



UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
FACULDADE DE ENGENHARIA ELÉTRICA

CLÁUDIO KLAUTAU MELLO

ACADEMICK : Assistente de estudos para o ambiente acadêmico

BELÉM/PA
2025

CLÁUDIO KLAUTAU MELLO

ACADEMICK : Assistente de estudos para o ambiente acadêmico

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia Elétrica, pela Universidade Federal do Pará.

Orientador: Prof. Dr. Aldebaro Barreto da Rocha
Klautau Júnior
Universidade Federal do Pará

Coorientadora: Prof. Dra. Ana Carolina Quintão Siravenha Müller
Universidade Federal do Pará

CLÁUDIO KLAUTAU MELLO

ACADEMICK : Assistente de estudos para o ambiente acadêmico

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia Elétrica, pela Universidade Federal do Pará.

DATA DE APROVAÇÃO: 23/06/2025

CONCEITO: Excelente

Prof. Dr. Aldebaro Barreto da Rocha
Klautau Júnior
Orientador - FCT/ITEC/UFPA

Prof. Dra. Ana Carolina Quintão Siravenha
Müller
Coorientadora - FEEB/ITEC/UFPA

Prof. Dra. Adriana Rosa Garcez Castro
Membro Interno - FEEB/ITEC/UFPA

Prof. Dr. Ronaldo de Freitas Zampolo
Membro Interno - FCT/ITEC/UFPA

BELÉM/PA
2025

RESUMO

O surgimento dos Grandes Modelos de Linguagem (LLMs) criou oportunidades sem precedentes para aprimorar os processos educacionais, no entanto, muitas instituições de ensino superior ainda não aproveitaram plenamente essas tecnologias. Este trabalho apresenta um sistema adaptativo de chatbot educacional que integra LLMs com Geração Aumentada por Recuperação (RAG) para fornecer assistência personalizada ao aprendizado. A principal inovação do sistema está em sua abordagem baseada em intenções, utilizando um modelo de classificação personalizado baseado na arquitetura Snowflake/snowflake-arctic-embed-l-v2.0 que alcançou 97,6% de precisão no conjunto de teste. Este classificador, treinado em um conjunto de dados bilíngue (português e inglês), permite que o sistema identifique as intenções dos estudantes e ajuste dinamicamente sua estratégia de resposta. O sistema também apresenta um algoritmo inovador de processamento de PDF para extração e indexação eficiente de materiais educacionais. O chatbot pode ser rapidamente implantado para apoiar diversos cenários educacionais, desde a compreensão de livros didáticos até a análise de artigos acadêmicos, exigindo apenas os materiais fonte em formato PDF e uma configuração mínima. Ao combinar esses componentes, o chatbot fornece respostas contextualizadas, cita com precisão os materiais fonte e adapta seu estilo de interação com base na intenção identificada do estudante. Este trabalho demonstra como a moderna tecnologia LLM pode ser efetivamente integrada em ferramentas educacionais para aprimorar a experiência de aprendizagem, mantendo considerações práticas de implementação.

Palavras-chave: Grandes Modelos de Linguagem (LLMs). Geração Aumentada por Recuperação (RAG). Tecnologia Educacional. Processamento de Linguagem Natural. Classificação de Intenções. Chatbot Educacional. Aprendizagem Adaptativa.

ABSTRACT

The emergence of Large Language Models (LLMs) has created unprecedented opportunities to enhance educational processes, yet many higher education institutions have not fully leveraged these technologies. This thesis presents an adaptive educational chatbot system that integrates LLMs with Retrieval-Augmented Generation (RAG) to provide personalized learning assistance. The system's key innovation lies in its intent-aware approach, utilizing a custom-trained classification model based on the Snowflake/snowflake-arctic-embed-lv2.0 architecture that achieved 97.6% accuracy on the test set. This classifier, trained on a bilingual dataset (Portuguese and English), enables the system to identify student intentions and dynamically adjust its response strategy. The system also features a novel PDF processing algorithm for efficient content extraction and indexing of educational materials. The chatbot can be rapidly deployed to support various educational scenarios, from textbook comprehension to academic article analysis, requiring only the source materials in PDF format and minimal setup configuration. By combining these components, the chatbot provides context-aware responses, accurately cites source materials, and adapts its interaction style based on the identified student intent. This work demonstrates how modern LLM technology can be effectively integrated into educational tools to enhance the learning experience while maintaining practical implementation considerations.

Keywords: Large Language Models (LLMs). Retrieval-Augmented Generation (RAG). Educational Technology. Natural Language Processing. Intent Classification. Educational Chatbot. Adaptive Learning.

LISTA DE FIGURAS

Figura 2.1 – Arquitetura do modelo <i>Transformer</i> , mostrando as camadas de codificação e decodificação com os mecanismos de auto-atenção de múltiplas cabeças.	16
Figura 2.2 – Arquitetura <i>decoder-only</i> dos modelos GPT, ilustrando o mecanismo de atenção mascarada onde cada <i>token</i> atende apenas aos <i>tokens</i> anteriores na sequência.	17
Figura 2.3 – Arquitetura do AcademiCK	21
Figura 2.4 – Estrutura de um <i>prompt</i> RAG educacional, demonstrando a integração de instruções sistemáticas, contexto recuperado estruturado e consulta do usuário para otimização da qualidade da resposta.	37
Figura 3.1 – Interface do AcademiCK com Recursos Avançados	58
Figura 3.2 – Painel Administrativo do AcademiCK	60
Figura 4.1 – Médias das métricas de avaliação do sistema RAG	66
Figura 4.2 – Distribuição de consultas acima e abaixo dos limiares estabelecidos	67
Figura 4.3 – Distribuição das pontuações por métrica de avaliação	67
Figura 4.4 – Resposta do AcademiCK sobre Redes Neurais Convolucionais	70
Figura 4.5 – Resposta do AcademiCK sobre Camadas de Convolução em CNNs	71
Figura 4.6 – Resposta do AcademiCK - Código Python para Rede Neural Simples	72
Figura 4.7 – Explicação sobre Redes Neurais Convolucionais (CNNs)	73
Figura 4.8 – Explicação sobre a operação de Pooling em CNNs	74

LISTA DE SIGLAS

ANN	Approximate Nearest Neighbor (Busca Aproximada por Vizinhos Mais Próximos)
BERT	Bidirectional Encoder Representations from Transformers
BGE-M3	BAAI General Embedding Model, versão M3
CNN	Convolutional Neural Networks (Redes Neurais Convolucionais)
ColBERT	Contextualized Late Interaction over BERT
DPO	Direct Preference Optimization (Otimização Direta de Preferências)
FAISS	Facebook AI Similarity Search
GPT	Generative Pre-trained Transformer
HNSW	Hierarchical Navigable Small World
IA	Inteligência Artificial
IVF	Inverted File Index
LLM	Large Language Model (Modelo Grande de Linguagem)
LSH	Locality-Sensitive Hashing
MAP	Mean Average Precision
NDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing (Processamento de Linguagem Natural)
PDF	Portable Document Format
PQ	Product Quantization
RAG	Retrieval-Augmented Generation (Geração Aumentada por Recuperação)
RLHF	Reinforcement Learning from Human Feedback
SCADA	Supervisory Control and Data Acquisition
SVM	Support Vector Machine (Máquina de Vetores de Suporte)
TF-IDF	Term Frequency-Inverse Document Frequency
TOC	Table of Contents (Tabela de Conteúdo)
UUID	Universally Unique Identifier

SUMÁRIO

1	INTRODUÇÃO	10
1.1	Impacto dos LLMs em Diferentes Setores	10
1.2	LLMs na Educação	12
1.3	Sistemas Especializados e RAG	13
1.4	Classificação de Intenções em Sistemas Educacionais	13
1.5	Objetivos	14
1.5.1	Definição do Problema	14
1.5.2	Objetivos Específicos	14
1.5.3	Contribuições Esperadas	14
2	FUNDAMENTAÇÃO TEÓRICA	15
2.1	Grandes Modelos de Linguagem	15
2.1.1	Arquitetura <i>Transformer</i>	15
2.1.2	Modelos GPT e Arquitetura <i>Decoder-Only</i>	17
2.1.3	Processos de Treinamento de LLMs	18
2.1.4	Capacidades Emergentes dos LLMs	18
2.1.5	Limitações e Desafios dos LLMs	19
2.2	Geração Aumentada por Recuperação (RAG)	19
2.2.1	Fundamentos Teóricos e Arquitetura RAG	20
2.2.2	Formulação Matemática da RAG	20
2.2.3	Variantes Arquiteturais de RAG	21
2.2.4	Métricas de Avaliação para Sistemas RAG	22
2.2.5	Desafios e Limitações dos Sistemas RAG	23
2.2.6	RAG no Contexto Educacional	23
2.2.7	Tendências Futuras em Sistemas RAG	24
2.3	Text <i>Embeddings</i>	24
2.3.1	Definição Formal e Propriedades Matemáticas	25
2.3.2	Evolução Histórica dos Modelos de <i>Text Embeddings</i>	25
2.3.2.1	<i>Embeddings</i> de Palavras Estáticos	25
2.3.2.2	<i>Embeddings</i> Contextuais	26
2.3.2.3	<i>Embeddings</i> de Sentenças e Documentos	26
2.3.3	Modelos de <i>Embedding</i> Modernos	26
2.3.4	Métricas de Similaridade e Operações Vetoriais	27
2.3.5	Técnicas de Busca Aproximada por Vizinhos Mais Próximos	28
2.3.6	<i>Embeddings</i> Densos vs. Esparsos	28
2.3.7	Desafios e Limitações dos <i>Text Embeddings</i>	29
2.3.8	Aplicações Específicas em Sistemas RAG Educacionais	29
2.3.9	Tendências Futuras e Desenvolvimentos Emergentes	30
2.4	Processamento de Documentos PDF	30
2.4.1	Desafios no Processamento de PDFs	31
2.4.2	Abordagens para Extração de Conteúdo	31
2.4.2.1	Extração Baseada em Regras Heurísticas	31
2.4.2.2	Extração Baseada em Modelos de <i>Layout</i>	31
2.4.2.3	Extração Baseada em Tabela de Conteúdo	32

