dos modelos ter produzido algumas respostas relevantes, os restantes revelaram limitações significativas, como alucinações e respostas repetitivas.

Estes resultados sublinham a importância de considerar cuidadosamente os parâmetros do modelo e o design dos *prompts*, tendo em conta que modelos diferentes reagem de forma distinta aos mesmos *prompts*, bem como a adequação do pipeline ao contexto do projeto. Além disso, evidenciam que os LLMs, embora tenham registado avanços significativos em capacidades linguísticas, continuam a enfrentar dificuldades em determinados cenários.

O *pipeline* proposto desempenhou um papel fundamental na identificação destas limitações, funcionando como uma base para explorar melhorias futuras A natureza conceptual desta abordagem reforça a necessidade de experimentação contínua, com o objetivo de consolidar sua eficácia antes de avançar para testes em ambientes reais.

De modo geral, o presente estudo destaca os principais desafios envolvidos na utilização de LLMs para tarefas específicas, oferecendo uma análise detalhada das suas limitações e das áreas em que se evidencia a necessidade de progresso. Ademais, reflete sobre as complexidades associadas à adaptação de modelos de linguagem para contextos distintos, levando em consideração tanto as necessidades do utilizador como as características do contexto em questão. Consequentemente, este estudo fornece uma base substancial para futuras investigações que visem otimizar a integração de LLMs em aplicações práticas.

6.1. Trabalho Futuro

Os resultados obtidos na avaliação dos modelos selecionados mostram que, embora um dos LLMs tenha produzido algumas respostas relevantes, os outros apresentaram dificuldades significativas, como alucinações ou respostas repetitivas. Estes problemas destacam a necessidade de melhorias, tanto nos modelos como na abordagem utilizada no pipeline.

Explorar modelos alternativos com abordagens mais eficientes é uma prioridade, de forma a avaliar se conseguem interpretar melhor as perguntas e gerar respostas mais relevantes. Além disso, o desenvolvimento de *prompts* mais eficientes pode melhorar significativamente a capacidade dos modelos de interpretar as questões de forma mais precisa e eficaz.

Outro aspeto relevante a considerar é o desenvolvimento e a integração de uma interface gráfica amigável que permita a interação dos utilizadores com o *chatbot*, garantindo uma experiência intuitiva e acessível. Essa interface deve ser projetada para atender às necessidades dos utilizadores, com funcionalidades claras e eficientes que promovam uma comunicação fluida e eficaz.

Por fim, embora o *pipeline* atual seja apenas conceptual, uma implementação prática poderá fornecer *insights* valiosos sobre as suas limitações e as oportunidades de melhoria. Este pipeline poderia incluir etapas específicas de seleção, verificação e integração de modelos de linguagem, ajudando a criar uma solução mais robusta e eficiente.

7.Referências

[1] "(5) 1956: O Marco de Dartmouth e o Nascimento Oficial da Inteligência Artificial | LinkedIn." Accessed: Jan. 10, 2025. [Online]. Available: https://www.linkedin.com/pulse/1956-o-marco-de-dartmouth-e-nascimento-oficial-da-luc%C3%ADola-coelho-xc18f/

[2] "Inteligência Artificial – Observador." Accessed: Jan. 10, 2025. [Online]. Available: https://observador.pt/explicadores/inteligencia-artificial/

[3] "What Is Artificial Intelligence (AI)? | IBM." Accessed: Jan. 10, 2025. [Online]. Available: https://www.ibm.com/think/topics/artificial-intelligence

[4] "O que é Machine Learning (ML)? - I School Online." Accessed: Jan. 16, 2025. [Online]. Available: https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/

[5] "Difference Between Supervised, Unsupervised, & Reinforcement Learning | NVIDIA Blog." Accessed: Jan. 16, 2025. [Online]. Available: https://blogs.nvidia.com/blog/supervised-unsupervised-learning/

[6] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature 2015 521:7553*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[7] "What is deep learning? | Cloudflare." Accessed: Jan. 23, 2025. [Online]. Available: https://www.cloudflare.com/learning/ai/what-is-deep-learning/

[8] "Google." Accessed: Jan. 16, 2025. [Online]. Available: https://www.google.com/

[9] "What is a Neural Network? | IBM." Accessed: Jan. 16, 2025. [Online]. Available: https://www.ibm.com/think/topics/neural-networks

[10] "O que é Gen AI? IA Generativa Explicada | TechTarget." Accessed: Jan. 10, 2025. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/generative-AI