2.4.3	Segmentação de Texto (<i>Chunking</i>)	32
2.4.4	Extração de Metadados Estruturais	32
2.4.5	Qualidade e Validação da Extração	33
2.4.6	Limitações e Desafios Técnicos	33
2.4.7	Tendências e Desenvolvimentos Futuros	33
2.5	Classificação de Intenções	34
2.5.1	Taxonomias de Intenção	34
2.5.2	Técnicas de Classificação de Intenções	34
2.5.2.1	Abordagens Tradicionais	34
2.5.2.2	Abordagens Baseadas em Redes Neurais	34
2.5.2.3	<i>Fine-tuning</i> de Modelos Pré-treinados	34
2.5.3	Desafios na Classificação de Intenções	35
2.6	Assistentes Educacionais Baseados em IA	35
2.6.1	Fundamentos Pedagógicos	35
2.6.2	Tipos de Assistentes Educacionais	35
2.6.3	Impacto dos LLMs na Educação	36
2.6.4	Desafios na Implementação de Assistentes Educacionais	36
2.7	Engenharia de <i>Prompt</i>	36
2.7.1	Fundamentos da Engenharia de <i>Prompts</i>	36
2.7.2	Técnicas Avançadas de Engenharia de <i>Prompts</i>	37
2.7.2.1	<i>Prompts</i> em Poucas Etapas (<i>Few-Shot Prompting</i>)	37
2.7.2.2	<i>Prompts</i> em Cadeia de Pensamento (<i>Chain-of-Thought Prompting</i>)	38
2.7.2.3	<i>Prompts</i> Adaptados à Intenção	38
2.7.3	Otimização de Consultas para Recuperação Eficaz	38
2.8	Considerações Éticas e Limitações	39
2.8.1	Considerações Éticas	39
2.8.2	Limitações Técnicas	39
2.8.3	Mitigação e Boas Práticas	39
2.9	Conclusão	40
3	IMPLEMENTAÇÃO	41
3.1	Processamento de Documentos PDF	41
3.1.1	Estrutura Geral do <i>Pipeline</i> de Processamento	41
3.1.2	Extração e Análise da Tabela de Conteúdo	41
3.1.3	Identificação Automatizada de Capítulos	42
3.1.4	Extração de Texto com Preservação de Contexto	43
3.1.5	Segmentação em <i>Chunks</i> Semanticamente Coerentes	43
3.1.6	Metadados e Enriquecimento de <i>Chunks</i>	44
3.1.7	Desafios e Soluções Implementadas	45
3.1.7.1	Desafio de <i>Layout</i> Variável	45
3.1.7.2	Desafio de Fronteiras Semânticas	45
3.1.8	Processamento em Lote e Gerenciamento de Memória	45
3.2	Classificação de Intenções	46
3.2.1	Taxonomia de Intenções no Contexto Educacional	46
3.2.2	Conjunto de Dados para Treinamento	46
3.2.2.1	Construção de exemplos iniciais	47
3.2.2.2	Aumento de Dados via LLMs	47
3.2.2.3	Balanceamento do Conjunto de Dados	47
3.2.2.4	Validação Multilíngue	47

3.2.3	Arquitetura do Modelo de Classificação	47
3.2.4	Processo de Treinamento via <i>Huggingface AutoTrain</i>	47
3.2.5	Implementação do Classificador no Sistema	48
3.2.6	Integração com o Sistema RAG	49
3.3	Geração Aumentada por Recuperação (RAG)	50
3.3.1	Arquitetura Geral do Sistema RAG	51
3.3.2	Geração e Indexação de <i>Embeddings</i> com BGE-M3	51
3.3.3	Estratégia de Recuperação Ponderada	52
3.3.4	Otimização de Consultas para Recuperação	53
3.3.5	Geração de <i>Prompts</i> Adaptados por Intenção	53
3.3.6	Fluxo de Processamento Integrado	54
3.3.7	Gestão de Contexto Conversacional	55
3.3.8	Considerações sobre Escalabilidade e Desempenho	55
3.3.9	Arquitetura do Servidor e Interface de Usuário	56
3.3.10	Tratamento de Multilinguismo	57
3.3.11	Adaptabilidade a Diferentes Domínios de Conhecimento	59
3.3.12	Integração com Interface Administrativa	59
3.3.13	Considerações Éticas na Implementação	61
3.3.14	Síntese Metodológica	61
4	RESULTADOS	63
4.1	Avaliação do Sistema RAG	63
4.2	Métricas de Avaliação	63
4.3	Resultados Quantitativos	64
4.3.1	Resultados do Classificador de Intenções	64
4.3.2	Resultados do sistema	65
4.4	Resultados Qualitativos	66
4.4.1	Desempenho por Categoria de Intenção	67
4.4.1.1	Busca por Informações (<i>Searching for Information</i>)	68
4.4.1.2	Resposta a Perguntas (<i>Question Answering</i>)	68
4.4.1.3	Programação (<i>Coding</i>)	68
4.4.1.4	Sumarização (<i>Summarization</i>)	68
4.4.2	Capacidade de Contextualização e Aprofundamento	68
4.4.3	Qualidade das Citações e Referenciamento	69
4.4.4	Adaptação Linguística e Terminológica	69
5	CONCLUSÃO	75
5.1	Contribuições Principais	75
5.2	Resultados Alcançados	75
5.3	Implicações Práticas	76
5.4	Limitações e Trabalhos Futuros	76
5.5	Considerações Finais	77
	REFERÊNCIAS	79

1 INTRODUÇÃO

Os últimos anos testemunharam uma revolução sem precedentes no campo de Inteligência Artificial (IA), marcada principalmente pelo surgimento e evolução dos Grandes Modelos de Linguagem (*Large Language Models - LLMs*). Estes modelos representam um dos mais significativos avanços tecnológicos do século XXI, transformando fundamentalmente a maneira como interagimos com a tecnologia e processamos informações. A capacidade desses sistemas de compreender e gerar linguagem natural de forma contextualmente apropriada tem revolucionado diversos setores da sociedade, desde a automação industrial até a pesquisa científica avançada (HAQUE; LI, 2024).

Os LLMs são sistemas de IA fundamentados em arquiteturas neurais complexas, especificamente baseados no mecanismo de atenção e na arquitetura *Transformer* Vaswani et al. (2017), que revolucionou o processamento de linguagem natural. Estes modelos são treinados em vastos conjuntos de dados textuais que abrangem praticamente todo o conhecimento humano disponível digitalmente, incluindo literatura científica, documentos técnicos, conteúdo educacional e discussões online. O processo de treinamento envolve a análise de padrões linguísticos em diferentes níveis de abstração, desde a estrutura sintática básica até nuances semânticas complexas. Esta abordagem permite que os modelos desenvolvam uma compreensão sofisticada da linguagem humana, capacitando-os a realizar tarefas que vão desde a simples compreensão de texto até o raciocínio analítico complexo (HUANG et al., 2023).

A evolução destes modelos tem seguido uma trajetória extraordinária, marcada por avanços significativos em intervalos cada vez menores como mostrado em MINAEE et al. (2024). O surgimento do BERT (*Bidirectional Encoder Representations from Transformers*) em 2018 pela Google em DELVIN et al. (2019) representou um marco inicial, introduzindo o conceito de compreensão bidirecional do contexto. O GPT-3 (*Generative Pre-trained Transformer 3*) BROWN et al. (2020), lançado pela OpenAI em 2020, com seus 175 bilhões de parâmetros, estabeleceu um novo paradigma em escala e capacidade. Em 2022, o lançamento do GPT-4 ACHIAM et al. (2023) elevou ainda mais os padrões de desempenho, demonstrando capacidades quase humanas em uma ampla gama de tarefas. Um desenvolvimento particularmente interessante ocorreu com o surgimento do DeepSeek r1 GUO et al. (2025), que conseguiu superar o modelo o1 da OpenAI OPENAI (2024) com um custo computacional significativamente menor, demonstrando que a eficiência e otimização dos modelos podem ser tão importantes quanto sua escala. Este avanço representou um passo significativo na democratização da tecnologia de IA, mostrando que é possível desenvolver modelos de alto desempenho com recursos mais limitados.

1.1 Impacto dos LLMs em Diferentes Setores

O impacto desta tecnologia tem se manifestado de maneira profunda e transformadora em diversos setores da sociedade. No setor industrial, os LLMs têm demonstrado um potencial transformador, atuando como catalisadores da nova era da manufatura inteligente. Esta tecnologia está revolucionando múltiplas fases do processo produtivo, desde a pesquisa e desenvolvimento até o controle operacional, possibilitando a análise de grandes volumes de dados para otimização de processos, detecção precoce de falhas e automatização da documentação técnica. A integração dos LLMs nos sistemas de produção tem permitido não apenas a interpretação mais eficiente de dados operacionais, mas também a geração

de *insights* valiosos para tomada de decisões em tempo real. Em sistemas SCADA, por exemplo, os LLMs estão sendo utilizados para desenvolver códigos de lógica industrial e protocolos de interação, elevando significativamente o nível de inteligência dos processos produtivos (FENG et al., 2024). Esta evolução marca uma transição fundamental na indústria, onde interfaces baseadas em linguagem natural estão gradualmente substituindo os métodos tradicionais de controle e documentação, promovendo uma integração mais fluida entre humanos e sistemas automatizados.

No setor de saúde, como mostrado em YANG et al. (2023), os LLMs têm demonstrado potencial transformador em múltiplas frentes da jornada do paciente. Na fase de pré-consulta, estes modelos auxiliam na triagem de pacientes e direcionamento eficiente de recursos, além de facilitar diagnósticos preliminares remotos. Durante o processo de diagnóstico, os LLMs se mostram particularmente úteis na geração de resumos concisos do histórico médico do paciente, incluindo comorbidades, consultas anteriores e respostas a tratamentos, permitindo consultas mais eficientes e abrangentes. Na gestão do tratamento, estas tecnologias têm revolucionado a educação do paciente através de conteúdo personalizado e linguagem acessível, melhorando a compreensão e adesão ao tratamento. Além disso, os LLMs estão transformando a educação médica e a pesquisa científica, sendo capazes de passar em exames médicos e auxiliar na redação de artigos científicos, sempre sob supervisão humana. Na área administrativa, têm otimizado processos como a elaboração de resumos de alta, relatórios operatórios e processamento de reembolsos médicos.

O setor financeiro tem experimentado transformações significativas com a integração de LLMs. Conforme LI et al. (2023a), estas tecnologias estão sendo implementadas em aplicações críticas como análise de risco, detecção de fraudes bancárias e serviços financeiros automatizados. Os modelos têm demonstrado capacidades significativas no processamento de linguagem natural para tarefas específicas do setor, como análise de sentimento de mercado, classificação de notícias financeiras e geração de respostas para consultas técnicas. No entanto, os autores destacam que existem limitações importantes a serem consideradas, incluindo desafios relacionados à produção de desinformação e manifestação de vieses nos modelos. Embora os LLMs ofereçam maior interpretabilidade comparada a modelos convencionais de *deep learning*, fornecendo passos de raciocínio quando adequadamente instruídos, eles ainda apresentam limitações significativas quanto à explicabilidade do conteúdo gerado.

De acordo com um extenso estudo com 816 pesquisadores, em LIAO et al. (2024), os LLMs têm demonstrado um impacto significativo na pesquisa científica, com aproximadamente 81% dos pesquisadores já incorporando estas ferramentas em seus fluxos de trabalho. A adoção mais expressiva concentra-se em tarefas relacionadas à preparação de manuscritos e revisão de literatura, com 49% e 45% dos pesquisadores utilizando LLMs para busca de informações e edição, respectivamente. Estas ferramentas têm se mostrado particularmente valiosas para promover equidade na pesquisa, auxiliando pesquisadores não nativos do inglês a superar barreiras linguísticas e permitindo que pesquisadores com recursos limitados acessem suporte de alta qualidade para escrita e análise. No entanto, a comunidade científica expressa preocupações significativas quanto aos riscos de alucinações e desinformação como mostrado em JI et al. (2023), bem como o potencial impacto na integridade da pesquisa e na criatividade dos pesquisadores. Em campos que envolvem análise de dados, como limpeza e geração de dados, a adoção tem sido mais cautelosa, com 69% e 73% dos pesquisadores, respectivamente, relatando nunca terem utilizado LLMs para estas finalidades.

Os LLMs têm revolucionado o campo jurídico através de diversas aplicações, desde a extração de elementos legais chave de documentos até a assistência em decisões

judiciais. A capacidade destes modelos de processar grandes volumes de textos legais não estruturados permite que profissionais do direito realizem consultas jurídicas de forma mais eficiente, auxiliem juízes na tomada de decisões e automatizem a geração de documentos legais. Uma característica particularmente relevante é a habilidade dos LLMs em compreender e estabelecer relações contextuais, permitindo a análise de casos similares e fornecendo embasamento para decisões legais mais consistentes (LAI et al., 2024). Esta transformação tem otimizado significativamente o trabalho jurídico, embora ainda existam desafios importantes relacionados à interpretabilidade dos modelos e à necessidade de preservar a independência judicial.

1.2 LLMs na Educação

No contexto educacional, o impacto dos LLMs tem sido particularmente profundo e multifacetado, como evidenciado em estudos recentes sobre a integração dessas tecnologias no ensino superior JUMAR et al. (2024). Esta transformação está redefinindo fundamentalmente os paradigmas tradicionais de ensino e aprendizagem, com os LLMs oferecendo possibilidades anteriormente inimagináveis, desde explicações detalhadas e exemplos personalizados até *feedback* instantâneo, criando assim novas oportunidades para o aprendizado personalizado em escala. As instituições de ensino superior encontram-se em um momento crucial de adaptação, onde professores descobrem maneiras inovadoras de utilizar estas ferramentas para criar material didático, desenvolver planos de aula mais eficazes e fornecer suporte individualizado aos estudantes. No entanto, esta mesma tecnologia também suscita questões importantes sobre integridade acadêmica e autenticidade do trabalho dos alunos, evidenciando a necessidade de um equilíbrio entre inovação e responsabilidade ética no ambiente acadêmico.

Um dos aspectos mais promissores dos LLMs no contexto educacional é sua capacidade de atuar como tutores virtuais disponíveis 24 horas por dia como mostrado em HAN et al. (2024). Estes sistemas podem adaptar suas explicações ao nível de compreensão do aluno, oferecendo múltiplas perspectivas sobre um mesmo conceito e auxiliando no processo de construção do conhecimento. Estudantes podem explorar tópicos complexos em seu próprio ritmo, fazer perguntas sem constrangimento e receber *feedback* imediato sobre seu entendimento. Esta disponibilidade constante de suporte educacional personalizado representa uma democratização significativa do acesso ao conhecimento.

No entanto, a integração efetiva destas tecnologias no ambiente educacional enfrenta desafios significativos (HWANG et al., 2020). Um dos principais obstáculos é a necessidade de contextualização precisa do conhecimento. Embora os LLMs possuam uma base de conhecimento impressionante, sua capacidade de fazer referências específicas a materiais didáticos particulares ou seguir metodologias de ensino específicas ainda é limitada. Além disso, existe o risco de os modelos fornecerem informações imprecisas ou desatualizadas, especialmente em campos altamente especializados ou em rápida evolução (LEWIS et al., 2020).

A questão da acessibilidade linguística também emerge como um aspecto crucial, particularmente em contextos educacionais multilíngues. Enquanto a maioria dos LLMs demonstra forte competência em inglês, sua eficácia em outros idiomas pode variar significativamente. Em países como o Brasil, onde o material didático e as interações acadêmicas frequentemente transitam entre o português e o inglês, a necessidade de sistemas que possam operar eficientemente em ambos os idiomas torna-se especialmente relevante (SHEN et al., 2024).

Além disso, o processo de aprendizagem frequentemente requer diferentes abordagens pedagógicas dependendo do objetivo do estudante. Um aluno pode estar buscando uma explicação conceitual, tentando resolver um problema específico, ou procurando relacionar diferentes aspectos do conteúdo. Os LLMs tradicionais, embora versáteis, nem sempre são capazes de identificar automaticamente estas diferentes necessidades e adaptar suas respostas de acordo.

1.3 Sistemas Especializados e RAG

A Geração Aumentada por Recuperação (RAG), como apresentada em LEWIS et al. (2020), representa uma evolução significativa na forma como os Grandes Modelos de Linguagem podem ser aplicados em contextos específicos. Esta abordagem combina a capacidade de geração de texto dos LLMs com um sistema de acesso a informações específicas, permitindo que o modelo fundamente suas respostas em fontes confiáveis e atualizadas.

Diferentemente dos LLMs tradicionais, que dependem exclusivamente do conhecimento adquirido durante seu treinamento, sistemas baseados em RAG podem acessar e incorporar informações de bases de conhecimento externas em tempo real. Esta característica é particularmente valiosa no contexto educacional, onde a precisão e a capacidade de citar fontes específicas são fundamentais para o processo de aprendizagem. A necessidade de contextualização emerge como um aspecto central nestes sistemas. No ambiente educacional, não basta fornecer respostas genéricas; é essencial que estas estejam alinhadas com o material didático específico utilizado no curso e com o contexto de aprendizagem do estudante. A RAG possibilita este alinhamento ao permitir que o sistema acesse e utilize diretamente os materiais de estudo como fonte de informação.

A importância da precisão nas citações se torna um diferencial significativo dos sistemas RAG. Ao fornecer respostas, o sistema pode indicar precisamente onde as informações podem ser encontradas no material de estudo, criando uma ponte direta entre a interação com o aluno e as fontes de conhecimento. Esta capacidade não apenas aumenta a confiabilidade das respostas, mas também incentiva os estudantes a desenvolverem habilidades de pesquisa e aprofundamento nos materiais originais.

1.4 Classificação de Intenções em Sistemas Educacionais

A eficácia de um sistema educacional baseado em LLMs está diretamente relacionada à sua capacidade de compreender e adaptar-se aos diferentes objetivos de aprendizagem dos estudantes. A classificação de intenções emerge como uma ferramenta fundamental neste contexto, permitindo que o sistema identifique automaticamente o propósito da interação do estudante e ajuste sua resposta de acordo. Diferentes objetivos de aprendizagem caracterizam as interações dos estudantes com sistemas educacionais. Um aluno pode estar buscando uma explicação conceitual sobre um tópico específico, tentando resolver um problema prático, procurando referências bibliográficas ou buscando estabelecer conexões entre diferentes conceitos. A capacidade de identificar corretamente estas diferentes intenções é crucial para fornecer o suporte adequado (SOLOMIN; IVANOVA, 2020).

A adaptação de respostas baseada na intenção identificada permite que o sistema forneça informações no formato mais apropriado para cada situação. Por exemplo, quando um estudante busca compreender um conceito teórico, o sistema pode priorizar explicações

detalhadas e exemplos ilustrativos. Em contraste, quando a intenção é resolver um problema específico, o foco pode ser direcionado para métodos de solução e casos práticos.

O desafio de identificar corretamente as intenções dos estudantes é particularmente complexo em contextos educacionais, onde as necessidades de aprendizagem podem ser sutis e multifacetadas. A utilização de modelos de classificação especializados, treinados com dados específicos do contexto educacional, representa um avanço significativo na capacidade de sistemas automatizados compreenderem e responderem adequadamente às necessidades dos estudantes.

1.5 Objetivos

1.5.1 Definição do Problema

Este trabalho propõe o desenvolvimento de um sistema educacional adaptativo que integra tecnologias de LLMs e RAG, com foco na personalização das interações através da classificação de intenções. O problema abordado é a necessidade de criar uma solução tecnológica que supere as limitações dos atuais sistemas de assistência educacional, oferecendo não apenas informações precisas e contextualizadas sobre o material didático, mas também adaptando suas respostas conforme os diferentes objetivos e necessidades dos estudantes em seus processos de aprendizagem.

1.5.2 Objetivos Específicos

- Desenvolver um sistema de chatbot educacional que integre LLMs com RAG para fornecer assistência personalizada ao aprendizado;
- Criar e implementar um classificador de intenções especializado para o contexto educacional;
- Estabelecer um método eficiente de processamento e indexação de materiais didáticos em formato PDF;
- Implementar um sistema multilíngue capaz de operar efetivamente em português e inglês;
- Avaliar a eficácia do sistema em um contexto educacional real.

1.5.3 Contribuições Esperadas

A principal contribuição deste trabalho é o desenvolvimento de um sistema adaptativo que demonstra como a moderna tecnologia de LLMs pode ser efetivamente integrada em ferramentas educacionais. O sistema proposto inova ao combinar classificação de intenções com RAG, permitindo respostas mais precisas e contextualizadas. A abordagem multilíngue e o algoritmo especializado de processamento de PDFs representam avanços na aplicabilidade prática destes sistemas.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os fundamentos teóricos necessários para a compreensão do sistema de assistente educacional baseado em IA desenvolvido neste trabalho. São abordados os principais conceitos e tecnologias que sustentam o sistema, incluindo LLMs, RAG, incorporações de texto (*embeddings*), processamento de documentos PDF, classificação de intenções e arquiteturas de assistentes educacionais baseados em IA.

2.1 Grandes Modelos de Linguagem

Os Grandes Modelos de Linguagem, conhecidos na literatura internacional como *Large Language Models* (LLMs), representam uma classe de sistemas de inteligência artificial fundamentados em arquiteturas de redes neurais profundas, especificamente na arquitetura *Transformer*. Estes modelos são treinados em vastos *corpora* de texto para aprender representações estatísticas complexas da linguagem, demonstrando capacidades emergentes de compreensão e geração de linguagem natural (BROWN et al., 2020).

Os LLMs constituem uma evolução significativa no campo do processamento de linguagem natural, caracterizando-se por sua escala sem precedentes tanto em número de parâmetros quanto em volume de dados de treinamento. Esta escala massiva permite que os modelos desenvolvam capacidades emergentes, incluindo raciocínio em contexto, generalização para tarefas não vistas durante o treinamento e adaptação a diferentes domínios de conhecimento sem necessidade de *fine-tuning* específico (MINAEE et al., 2024).

2.1.1 Arquitetura *Transformer*

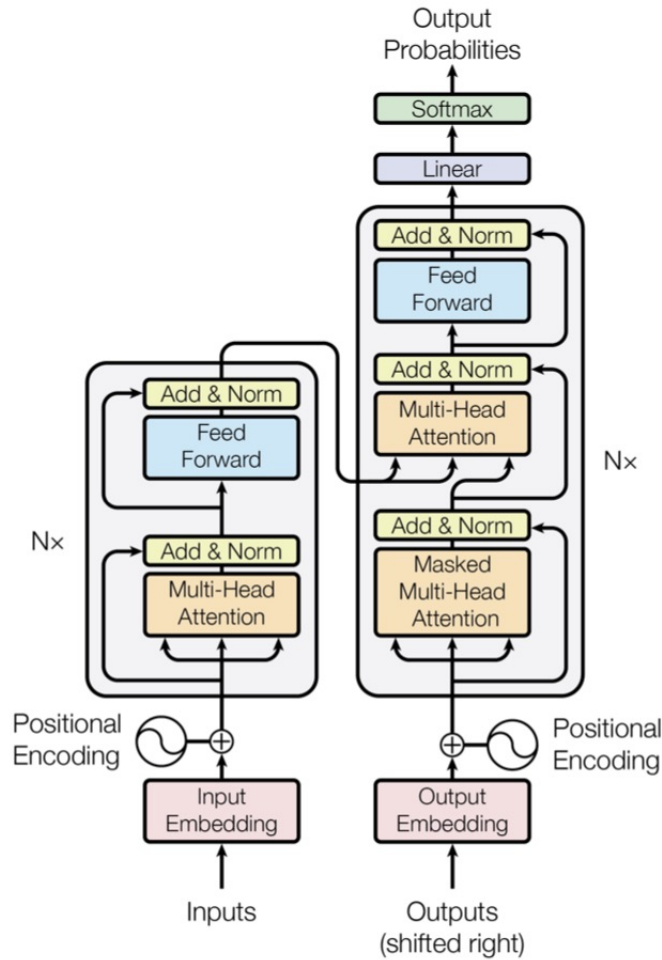
A arquitetura *Transformer*, introduzida por Vaswani et al. (2017), representa um marco fundamental na evolução dos modelos de linguagem, revolucionando o campo do processamento de linguagem natural ao substituir as arquiteturas recorrentes tradicionais por mecanismos de atenção. A principal inovação desta arquitetura é o mecanismo de auto-atenção (*self-attention*), que permite ao modelo processar todos os elementos de uma sequência simultaneamente, capturando eficientemente dependências de longo alcance sem as limitações inerentes ao processamento sequencial.