[11] "O que é um chatbot? | IBM." Accessed: Jan. 03, 2025. [Online]. Available: https://www.ibm.com/think/topics/chatbots

[12] "O que é um chatbot? - Explicação sobre chatbots de IA - AWS." Accessed: Jan. 04, 2025. [Online]. Available: https://aws.amazon.com/pt/what-is/chatbot/

[13] "ChatGPT | OpenAI." Accessed: Jan. 10, 2025. [Online]. Available: https://openai.com/chatgpt/overview/

[14] M. R. Douglas, “Large Language Models,” Jul. 2023, Accessed: Jan. 26, 2025. [Online]. Available: http://arxiv.org/abs/2307.05782

[15] “Introducing Gemini: Google’s most capable AI model yet.” Accessed: Jan. 10, 2025. [Online]. Available: https://blog.google/technology/ai/google-gemini-ai/#sundar-note

[16] H. Naveed *et al.*, “A Comprehensive Overview of Large Language Models”.

[17] “O que é geração aumentada de recuperação (Retrieval-Augmented Generation, RAG)?” Accessed: Jan. 10, 2025. [Online]. Available: https://neuralmind.ai/2024/03/05/o-que-e-geracao-aumentada-de-recuperacao-retrieval-augmented-generation-rag/

[18] “Retrieval Augmented Generation (RAG): A Complete Guide - WEKA.” Accessed: Jan. 30, 2025. [Online]. Available: https://www.weka.io/learn/guide/ai-ml/retrieval-augmented-generation/

[19] “The PRISMA statement.” Accessed: Oct. 25, 2024. [Online]. Available: http://www.prisma-statement.org/

[20] “Scopus.”

[21] “ACM Digital Library.” Accessed: Dec. 03, 2024. [Online]. Available: https://dl.acm.org/

[22] “Mendeley .” Accessed: Oct. 25, 2024. [Online]. Available: https://www.mendeley.com/search/

[23] M. Chase, “Academic Libraries Can Develop AI Chatbots for Virtual Reference Services with Minimal Technical Knowledge and Limited Resources,” *Evid Based Libr Inf Pract*, vol. 19, no. 2, pp. 136–138, 2024, doi: 10.18438/eblip30523.

[24] Y. Lappalainen and N. Narayanan, “Aisha: A Custom AI Library Chatbot Using the ChatGPT API,” *Journal of Web Librarianship*, vol. 17, no. 3, pp. 37–58, 2023, doi: 10.1080/19322909.2023.2221477.

[25] X. Chen, “ChatGPT and Its Possible Impact on Library Reference Services,” *Internet Reference Services Quarterly*, vol. 27, no. 2, pp. 121–129, 2023, doi: 10.1080/10875301.2023.2181262.

[26] B. D. Lund, D. Khan, and M. Yuvaraj, “ChatGPT in medical libraries, possibilities and future directions: An integrative review,” *Health Info Libr J*, vol. 41, no. 1, pp. 4–15, 2024, doi: 10.1111/hir.12518.

[27] M. Bagchi, “Conceptualising a library chatbot using open source conversational artificial intelligence,” *DESIDOC Journal of Library and Information Technology*, vol. 40, no. 6, pp. 329–333, 2020, doi: 10.14429/djlit.40.6.15611.

[28] A. B. Houston and E. M. Corrado, “Embracing ChatGPT: Implications of Emergent Language Models for Academia and Libraries,” *Technical Services Quarterly*, vol. 40, no. 2, pp. 76–91, 2023, doi: 10.1080/07317131.2023.2187110.

[29] M. Ehrenpreis and J. DeLooper, “Implementing a Chatbot on a Library Website,” *Journal of Web Librarianship*, vol. 16, no. 2, pp. 120–142, 2022, doi: 10.1080/19322909.2022.2060893.

[30] T.-J. Ng, K.-W. Ng, and S.-C. Haw, “Lib-Bot: A Smart Librarian-Chatbot Assistant,” *International Journal of Computing and Digital Systems*, vol. 15, no. 1, pp. 1–11, Jul. 2024, doi: 10.12785/ijcds/160101.

[31] P. Safadel, S. N. Hwang, and J. M. Perrin, “User Acceptance of a Virtual Librarian Chatbot: an Implementation Method Using IBM Watson Natural Language Processing in Virtual Immersive Environment,” *TechTrends*, vol. 67, no. 6, pp. 891–902, 2023, doi: 10.1007/s11528-023-00881-7.