A arquitetura básica do *Transformer* é composta por camadas de codificação (*encoder*) e decodificação (*decoder*). Cada camada contém *sublayers* de auto-atenção de múltiplas cabeças (*multi-head self-attention*) e redes neurais *feedforward* posicionalmente conectadas, com conexões residuais e normalização de camadas aplicadas em torno de cada *sublayer*. A Figura 2.1 ilustra esta arquitetura fundamental.

O mecanismo de auto-atenção constitui o núcleo da arquitetura *Transformer* e é definido matematicamente como:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.1)$$

onde Q (*query*/consulta), K (*key*/chave) e V (*value*/valor) são matrizes derivadas das representações de entrada através de transformações lineares aprendidas, e d_k representa a dimensão das chaves, utilizada para normalização e estabilização do gradiente. Esta



Fonte: Vaswani et al. (2017).

Figura 2.1 – Arquitetura do modelo *Transformer*, mostrando as camadas de codificação e decodificação com os mecanismos de auto-atenção de múltiplas cabeças.

formulação permite que cada posição na sequência “consulte” todas as outras posições, determinando dinamicamente quais elementos são mais relevantes para o processamento.

As matrizes de consulta, chave e valor são obtidas através de projeções lineares da matriz de entrada X :

$$\begin{aligned} Q &= XW^Q \\ K &= XW^K \\ V &= XW^V \end{aligned} \quad (2.2)$$

onde W^Q , W^K e W^V são matrizes de pesos aprendidas durante o processo de treinamento. Estas projeções permitem que o modelo aprenda diferentes tipos de relações entre os elementos da sequência.

A arquitetura implementa atenção de múltiplas cabeças (*multi-head attention*) para capturar diferentes tipos de dependências e relações semânticas em paralelo:

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O \quad (2.3)$$

onde cada cabeça de atenção head_i é computada independentemente:

$$\text{head}_i = \text{Attention}(XW_i^Q, XW_i^K, XW_i^V) \quad (2.4)$$

e W^O é uma matriz de projeção final que combina as saídas de todas as cabeças. Este mecanismo permite que o modelo capture simultaneamente diferentes tipos de relações linguísticas, desde dependências sintáticas até associações semânticas complexas.

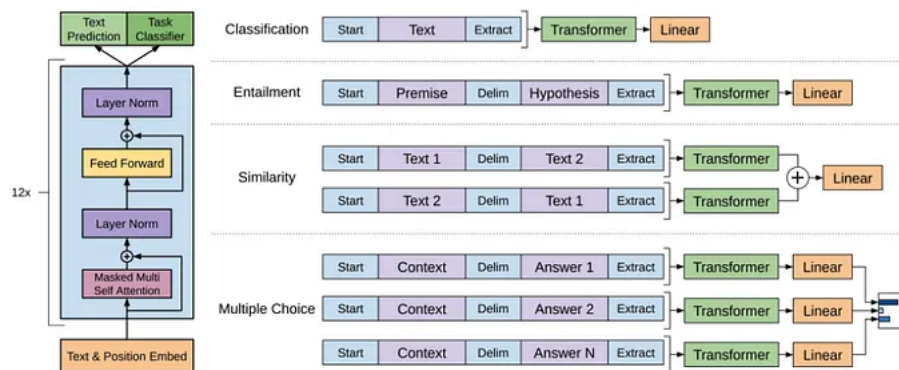
2.1.2 Modelos GPT e Arquitetura *Decoder-Only*

Os modelos GPT (*Generative Pre-trained Transformer*) representam uma classe específica de LLMs baseados exclusivamente na arquitetura de decodificação (*decoder-only*) do *Transformer* original, como mostrado na Figura 2.2. Esta escolha arquitetônica, inicialmente proposta por (BROWN et al., 2020), demonstrou-se particularmente eficaz para tarefas de geração de linguagem e modelagem autorregressiva.

A arquitetura *decoder-only* dos modelos GPT é otimizada para a predição sequencial de *tokens*, onde o modelo aprende a prever o próximo *token* em uma sequência condicionado exclusivamente nos *tokens* precedentes. Durante o pré-treinamento, o modelo é exposto a enormes quantidades de texto e aprende a minimizar a função de perda de verossimilhança negativa logarítmica:

$$\mathcal{L}(\theta) = - \sum_{i=1}^n \log P_{\theta}(x_i | x_{<i}) \quad (2.5)$$

onde θ representa o conjunto completo de parâmetros do modelo, x_i é o *token* na posição i , e $x_{<i}$ denota todos os *tokens* anteriores na sequência. Esta função objetivo incentiva o modelo a atribuir alta probabilidade aos *tokens* corretos dado o contexto precedente, desenvolvendo implicitamente representações ricas da estrutura e semântica da linguagem.



Fonte: R. (2023).

Figura 2.2 – Arquitetura *decoder-only* dos modelos GPT, ilustrando o mecanismo de atenção mascarada onde cada *token* atende apenas aos *tokens* anteriores na sequência.

Uma característica fundamental dos modelos *decoder-only* é a implementação de máscara de atenção (*masked attention*), que garante que cada *token* possa “atender” apenas aos *tokens* anteriores na sequência, preservando a natureza autorregressiva da modelagem. Esta restrição é implementada através da aplicação de uma máscara na matriz de atenção:

$$\text{MaskedAttention}(Q, K, V) = \text{softmax} \left(\frac{QK^T + M}{\sqrt{d_k}} \right) V \quad (2.6)$$

onde M é uma matriz triangular superior com valores $-\infty$ nas posições futuras (que devem ser mascaradas) e 0 nas posições passadas e presente (que são permitidas). Esta abordagem impede que o modelo “veja” *tokens* futuros durante o treinamento, mantendo a integridade do processo de predição sequencial.

2.1.3 Processos de Treinamento de LLMs

O treinamento de LLMs modernos é tipicamente estruturado em múltiplas fases sequenciais, cada uma com objetivos específicos para otimizar diferentes aspectos do comportamento do modelo:

1. **Pré-treinamento (*Pre-training*)**: Nesta fase inicial, o modelo é treinado em *corpora* massivos de texto não estruturado utilizando objetivos de modelagem de linguagem, como predição do próximo *token*. Esta fase desenvolve as capacidades fundamentais de compreensão e geração de linguagem, estabelecendo representações internas robustas de conhecimento factual, padrões linguísticos e estruturas semânticas.
2. ***Fine-tuning* Supervisionado (*Supervised Fine-tuning*)**: O modelo pré-treinado é refinado utilizando conjuntos de dados criteriosamente selecionados, contendo exemplos de entrada-saída para tarefas específicas. Esta fase adapta as capacidades gerais do modelo para comportamentos mais direcionados e alinhados com aplicações práticas, melhorando a qualidade e relevância das respostas geradas.
3. **Alinhamento com Preferências Humanas**: Técnicas avançadas como RLHF (*Reinforcement Learning from Human Feedback*) ou DPO (*Direct Preference Optimization*) são aplicadas para alinhar o comportamento do modelo com valores, preferências e expectativas humanas. Esta fase é crucial para desenvolver modelos que sejam não apenas tecnicamente competentes, mas também seguros, úteis e eticamente apropriados.

O processo de alinhamento merece atenção especial, pois é fundamental para o desenvolvimento de assistentes virtuais eficazes. Como demonstrado por GUO et al. (2025), OPENAI (2024) e OUYANG et al. (2022), técnicas de alinhamento melhoram significativamente a utilidade prática, segurança operacional e conformidade com intenções humanas, resultando em sistemas mais confiáveis e apropriados para aplicações educacionais.

2.1.4 Capacidades Emergentes dos LLMs

Os LLMs modernos demonstram um conjunto impressionante de capacidades emergentes que surgem da escala massiva de treinamento, incluindo:

- **Geração de Texto Contextualmente Coerente**: Capacidade de produzir texto que mantém coerência temática, estilística e factual ao longo de extensas sequências.
- **Compreensão e Resposta a Consultas Complexas**: Habilidade de interpretar perguntas multifacetadas e fornecer respostas estruturadas e abrangentes.
- **Sumarização Inteligente**: Capacidade de extrair e condensar informações-chave de textos extensos mantendo fidelidade ao conteúdo original.
- **Tradução e Processamento Multilíngue**: Competência em múltiplos idiomas com capacidade de tradução e compreensão cross-linguística.
- **Análise e Geração de Código**: Habilidades sofisticadas de programação, incluindo interpretação, depuração e geração de código em múltiplas linguagens.
- **Raciocínio Analítico**: Capacidade de realizar inferências lógicas, resolver problemas abstratos e realizar raciocínio de senso comum.

- **Adaptação *Few-Shot*:** Habilidade de se adaptar a novas tarefas com apenas alguns exemplos, sem necessidade de retreinamento.

Estas capacidades têm possibilitado aplicações transformadoras dos LLMs em diversos setores, incluindo saúde (YANG et al., 2023), finanças (LI et al., 2023a), manufatura (FENG et al., 2024), direito (LAI et al., 2024) e pesquisa científica (LIAO et al., 2024), demonstrando o potencial revolucionário desta tecnologia.

2.1.5 Limitações e Desafios dos LLMs

Apesar de suas capacidades impressionantes, os LLMs apresentam limitações significativas que devem ser cuidadosamente consideradas, especialmente em aplicações educacionais críticas:

- **Alucinações:** Tendência dos modelos a gerar informações factualmente incorretas ou fabricadas, apresentadas com alta confiança aparente. Este fenômeno é particularmente problemático em contextos educacionais onde a precisão factual é fundamental.
- **Conhecimento Estático e Limitado:** O conhecimento dos modelos é restrito aos dados de treinamento, sem acesso a informações atualizadas ou especializadas não presentes no corpus de treinamento. Esta limitação é especialmente relevante para campos em rápida evolução ou domínios altamente especializados.
- **Ausência de Contexto Específico:** Incapacidade de acessar materiais didáticos específicos, bases de conhecimento proprietárias ou informações contextuais relevantes para aplicações especializadas.
- **Limitações em Raciocínio Matemático Rigoroso:** Dificuldades com cálculos complexos, problemas matemáticos que exigem múltiplas etapas de raciocínio sequencial ou verificação formal de provas matemáticas.
- **Falta de Consciência Epistêmica:** Dificuldade em expressar adequadamente incerteza, reconhecer limitações de conhecimento ou distinguir entre diferentes níveis de confiança em suas respostas.
- **Sensibilidade a Formulação de *Prompts*:** Variabilidade significativa na qualidade das respostas dependendo de como as consultas são formuladas, exigindo engenharia cuidadosa de *prompts* para resultados consistentes.
- **Vieses Sistêmicos:** Reflexão de vieses presentes nos dados de treinamento, potencialmente perpetuando estereótipos ou preconceitos indesejáveis (SHEN et al., 2024).

Estas limitações fundamentais motivam fortemente a adoção de abordagens híbridas, como sistemas RAG, que combinam as capacidades generativas dos LLMs com sistemas de recuperação de informação estruturados e confiáveis. Tais abordagens permitem mitigar muitas das limitações inerentes aos LLMs puros, fornecendo uma base mais sólida para aplicações educacionais onde precisão, confiabilidade e rastreabilidade são requisitos essenciais.

A compreensão tanto das capacidades quanto das limitações dos LLMs é fundamental para o desenvolvimento de sistemas de apoio a educação eficazes que aproveitem suas forças enquanto compensam suas fraquezas através de arquiteturas e metodologias apropriadas.

2.2 Geração Aumentada por Recuperação (RAG)

A Geração Aumentada por Recuperação, conhecida na literatura internacional como *Retrieval-Augmented Generation* (RAG), representa uma abordagem híbrida inovadora que combina sistemas de recuperação de informação com modelos de linguagem

generativos. Esta técnica, introduzida por (LEWIS et al., 2020), endereça diretamente as limitações fundamentais dos LLMs ao fornecer informações contextuais relevantes e atualizadas para fundamentar o processo de geração de respostas.

O paradigma RAG surge como uma solução elegante para os desafios inerentes aos LLMs puros, particularmente no contexto educacional onde a precisão factual, a capacidade de citação de fontes e o acesso a materiais didáticos específicos são requisitos essenciais. Ao integrar capacidades de recuperação de informação com geração de linguagem natural, A RAG permite que sistemas baseados em LLMs acessem e incorporem conhecimento externo de forma dinâmica, superando as limitações de conhecimento estático e reduzindo significativamente o fenômeno de alucinações (GAO et al., 2023).

2.2.1 Fundamentos Teóricos e Arquitetura RAG

A arquitetura RAG baseia-se na premissa fundamental de que a qualidade da geração de texto pode ser significativamente melhorada através do fornecimento de contexto relevante e factualmente preciso. O sistema opera através da integração coordenada de três componentes principais, cada um desempenhando um papel crítico no *pipeline* de processamento.

O componente de indexação é responsável pelo processamento *offline* de documentos fonte, convertendo-os em representações vetoriais (*embeddings*) que são armazenadas em um índice estruturado. Este processo envolve a segmentação de documentos, a geração de *embeddings* semânticos e a organização eficiente das representações para recuperação posterior. A qualidade desta fase determina fundamentalmente a eficácia de todo o sistema RAG.

O componente de recuperação constitui a interface entre as consultas dos usuários e a base de conhecimento indexada. Recebe consultas em linguagem natural, converte-as em representações vetoriais compatíveis e executa buscas por similaridade semântica para identificar os documentos ou passagens mais relevantes. Este componente implementa algoritmos sofisticados de busca vetorial e técnicas de *re-ranking* para otimizar a qualidade da recuperação.

O componente de geração integra um LLM que utiliza tanto a consulta original do usuário quanto as informações recuperadas para gerar respostas coerentes, precisas e contextualmente apropriadas. Este componente é responsável pela síntese das informações recuperadas, pela manutenção da coerência textual e pela geração de respostas que atendam adequadamente às necessidades do usuário.

A Figura 2.3 ilustra a arquitetura integrada destes componentes e o fluxo de processamento característico de sistemas RAG.

2.2.2 Formulação Matemática da RAG

O processo RAG pode ser formalizado matematicamente através de uma abordagem probabilística que modela a geração de respostas como uma distribuição condicionada tanto na consulta quanto no contexto recuperado. A formulação fundamental, conforme apresentada por LEWIS et al. (2020), é expressa como o teorema da probabilidade total para probabilidades condicionais:

$$P(y|x) = \sum_{z \in \mathcal{Z}} P(y|x,z) \cdot P(z|x) \quad (2.7)$$

onde os componentes desta formulação representam:

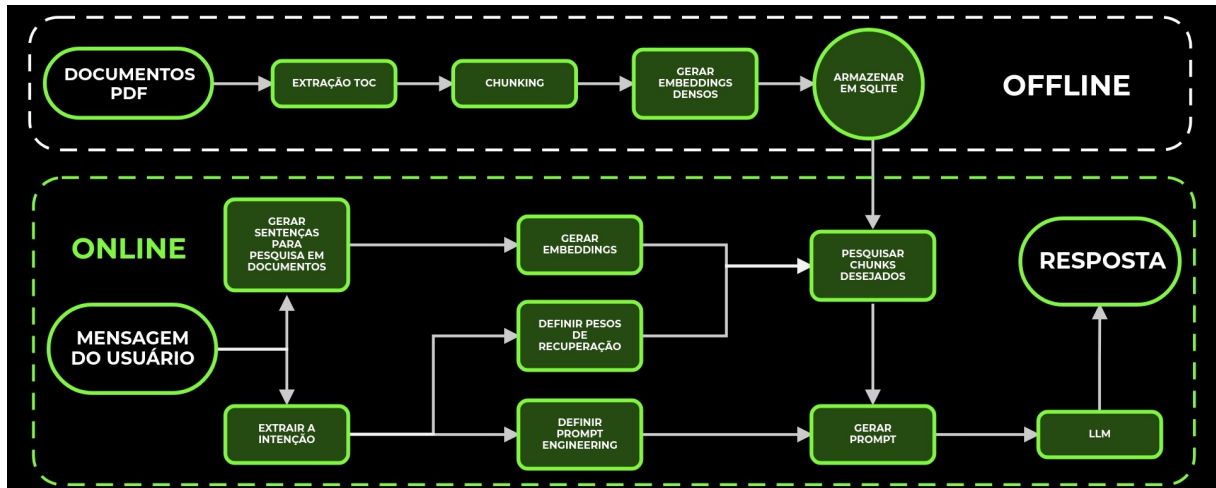


Figura 2.3 – Arquitetura do AcademiCK

- x denota a consulta original do usuário em linguagem natural
- y representa a resposta gerada pelo sistema
- z corresponde a um documento ou passagem específica recuperada do corpus \mathcal{Z}
- $P(z|x)$ expressa a probabilidade de recuperação, determinada pelo sistema de recuperação baseado em similaridade semântica
- $P(y|x, z)$ representa a probabilidade de geração condicionada, determinada pelo LLM utilizando tanto a consulta quanto o contexto recuperado

Na implementação prática, para uma consulta de usuário x , o sistema RAG executa um processo estruturado que primeiro recupera os k documentos mais semanticamente relevantes $\{z_1, z_2, \dots, z_k\}$ utilizando métricas de similaridade vetorial. Estes documentos são então organizados e formatados em um contexto estruturado c que tipicamente inclui a consulta original e as passagens recuperadas:

$$c = \text{format}(x, z_1, z_2, \dots, z_k) \quad (2.8)$$

O LLM subsequentemente gera uma resposta y condicionada neste contexto enriquecido:

$$y \sim P_{\text{LLM}}(y|c) \quad (2.9)$$

Esta abordagem permite que o modelo acesse informações específicas e atualizadas que não estavam presentes em seus dados de treinamento, expandindo significativamente suas capacidades e melhorando a precisão factual das respostas geradas.

2.2.3 Variantes Arquiteturais de RAG

A evolução dos sistemas RAG tem resultado no desenvolvimento de múltiplas variantes arquiteturais, cada uma otimizada para diferentes tipos de aplicações e requisitos específicos. A literatura apresenta diversas taxonomias destas variantes, conforme sistematizado por GAO et al. (2023):

- **RAG Básico (*Naive RAG*):** Implementa o fluxo fundamental de recuperação seguida de geração, adequado para aplicações simples com requisitos básicos de precisão e relevância.
- **RAG Iterativo (*Iterative RAG*):** Executa múltiplas rodadas de recuperação durante o processo de geração, permitindo refinamento progressivo das informações recuperadas e melhoria da qualidade da resposta final.

- **RAG Adaptativo (*Adaptive RAG*)**: Implementa ajuste dinâmico da estratégia de recuperação baseado na complexidade, tipo e contexto da consulta, otimizando recursos computacionais e qualidade da resposta.
- **RAG Hierárquico (*Hierarchical RAG*)**: Realiza recuperação em múltiplos níveis de granularidade, começando com recuperação de documentos de alto nível seguida de recuperação de passagens específicas.
- **RAG Recursivo (*Recursive RAG*)**: Decompõe consultas complexas em subconsultas mais simples, executando recuperação específica para cada componente antes da síntese final.
- **RAG com *Re-ranking***: Incorpora estágios adicionais de re-ordenação dos documentos recuperados utilizando modelos especializados para melhorar a relevância do contexto fornecido ao LLM.

A seleção da variante arquitetural apropriada depende fundamentalmente dos requisitos específicos da aplicação, incluindo considerações de latência, qualidade, recursos computacionais disponíveis e complexidade das consultas esperadas.

2.2.4 Métricas de Avaliação para Sistemas RAG

A avaliação abrangente de sistemas RAG requer métricas especializadas que capturem tanto a qualidade do componente de recuperação quanto a precisão do componente de geração. Esta avaliação dual é essencial pois a eficácia global do sistema depende criticamente do desempenho integrado de ambos os componentes. Atualmente, existem frameworks de avaliação como IP e AI (2024) que calculam métricas de recuperação.

Para o componente de recuperação, métricas tradicionais de recuperação de informação são adaptadas e refinadas:

- **Precision@k e Recall@k**: Avaliam respectivamente a precisão e a abrangência dos k documentos mais relevantes recuperados, fundamentais para compreender a qualidade da seleção de contexto.
- **Mean Average Precision (MAP)**: Fornece uma medida agregada da qualidade da classificação dos documentos recuperados ao longo de múltiplas consultas.
- **Normalized Discounted Cumulative Gain (NDCG)**: Considera a posição dos documentos relevantes na lista classificada, penalizando documentos relevantes posicionados inadequadamente.

Para o componente de geração, métricas especializadas avaliam a qualidade, precisão e utilidade do texto gerado:

- **Precisão Factual (*Factual Accuracy*)**: Avalia o grau de correção das informações factuais apresentadas na resposta gerada, tipicamente através de verificação manual ou automática contra fontes confiáveis.
- **Fidelidade (*Faithfulness*)**: Mede o grau em que a resposta gerada permanece consistente com as informações presentes no contexto recuperado, evitando adicionar informações não fundamentadas.
- **Relevância de Resposta (*Answer Relevancy*)**: Avalia quão adequadamente a resposta gerada aborda a consulta original do usuário, independentemente da qualidade do contexto recuperado.
- **Relevância Contextual (*Contextual Relevancy*)**: Mede a adequação do contexto recuperado em relação à consulta específica do usuário.

A avaliação integrada destes componentes frequentemente emprega abordagens híbridas que combinam métricas automáticas com avaliação humana especializada, especi-

almente importante para aplicações educacionais onde aspectos pedagógicos são difíceis de quantificar automaticamente.

2.2.5 Desafios e Limitações dos Sistemas RAG

Apesar de suas vantagens significativas, os sistemas RAG enfrentam desafios técnicos e limitações que devem ser cuidadosamente considerados durante o desenvolvimento e implementação:

- **Qualidade da Recuperação:** A eficácia do sistema depende criticamente da qualidade da recuperação inicial. Recuperação inadequada pode levar a contexto irrelevante ou insuficiente, comprometendo a qualidade da resposta final.
- **Limitações de Contexto:** LLMs possuem janelas de contexto limitadas, restringindo a quantidade de informação recuperada que pode ser efetivamente processada simultaneamente.
- **Latência Computacional:** O processo de recuperação adiciona latência significativa comparado a LLMs puros, podendo impactar a experiência do usuário em aplicações interativas.
- **Qualidade dos *Embeddings*:** A representação vetorial inadequada de documentos ou consultas pode resultar em recuperação imprecisa, especialmente para domínios altamente especializados.
- **Fragmentação de Conhecimento:** A segmentação de documentos longos pode separar informações relacionadas, dificultando a recuperação de contexto completo para consultas complexas.
- **Viés de Recuperação:** Tendências sistemáticas na recuperação podem favorecer certos tipos de conteúdo ou fontes, potencialmente introduzindo vieses nas respostas geradas.
- **Manutenção de Índices:** Sistemas RAG requerem manutenção contínua dos índices de recuperação para garantir atualização e precisão das informações disponíveis.