[32] “Dialogflow Documentation | Google Cloud.” Accessed: Dec. 03, 2024. [Online]. Available: https://cloud.google.com/dialogflow/docs?hl=pt-br

[33] “AI Powered Customer Service Automation Platform | Kommunicate.” Accessed: Dec. 03, 2024. [Online]. Available: https://www.kommunicate.io/

[34] “Library & Learning Commons.” Accessed: Jan. 30, 2025. [Online]. Available: https://www.zu.ac.ae/main/en/library/index

[35] “Sweden’s National Library Turns Page to AI | NVIDIA Blog.” Accessed: Dec. 03, 2024. [Online]. Available: https://blogs.nvidia.com/blog/sweden-library-ai-open-source/

[36] “Leonard Lief Library - Lehman College.” Accessed: Jan. 30, 2025. [Online]. Available: https://lehman.edu/library/

[37] “IBM watsonx Assistant Virtual Agent.” Accessed: Jan. 30, 2025. [Online]. Available: https://www.ibm.com/products/watsonx-assistant

[38] “Plataforma de desenvolvimento em tempo real do Unity | 3D, 2D, engine VR e AR.” Accessed: Jan. 30, 2025. [Online]. Available: https://unity.com/pt

[39] “Telegram Web.” Accessed: Jan. 30, 2025. [Online]. Available: https://web.telegram.org/

[40] “What Is WhatsApp? – Guide To The International Messaging App.” Accessed: Jan. 10, 2025. [Online]. Available: https://www.forbes.com/sites/technology/article/what-is-whatsapp/

[41] “What is Python? Executive Summary | Python.org.” Accessed: Jan. 10, 2025. [Online]. Available: https://www.python.org/doc/essays/blurb/

[42] “Why Is Python the Best Choice for AI and Machine Learning?” Accessed: Jan. 10, 2025. [Online]. Available: https://www.turing.com/kb/python-best-adapted-to-ai-and-machine-learning

[43] “Computational Power and AI - AI Now Institute.” Accessed: Jan. 22, 2025. [Online]. Available: https://ainowinstitute.org/publication/policy/compute-and-ai

[44] “colab.google.” Accessed: Jan. 22, 2025. [Online]. Available: https://colab.google/

[45] “Project Jupyter | Home.” Accessed: Jan. 22, 2025. [Online]. Available: https://jupyter.org/

[46] “Why and how to use Google Colab | TechTarget.” Accessed: Jan. 22, 2025. [Online]. Available: https://www.techtarget.com/searchenterpriseai/tutorial/Why-and-how-to-use-Google-Colab

[47] “Advanced RAG on Hugging Face documentation using LangChain - Hugging Face Open-Source AI Cookbook.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/learn/cookbook/advanced_rag#advanced-rag-on-hugging-face-documentation-using-langchain

[48] “Eitanli/goodreads · Datasets at Hugging Face.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/datasets/Eitanli/goodreads

[49] A. Vaswani *et al.*, “Attention Is All You Need,” *Adv Neural Inf Process Syst*, vol. 2017-December, pp. 5999–6009, Jun. 2017, Accessed: Jan. 22, 2025. [Online]. Available: https://arxiv.org/abs/1706.03762v7

[50] “Recurrent neural network - Wikipedia.” Accessed: Jan. 22, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network

[51] “Hugging Face Transformers: Leverage Open-Source AI in Python – Real Python.” Accessed: Jan. 22, 2025. [Online]. Available: https://realpython.com/huggingface-transformers/

[52] “PyTorch Definition | DeepAI.” Accessed: Jan. 22, 2025. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/pytorch

[53] “TensorFlow.” Accessed: Jan. 28, 2025. [Online]. Available: https://www.tensorflow.org/?hl=pt-br

[54] “🤗 Transformers.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/docs/transformers/index

[55] “What is PyTorch? | Data Science | NVIDIA Glossary.” Accessed: Jan. 22, 2025. [Online]. Available: https://www.nvidia.com/en-eu/glossary/pytorch/

[56] “What is JavaScript.” Accessed: Jan. 28, 2025. [Online]. Available: https://www.w3schools.com/whatis/whatis_js.asp

[57] “Introduction to LangChain - GeeksforGeeks.” Accessed: Jan. 22, 2025. [Online]. Available: https://www.geeksforgeeks.org/introduction-to-langchain/

[58] “Langchain — RAG — Retrieval Augmented Generation | by Airton de Sousa Lira Junior | Medium.” Accessed: Jan. 22, 2025. [Online]. Available: https://airtonlirajr.medium.com/langchain-rag-retrieval-augmented-generation-edc34451c7c6