2.2.6 RAG no Contexto Educacional

A aplicação de sistemas RAG no domínio educacional apresenta características e requisitos únicos que influenciam significativamente o *design* e implementação destes sistemas. A implementação eficaz em contextos educacionais exige consideração cuidadosa de aspectos pedagógicos, epistemológicos e práticos específicos do ambiente de aprendizagem.

A necessidade de precisão factual em aplicações educacionais é particularmente crítica, pois informações incorretas podem ter impacto duradouro no processo de aprendizagem dos estudantes. Sistemas RAG educacionais devem implementar mecanismos robustos de verificação de fatos e citação de fontes para manter a integridade acadêmica e facilitar a verificação independente das informações apresentadas.

A adaptação ao nível de conhecimento do estudante representa outro desafio específico do contexto educacional. Sistemas eficazes devem ser capazes de ajustar a complexidade e profundidade das explicações baseado no nível educacional e conhecimento prévio do usuário, conforme explorado por LIU et al. (2023b).

A capacidade de fornecimento de explicações pedagogicamente estruturadas distingue sistemas RAG educacionais de aplicações genéricas. Isto inclui a habilidade de decompor conceitos complexos, fornecer exemplos apropriados, estabelecer conexões com conhecimento prévio e oferecer caminhos de aprofundamento no material didático.

A integração com materiais didáticos específicos constitui uma vantagem fundamental dos sistemas RAG educacionais, permitindo que estudantes tenham acesso personalizado aos recursos utilizados em seus cursos específicos. Esta integração facilita a manutenção da consistência curricular e permite referenciamento preciso de páginas, capítulos e seções específicas do material de estudo.

O suporte multilíngue assume importância especial em contextos educacionais internacionais ou em países onde materiais didáticos frequentemente transitam entre múltiplos idiomas. A capacidade de processar consultas e fornecer respostas apropriadas em diferentes idiomas, mantendo a precisão das citações e referências, representa um requisito técnico significativo.

2.2.7 Tendências Futuras em Sistemas RAG

O campo de sistemas RAG continua evoluindo rapidamente, com várias tendências emergentes que prometem melhorar significativamente suas capacidades e aplicabilidade:

- **RAG Multimodal:** Integração de capacidades de processamento de texto, imagem, áudio e vídeo para recuperação mais rica e abrangente de informações.
- **RAG com Aprendizagem Contínua:** Sistemas capazes de aprender e adaptar-se continuamente baseado no feedback dos usuários e novas informações.
- **RAG Federado:** Arquiteturas que permitem recuperação de informações distribuídas em múltiplas bases de conhecimento preservando privacidade e controle de acesso.
- **RAG com Raciocínio Avançado:** Integração de capacidades de raciocínio formal e verificação de consistência lógica nas respostas geradas.
- **RAG Personalizado:** Sistemas que se adaptam progressivamente às preferências, nível de conhecimento e estilo de aprendizagem de usuários individuais.

Estas tendências indicam um futuro promissor para sistemas RAG educacionais, com potencial para transformar significativamente a forma como estudantes acessam, processam e integram conhecimento em seus processos de aprendizagem.

2.3 Text *Embeddings*

Os *Text Embeddings* representam representações vetoriais densas de elementos textuais que capturam significado semântico em espaços vetoriais de alta dimensionalidade. O termo *embedding*, derivado do conceito matemático de imersão (*embedding*) de espaços, refere-se ao mapeamento de dados discretos (como palavras ou sentenças) para representações contínuas em espaços vetoriais, preservando relações semânticas e sintáticas importantes. Esta técnica constitui um componente fundamental para sistemas RAG eficientes, permitindo comparações quantitativas de similaridade semântica entre diferentes textos através de operações algébricas no espaço vetorial (REIMERS; GUREVYCH, 2019).

A importância dos *text embeddings* na era moderna do processamento de linguagem natural não pode ser subestimada. Eles representam a interface crítica entre a natureza discreta e simbólica da linguagem humana e os requisitos de processamento contínuo dos algoritmos de aprendizado de máquina. Esta ponte permite que sistemas computacionais processem linguagem natural de forma matematicamente tratável, mantendo simultaneamente as nuances semânticas essenciais para compreensão e geração de linguagem eficazes.

No contexto específico de sistemas RAG educacionais, os *text embeddings* servem como a base tecnológica que permite a recuperação semântica inteligente de informações

relevantes de vastas bases de conhecimento. A qualidade destes *embeddings* determina diretamente a eficácia da recuperação, influenciando consequentemente a precisão e relevância das respostas geradas pelo sistema completo.

2.3.1 Definição Formal e Propriedades Matemáticas

Um *text embedding* pode ser formalmente definido como uma função f que mapeia uma sequência de *tokens* linguísticos $T = [t_1, t_2, \dots, t_n]$ para um vetor $\mathbf{v} \in \mathbb{R}^d$ em um espaço vetorial de dimensão fixa d :

$$f : \mathcal{T} \rightarrow \mathbb{R}^d \quad (2.10)$$

onde \mathcal{T} representa o espaço de todas as possíveis sequências textuais válidas no domínio de interesse. Esta função de mapeamento deve preservar propriedades semânticas importantes, de forma que textos com significados similares sejam mapeados para vetores próximos no espaço *embedding*.

Text embeddings de alta qualidade devem satisfazer propriedades matemáticas e linguísticas específicas que garantem sua utilidade prática em aplicações de processamento de linguagem natural. Estas propriedades, estabelecidas através de décadas de pesquisa iniciada com trabalhos seminais como MIKOLOV et al. (2013), incluem:

- **Preservação de Similaridade Semântica:** Textos com significados relacionados devem possuir *embeddings* com alta similaridade coseno, formalmente expressa como $\text{sim}(\mathbf{v}_i, \mathbf{v}_j) \propto \text{semantic_similarity}(T_i, T_j)$.
- **Invariância a Reformulações:** Sentenças com significados equivalentes mas formulações sintáticas diferentes devem produzir *embeddings* similares, garantindo robustez a variações estilísticas e dialetais.
- **Composicionalidade Semântica:** Para *embeddings* de sentenças ou documentos, o significado do conjunto deve emergir adequadamente da combinação dos componentes individuais.
- **Generalização Entre Domínios:** *Embeddings* treinados em corpora gerais devem manter eficácia em domínios especializados, essencial para aplicações educacionais com terminologia técnica específica.
- **Capacidade Multilíngue:** Para aplicações internacionais, *embeddings* devem capturar relações semânticas equivalentes entre textos em diferentes idiomas, conforme destacado por (FENG et al., 2022) e (ARTETXE; SCHWENK, 2019).

2.3.2 Evolução Histórica dos Modelos de *Text Embeddings*

A evolução dos *text embeddings* reflete o progresso fundamental no campo do processamento de linguagem natural, com cada geração de modelos abordando limitações específicas das abordagens anteriores enquanto introduz novas capacidades e complexidades.

2.3.2.1 *Embeddings* de Palavras Estáticos

Os primeiros modelos de *embedding*, incluindo Word2Vec, GloVe e FastText, focalizaram na criação de representações vetoriais para palavras individuais. Estes modelos implementam o mapeamento fundamental:

$$f_{\text{word}}(w_i) = \mathbf{v}_i \in \mathbb{R}^d \quad (2.11)$$

onde cada palavra w_i no vocabulário é associada a um vetor único \mathbf{v}_i . Estes modelos demonstraram a capacidade de capturar relações semânticas através de operações aritméticas vetoriais, como na famosa relação analógica:

$$\mathbf{V}_{\text{king}} - \mathbf{V}_{\text{man}} + \mathbf{V}_{\text{woman}} \approx \mathbf{V}_{\text{queen}} \quad (2.12)$$

No entanto, uma limitação fundamental destes modelos é a ausência de sensibilidade contextual: cada palavra recebe um único *embedding* independentemente do contexto em que aparece, resultando em representações inadequadas para palavras polissêmicas ou ambíguas.

2.3.2.2 *Embeddings* Contextuais

A introdução de modelos contextuais, exemplificados pelo BERT (DELVIN et al., 2019) e arquiteturas derivadas, representou um avanço qualitativo significativo. Estes modelos geram *embeddings* que dependem explicitamente do contexto linguístico:

$$f_{\text{contextual}}(w_i | w_1, w_2, \dots, w_n) = \mathbf{v}_i^{\text{context}} \in \mathbb{R}^d \quad (2.13)$$

Esta abordagem permite que a mesma palavra tenha representações diferentes em contextos distintos, abordando efetivamente o problema da polissemia e melhorando significativamente a qualidade das representações para tarefas *downstream*.

2.3.2.3 *Embeddings* de Sentenças e Documentos

Modelos especializados como Sentence-BERT (REIMERS; GUREVYCH, 2019) foram desenvolvidos especificamente para gerar *embeddings* de unidades textuais maiores, como sentenças ou parágrafos completos. Estes modelos implementam:

$$f_{\text{sentence}}(S) = \mathbf{v}_S \in \mathbb{R}^d \quad (2.14)$$

onde S representa uma sentença ou documento completo. A otimização destes modelos foca especificamente em tarefas de similaridade semântica, resultando em *embeddings* particularmente eficazes para aplicações de recuperação de informação.

Melhorias subsequentes nas técnicas de *sentence embeddings*, conforme propostas por NI et al. (2022), focalizaram na preservação da semântica em diferentes domínios e na melhoria da robustez a variações estilísticas e estruturais.

2.3.3 Modelos de *Embedding* Modernos

A geração atual de modelos de *embedding* integra múltiplas modalidades de representação para capturar aspectos complementares da semântica textual. O modelo BGE-M3 (*BAAI General Embedding Model, versão M3*), utilizado neste trabalho, exemplifica esta abordagem avançada ao fornecer três tipos distintos de *embeddings* (CHEN et al., 2024):

- ***Embeddings* Densos:** Representações vetoriais tradicionais que capturam similaridade semântica global através de operações de produto interno ou similaridade cosseno.
- ***Embeddings* Esparsos (Lexicais):** Vetores baseados no vocabulário completo do modelo onde cada dimensão representa um token e apenas tokens presentes recebem pesos. Ideais para correspondência lexical exata.

- **Embeddings ColBERT**: Geram um vetor por token do texto (multi-vector) permitindo comparação token-a-token entre query e documento. Capturam interações contextuais granulares com maior precisão

Esta abordagem híbrida permite que sistemas RAG combinem as vantagens de diferentes paradigmas de recuperação, otimizando tanto para similaridade semântica global quanto para correspondências lexicais específicas, conforme apropriado para cada consulta particular.

O modelo BGE-M3 representa avanços significativos em capacidades multilíngues, construindo sobre pesquisas fundamentais em *embeddings* multi-linguais realizadas por FENG et al. (2022) e ARTETXE e SCHWENK (2019). Estas capacidades são particularmente relevantes para o contexto educacional brasileiro, onde materiais didáticos frequentemente transitam entre português e inglês.

2.3.4 Métricas de Similaridade e Operações Vetoriais

A utilidade prática dos *text embeddings* depende fundamentalmente da capacidade de quantificar similaridade semântica através de operações matemáticas no espaço vetorial. A métrica de similaridade mais amplamente utilizada é a similaridade cosseno, definida como:

$$\text{sim}_{\text{cos}}(\mathbf{v}_1, \mathbf{v}_2) = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\|} = \frac{\sum_{i=1}^d v_{1i} v_{2i}}{\sqrt{\sum_{i=1}^d v_{1i}^2} \sqrt{\sum_{i=1}^d v_{2i}^2}} \quad (2.15)$$

onde \mathbf{v}_1 e \mathbf{v}_2 são *embeddings* de dois textos, e v_{1i} , v_{2i} representam seus componentes individuais. Esta métrica produz valores no intervalo $[-1, 1]$, onde valores próximos a 1 indicam alta similaridade semântica, valores próximos a 0 indicam independência semântica, e valores próximos a -1 indicam oposição semântica.

Em sistemas RAG, o processo de recuperação utiliza esta métrica para identificar documentos relevantes. Dada uma consulta do usuário q e um conjunto de documentos candidatos $D = \{d_1, d_2, \dots, d_n\}$, o processo envolve:

1. Computação do *embedding* da consulta: $\mathbf{v}_q = f(q)$
2. Computação dos *embeddings* dos documentos: $\mathbf{v}_{d_i} = f(d_i)$ para $i \in \{1, 2, \dots, n\}$
3. Cálculo da similaridade: $s_i = \text{sim}_{\text{cos}}(\mathbf{v}_q, \mathbf{v}_{d_i})$ para cada documento
4. Seleção dos k documentos com maiores valores de similaridade

Além da similaridade cosseno, outras métricas são utilizadas em contextos específicos:

- **Distância Euclidiana**: $d(\mathbf{v}_1, \mathbf{v}_2) = \|\mathbf{v}_1 - \mathbf{v}_2\|_2$, útil quando a magnitude dos vetores carrega informação semântica relevante.
- **Produto Interno**: $\mathbf{v}_1 \cdot \mathbf{v}_2$, equivalente à similaridade cosseno para vetores normalizados, mas preservando informações de magnitude.
- **Similaridade de Jaccard para Embeddings Esparsos**: Para dois conjuntos finitos A e B , a similaridade de Jaccard é definida como:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (2.16)$$

onde $A \cap B$ representa a interseção dos conjuntos (elementos presentes em ambos) e $A \cup B$ denota a união (todos os elementos únicos). Esta formulação produz valores no intervalo $[0, 1]$, onde $J(A, B) = 1$ indica conjuntos idênticos e $J(A, B) = 0$ indica conjuntos disjuntos.

2.3.5 Técnicas de Busca Aproximada por Vizinhos Mais Próximos

A aplicação prática dos *text embeddings* em sistemas de larga escala requer algoritmos eficientes de busca por vizinhos mais próximos (*Nearest Neighbor Search*). Para bases de conhecimento contendo milhões ou bilhões de documentos, a busca exaustiva torna-se computacionalmente inviável, motivando o desenvolvimento de técnicas de busca aproximada por vizinhos mais próximos (*Approximate Nearest Neighbor* - ANN).

Estas técnicas, fundamentais para a escalabilidade de sistemas RAG, incluem várias abordagens algorítmicas complementares:

- **Locality-Sensitive Hashing (LSH)**: Implementa funções de *hash* que mapeiam vetores similares para os mesmos conjuntos com alta probabilidade, reduzindo drasticamente o espaço de busca. A complexidade de consulta é tipicamente $O(\log n)$ comparada a $O(n)$ para busca exaustiva.
- **Hierarchical Navigable Small World (HNSW)**: Constrói grafos hierárquicos com propriedades de “mundo pequeno” que permitem navegação eficiente através do espaço de *embeddings*. Esta técnica, implementada por Xiong et al. (2021), oferece excelente *trade-off* entre velocidade e precisão.
- **Inverted File Index (IVF)**: Particiona o espaço vetorial em *clusters* e realiza busca apenas nos *clusters* mais promissores, reduzindo significativamente o número de comparações necessárias.
- **Product Quantization (PQ)**: Comprime vetores de alta dimensionalidade através de quantização, reduzindo requisitos de memória e acelerando operações de similaridade sem perda significativa de precisão.
- **Índices Híbridos**: Combinam múltiplas técnicas para otimizar diferentes aspectos do *trade-off* entre velocidade, precisão e uso de memória.

A seleção da técnica ANN apropriada depende dos requisitos específicos da aplicação, incluindo considerações de latência, precisão, uso de memória e padrões de consulta esperados. Para sistemas RAG educacionais, onde tanto a velocidade de resposta quanto a precisão da recuperação são críticas, técnicas híbridas frequentemente oferecem a melhor solução.

2.3.6 *Embeddings* Densos vs. Esparsos

A distinção entre *embeddings* densos e esparsos representa uma dicotomia fundamental na representação vetorial de texto, cada abordagem oferecendo vantagens específicas para diferentes tipos de consultas e aplicações.

Os *embeddings* densos, caracterizados por vetores onde a maioria dos componentes possui valores não-zero, são particularmente eficazes para capturar similaridade semântica global. Estes *embeddings*, tipicamente gerados por modelos neurais como BERT ou BGE-M3, codificam informações semânticas de forma distribuída através de todas as dimensões do vetor. Esta representação permite que conceitos semanticamente relacionados, mesmo quando expressos através de vocabulários diferentes, sejam identificados como similares.

Por outro lado, os *embeddings* esparsos, onde apenas uma pequena fração dos componentes possui valores não-zero, preservam informações sobre correspondências lexicais específicas. Técnicas como TF-IDF ou BM25 geram naturalmente representações esparsas que são altamente eficazes para consultas que requerem correspondência exata de termos específicos, nomes próprios ou terminologia técnica precisa.

O modelo BGE-M3 utilizado neste trabalho integra ambas as abordagens, conforme descrito por KARPUKHIN et al. (2020), permitindo que sistemas RAG otimizem a

recuperação combinando similaridade semântica global (através de *embeddings* densos) com correspondências lexicais precisas (através de *embeddings* esparsos). Esta integração é particularmente valiosa em contextos educacionais onde consultas podem requerer tanto compreensão conceitual quanto identificação de termos técnicos específicos.

2.3.7 Desafios e Limitações dos *Text Embeddings*

Apesar de seus avanços significativos, os *text embeddings* enfrentam desafios importantes que influenciam sua aplicação em sistemas RAG educacionais:

- **Curse of Dimensionality:** Espaços de alta dimensionalidade podem resultar em distâncias uniformes entre pontos, reduzindo a eficácia das métricas de similaridade. Este fenômeno é particularmente problemático para *embeddings* com milhares de dimensões.
- **Viés de Domínio:** *Embeddings* treinados em *corpora* gerais podem não capturar adequadamente nuances terminológicas de domínios especializados, como terminologia médica, jurídica ou técnica específica.
- **Sensibilidade a Variações Superficiais:** Pequenas mudanças na formulação podem resultar em *embeddings* significativamente diferentes, especialmente para modelos sensíveis a ordem de palavras ou estrutura sintática.
- **Limitações de Comprimento:** Muitos modelos possuem limitações na extensão de texto que podem processar eficazmente, necessitando estratégias especiais para documentos longos.
- **Interpretabilidade Limitada:** A natureza distribuída dos *embeddings* densos torna difícil interpretar quais aspectos específicos do significado são capturados por diferentes dimensões.
- **Desempenho Desigual Entre Idiomas:** Mesmo modelos multilíngues podem demonstrar performance variável entre diferentes idiomas, especialmente para idiomas com menos recursos de treinamento disponíveis.

2.3.8 Aplicações Específicas em Sistemas RAG Educacionais

No contexto específico de sistemas RAG educacionais, os *text embeddings* desempenham papéis críticos que vão além da simples recuperação de documentos relevantes. Eles devem facilitar a compreensão pedagógica e suportar diferentes modalidades de aprendizagem.

A capacidade de capturar relações conceituais hierárquicas é fundamental para aplicações educacionais, onde conceitos frequentemente possuem pré-requisitos e dependências complexas. *Embeddings* eficazes devem ser capazes de identificar não apenas similaridade direta, mas também relações de generalização, especialização e dependência conceitual.

A adaptação a diferentes níveis de complexidade representa outro requisito específico do domínio educacional. Idealmente, *embeddings* deveriam capturar não apenas o conteúdo semântico, mas também o nível de sofisticação ou complexidade do texto, permitindo que sistemas RAG adaptem suas respostas ao nível educacional apropriado para cada usuário.

A preservação de precisão terminológica é crítica em contextos educacionais onde definições precisas e uso correto de terminologia técnica são essenciais para o aprendizado eficaz. *Embeddings* educacionais devem balancear flexibilidade semântica com precisão terminológica, evitando confusões entre conceitos relacionados mas distintos.

A capacidade multilíngue assume importância especial em contextos educacionais globais ou em países onde materiais didáticos transitam entre múltiplos idiomas. Para o contexto brasileiro, a capacidade de processar eficazmente tanto português quanto inglês, mantendo equivalência semântica entre traduções de conceitos técnicos, é fundamental para a utilidade prática do sistema.

2.3.9 Tendências Futuras e Desenvolvimentos Emergentes

O campo dos *text embeddings* continua evoluindo rapidamente, com várias tendências emergentes prometendo melhorar significativamente suas capacidades para aplicações educacionais:

- ***Embeddings* Adaptativos:** Modelos que ajustam suas representações baseado no contexto específico da aplicação e *feedback* do usuário, permitindo personalização contínua para diferentes domínios educacionais.
- ***Embeddings* Hierárquicos:** Representações que capturam explicitamente estruturas hierárquicas de conhecimento, essenciais para modelar relações conceituais em materiais educacionais.
- ***Embeddings* Multimodais:** Integração de informações textuais com outros tipos de mídia (imagens, áudio, vídeo) comuns em materiais educacionais modernos.
- ***Embeddings* com Incerteza:** Representações que incluem medidas explícitas de confiança e incerteza, permitindo que sistemas RAG comuniquem adequadamente limitações de conhecimento.
- ***Embeddings* Explicáveis:** Desenvolvimento de técnicas que tornam mais transparente quais aspectos do significado são capturados por diferentes dimensões do *embedding*, facilitando debug e refinamento de sistemas educacionais.

Estas tendências indicam um futuro promissor para a aplicação de *text embeddings* em sistemas educacionais, com potencial para suportar formas mais sofisticadas e eficazes de recuperação e apresentação de conhecimento personalizado para diferentes contextos de aprendizagem.

2.4 Processamento de Documentos PDF

O processamento eficiente de documentos do tipo *Portable Document Format* (PDF) constitui um componente fundamental para sistemas RAG educacionais, pois a maioria dos materiais didáticos acadêmicos são distribuídos neste formato. O PDF foi projetado para preservar a formatação visual e *layout* de documentos independentemente da plataforma de visualização, característica que, embora vantajosa para apresentação, introduz complexidades significativas para extração automatizada de conteúdo estruturado (BAST; KORZEN, 2017).

A importância do processamento adequado de PDFs em contextos educacionais deriva da necessidade de transformar materiais didáticos estáticos em bases de conhecimento estruturadas e pesquisáveis. Esta transformação permite que sistemas RAG acessem e recuperem informações específicas de livros, artigos e outros recursos educacionais, mantendo a fidelidade ao conteúdo original e preservando metadados importantes como estrutura hierárquica, citações e referências.

2.4.1 Desafios no Processamento de PDFs

Os documentos PDF apresentam desafios únicos para extração de informação que distinguem significativamente este processo da manipulação de texto plano ou formatos estruturados como HTML ou XML:

- **Complexidade de *Layout***: Documentos acadêmicos frequentemente utilizam *layouts* sofisticados com múltiplas colunas, elementos flutuantes, cabeçalhos e rodapés, caixas de texto e elementos gráficos integrados que dificultam a identificação da ordem de leitura apropriada.
- **Diversidade de Formatação**: Diferentes editoras, autores e ferramentas de criação resultam em variações significativas nos padrões de formatação, exigindo algoritmos robustos que possam adaptar-se a múltiplos estilos e convenções.
- **Heterogeneidade de Elementos**: PDFs acadêmicos misturam texto corrido, fórmulas matemáticas, tabelas, figuras, gráficos e referências bibliográficas, cada tipo requerendo estratégias de processamento específicas.
- **Preservação de Hierarquia**: A estrutura hierárquica de capítulos, seções e subseções deve ser identificada e preservada para manter a organização conceitual do material didático.
- **Codificação de Caracteres**: Problemas de codificação, caracteres especiais, símbolos matemáticos e elementos tipográficos podem resultar em extração incorreta ou perda de informação crítica.
- **Elementos Não-Textuais**: Informações importantes frequentemente residem em figuras, diagramas, tabelas e outros elementos não-textuais que requerem técnicas especializadas de processamento.