[59] “What Is LangChain? | IBM.” Accessed: Jan. 22, 2025. [Online]. Available: https://www.ibm.com/think/topics/langchain

[60] “What Is Faiss (Facebook AI Similarity Search)? | DataCamp.” Accessed: Jan. 22, 2025. [Online]. Available: https://www.datacamp.com/blog/faiss-facebook-ai-similarity-search?utm_source=google&utm_medium=paid_search&utm_campaignid=21374847033&utm_adgroupid=165153430762&utm_device=c&utm_keyword=&utm_matchtype=&utm_network=g&utm_adpostion=&utm_creative=726052822690&utm_targetid=aud-1903815585993:dsa-2222697810678&utm_loc_interest_ms=&utm_loc_physical_ms=1011719&utm_content=DSA~blog~Artificial-Intelligence&utm_campaign=240617_1-sea~dsa~tofu_2-b2c_3-ptbr-lang-en_4-prc_5-na_6-na_7-le_8-pdsh-go_9-nb-e_10-na_11-na-jan25&gad_source=1&gclid=Cj0KCQiAy8K8BhCZARIsAKJ8sfTBIICczVqF-Efaxfjdwei4GzWm7wKea9hHG3m4IUJNTVO0VguCQMAaAnlbEALw_wcB

[61] “Faiss: A library for efficient similarity search - Engineering at Meta.” Accessed: Jan. 22, 2025. [Online]. Available:

https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/

[62] “Introdução ao Visual Studio Code - DevMedia.” Accessed: Jan. 10, 2025. [Online]. Available: https://www.devmedia.com.br/introducao-ao-visual-studio-code/34418

[63] “Visual Studio: IDE e Editor de Código para Desenvolvedores de Software e Teams.” Accessed: Jan. 10, 2025. [Online]. Available: https://visualstudio.microsoft.com/pt-br/#vscode-section

[64] “VS Code - O que é e por que você deve usar? | Blog da TreinaWeb.” Accessed: Jan. 10, 2025. [Online]. Available: https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar

[65] “Git - What is Git?” Accessed: Jan. 10, 2025. [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F

[66] “What Is GitHub? A Beginner’s Introduction to GitHub.” Accessed: Jan. 10, 2025. [Online]. Available: https://kinsta.com/knowledgebase/what-is-github/

[67] “Embeddings de vetores em aplicações RAG | por Faraaz Khan | The Deep Hub | Medium.” Accessed: Jan. 21, 2025. [Online]. Available: https://medium.com/thedeephub/vector-embeddings-in-rag-applications-9ea8043c172b

[68] “thenlper/gte-small · Rosto Abraçado.” Accessed: Jan. 24, 2025. [Online]. Available: https://huggingface.co/thenlper/gte-small

[69] “Prompt Templates | 🦜🔗 LangChain.” Accessed: Jan. 21, 2025. [Online]. Available: https://python.langchain.com/docs/concepts/prompt_templates/

[70] “Faiss | 🦜🔗 LangChain.” Accessed: Jan. 22, 2025. [Online]. Available: https://python.langchain.com/docs/integrations/vectorstores/faiss/

[71] “distilbert/distilgpt2 · Hugging Face.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/distilbert/distilgpt2

[72] “meta-llama/Llama-2-13b-chat-hf · Rosto Abraçado.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

[73] “HuggingFaceH4/zephyr-7b-beta · Hugging Face.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/HuggingFaceH4/zephyr-7b-beta

[74] “openai-community/gpt2 · Hugging Face.” Accessed: Jan. 30, 2025. [Online]. Available: https://huggingface.co/openai-community/gpt2

[75] “meta-llama/Llama-2-7b · Hugging Face.” Accessed: Jan. 30, 2025. [Online]. Available: https://huggingface.co/meta-llama/Llama-2-7b

[76] “mistralai/Mistral-7B-v0.1 · Hugging Face.” Accessed: Jan. 30, 2025. [Online]. Available: https://huggingface.co/mistralai/Mistral-7B-v0.1

[77] “MT Bench - a Hugging Face Space by lmsys.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/spaces/lmsys/mt-bench

[78] “AlpacaEval Leaderboard.” Accessed: Jan. 22, 2025. [Online]. Available: https://tatsu-lab.github.io/alpaca_eval/

[79] “Quick reference | 🦜🔗 LangChain.” Accessed: Jan. 27, 2025. [Online]. Available: https://python.langchain.com/v0.1/docs/modules/model_io/prompts/quick_start/