2.4.2 Abordagens para Extração de Conteúdo

Diferentes estratégias têm sido desenvolvidas para abordar os desafios da extração de conteúdo de documentos PDF, cada uma com vantagens e limitações específicas:

2.4.2.1 Extração Baseada em Regras Heurísticas

Abordagens tradicionais utilizam conjuntos de regras programáticas para identificar padrões estruturais como cabeçalhos, parágrafos, listas e citações. Estas técnicas dependem de análise de propriedades visuais como tamanho de fonte, posicionamento, espaçamento e formatação tipográfica. Embora computacionalmente eficientes, estas abordagens sofrem de limitações significativas em robustez e generalização entre diferentes estilos de documento.

2.4.2.2 Extração Baseada em Modelos de *Layout*

Modelos de aprendizado profundo especializados, como LayoutLM (XU et al., 2020) e evoluções como LayoutLMv2 (XU et al., 2021), incorporam informações espaciais e visuais para melhor compreensão da estrutura do documento. Estes modelos são pré-treinados em grandes corpus de documentos e podem identificar regiões de interesse, classificar elementos estruturais e extrair informações com maior precisão que abordagens puramente baseadas em regras.

2.4.2.3 Extração Baseada em Tabela de Conteúdo

Esta abordagem utiliza a tabela de conteúdo (TOC) existente no documento PDF como guia estrutural para a extração. A TOC fornece informações hierárquicas sobre organização de capítulos, seções e subseções, permitindo mapeamento direto do conteúdo para esta estrutura pré-definida. Esta técnica é particularmente eficaz para documentos acadêmicos bem estruturados, como livros didáticos e monografias.

2.4.3 Segmentação de Texto (*Chunking*)

A segmentação ou *chunking* representa um processo crítico na preparação de documentos para sistemas RAG, envolvendo a divisão do texto extraído em segmentos de tamanho apropriado para processamento eficiente e recuperação precisa. A qualidade desta segmentação influencia diretamente tanto a eficácia da recuperação quanto a relevância das respostas geradas (GIDIOTIS; TSOUMAKAS, 2020).

As estratégias de *chunking* variam em sofisticação e adequação para diferentes tipos de conteúdo:

- **Segmentação por Tamanho Fixo:** Divide o texto em blocos de tamanho predeterminado (caracteres ou *tokens*), sendo simples de implementar mas podendo fragmentar unidades semânticas coerentes.
- **Segmentação por Delimitadores Naturais:** Utiliza marcadores linguísticos como parágrafos, sentenças ou seções para determinar fronteiras de segmentos, preservando melhor a coerência semântica.
- **Segmentação Estrutural:** Baseia-se na estrutura hierárquica do documento, criando segmentos que correspondem a unidades conceituais como seções ou subseções.
- **Segmentação com Sobreposição:** Inclui conteúdo compartilhado entre segmentos adjacentes para preservar contexto e evitar perda de informação em fronteiras.
- **Segmentação Semântica:** Utiliza técnicas de processamento de linguagem natural para identificar fronteiras semanticamente significativas, resultando em segmentos mais coerentes tematicamente.

2.4.4 Extração de Metadados Estruturais

A extração de metadados estruturais envolve a identificação e preservação de informações sobre a organização hierárquica do documento, incluindo títulos de capítulos, seções, subseções, numeração, e outras informações estruturais que facilitam navegação e referenciamento precisos.

Estes metadados são fundamentais para sistemas RAG educacionais pois permitem:

- **Contextualização Hierárquica:** Compreensão de onde cada segmento de texto se situa na estrutura global do documento.
- **Referenciamento Preciso:** Capacidade de fornecer citações específicas incluindo livro, capítulo, seção e página.
- **Navegação Estrutural:** Facilitar movimentação entre seções relacionadas ou hierarquicamente conectadas.
- **Filragem Contextual:** Possibilitar recuperação focada em seções específicas baseada em preferências do usuário ou contexto da consulta.

2.4.5 Qualidade e Validação da Extração

A validação da qualidade da extração de PDFs é crucial para garantir a confiabilidade de sistemas RAG educacionais. Aspectos importantes incluem:

- **Fidelidade Textual:** Verificação de que o texto extraído corresponde fielmente ao conteúdo original, sem omissões, duplicações ou corrupções.
- **Preservação Estrutural:** Confirmação de que a hierarquia e organização do documento foram mantidas adequadamente.
- **Integridade de Metadados:** Validação de que informações como títulos, numeração e referências foram extraídas corretamente.
- **Completeness da Extração:** Verificação de que seções importantes não foram omitidas ou mal processadas.
- **Consistência de Formatação:** Garantia de que elementos como fórmulas matemáticas, símbolos especiais e formatação técnica foram preservados apropriadamente.

2.4.6 Limitações e Desafios Técnicos

Apesar dos avanços significativos, o processamento automatizado de documentos PDF ainda enfrenta limitações importantes:

- **Variabilidade de Qualidade:** PDFs gerados por diferentes ferramentas ou digitalizados podem apresentar qualidade inconsistente, afetando a precisão da extração.
- **Elementos Gráficos Complexos:** Diagramas, gráficos e figuras contêm informações importantes que são difíceis de extrair automaticamente.
- **Fórmulas Matemáticas:** Equações e expressões matemáticas frequentemente requerem processamento especializado para preservar significado e formatação.
- **Multilinguismo:** Documentos em múltiplos idiomas ou com caracteres especiais podem apresentar desafios adicionais de codificação e processamento.
- **Escalabilidade:** Processamento de grandes volumes de documentos requer otimizações significativas para manter eficiência computacional.

2.4.7 Tendências e Desenvolvimentos Futuros

O campo de processamento de documentos PDF continua evoluindo, com tendências promissoras incluindo:

- **Modelos Multimodais:** Integração de processamento de texto, imagens e *layout* em modelos unificados que compreendem documentos de forma mais holística.
- **Processamento Semântico:** Utilização de modelos de linguagem avançados para compreender não apenas estrutura, mas também semântica e intenção do conteúdo.
- **Extração Adaptativa:** Sistemas que se adaptam automaticamente a diferentes estilos e formatos de documento sem necessidade de configuração manual.
- **Processamento de Elementos Visuais:** Avanços em visão computacional para extração automática de informações de figuras, gráficos e diagramas.
- **Validação Automática:** Desenvolvimento de métricas e técnicas automáticas para avaliar qualidade e completude da extração sem intervenção humana.

Estes desenvolvimentos prometem tornar o processamento de documentos PDF mais preciso, eficiente e adequado para aplicações educacionais que demandam alta fidelidade e preservação de estrutura conceitual.

2.5 Classificação de Intenções

A classificação de intenções do usuário é fundamental para sistemas conversacionais adaptativos, permitindo personalizar a estratégia de recuperação e geração de acordo com o objetivo da consulta.

2.5.1 Taxonomias de Intenção

As taxonomias de intenção variam de acordo com o domínio de aplicação. Para assistentes educacionais, uma taxonomia típica pode incluir:

- **Resposta a Perguntas (*Question Answering*)**: Busca por informações específicas ou explicações sobre conceitos.
- **Sumarização (*Summarization*)**: Solicitação de resumos concisos de tópicos ou conceitos.
- **Programação (*Coding*)**: Busca por ajuda com implementação ou explicação de código.
- **Busca por Informação (*Searching for Information*)**: Localização de informações específicas em materiais de estudo.
- **Resolução de Problemas (*Problem Solving*)**: Assistência na resolução de exercícios ou problemas complexos.
- **Explicação Conceitual (*Conceptual Explanation*)**: Solicitação de explicações sobre conceitos abstratos ou teóricos.

2.5.2 Técnicas de Classificação de Intenções

As técnicas para classificação de intenções evoluíram significativamente com os avanços na área de Processamento de Linguagem Natural (NLP) (SOLOMIN; IVANOVA, 2020):

2.5.2.1 Abordagens Tradicionais

- **Classificadores Baseados em Regras**: Utilizam padrões e expressões regulares para identificar intenções.
- **Classificadores de Máquina de Vetores de Suporte (SVM)**: Utilizam características extraídas manualmente (TF-IDF, n-gramas, etc.).
- **Classificadores Bayesianos**: Aplicam teorema de Bayes para estimar a probabilidade de cada intenção.

2.5.2.2 Abordagens Baseadas em Redes Neurais

- **Redes Recorrentes (LSTM, GRU)**: Capturam dependências sequenciais em texto.
- **Redes Convolucionais (CNN)**: Identificam padrões locais em sequências de texto.
- **Modelos Baseados em *Transformer***: Utilizam arquiteturas como BERT para classificação de intenções, com resultados estado-da-arte.

2.5.2.3 *Fine-tuning* de Modelos Pré-treinados

A abordagem moderna mais eficaz envolve o *fine-tuning* de modelos de linguagem pré-treinados para a tarefa específica de classificação de intenções como mostrado em

SOLOMIN e IVANOVA (2020):

$$P(i|q) = \text{softmax}(W \cdot f(q) + b) \quad (2.17)$$

onde:

- q é a consulta do usuário
- $f(q)$ é o *embedding* da consulta, gerado pelo modelo pré-treinado
- W e b são os parâmetros da camada de classificação
- $P(i|q)$ é a distribuição de probabilidade sobre as possíveis intenções

2.5.3 Desafios na Classificação de Intenções

A classificação de intenções enfrenta diversos desafios:

- **Ambiguidade:** Uma mesma consulta pode ter múltiplas interpretações.
- **Intenções Sobrepostas:** Uma consulta pode conter múltiplas intenções simultâneas.
- **Variações Linguísticas:** Diferentes formulações da mesma intenção.
- **Dependência de Contexto:** A intenção pode depender do histórico da conversa.
- **Multilinguismo:** Necessidade de entender intenções em diferentes idiomas.

Sistemas modernos de classificação de intenções abordam estes desafios através de modelos mais sofisticados, conjuntos de dados diversos e técnicas de aumento de dados.

2.6 Assistentes Educacionais Baseados em IA

Os assistentes educacionais baseados em IA representam uma aplicação especializada de sistemas conversacionais, com foco em suporte ao aprendizado e desenvolvimento de conhecimento.

2.6.1 Fundamentos Pedagógicos

Os assistentes educacionais eficazes são fundamentados em princípios pedagógicos estabelecidos em WOLLNY et al. (2021):

- **Aprendizado Personalizado:** Adaptação às necessidades e estilo de aprendizado individuais.
- **Scaffolding:** Fornecimento de suporte gradual, ajustado ao nível de compreensão do estudante.
- **Feedback Imediato:** Correção e orientação em tempo real.
- **Aprendizado Ativo:** Promoção de engajamento e reflexão crítica.
- **Zona de Desenvolvimento Proximal:** Desafios que estendem o conhecimento atual, mas permanecem acessíveis.

2.6.2 Tipos de Assistentes Educacionais

Diferentes tipos de assistentes educacionais atendem a objetivos pedagógicos específicos:

- **Tutores Inteligentes:** Sistemas que guiam o estudante através de um currículo estruturado, adaptando-se ao seu progresso e dificuldades.
- **Assistentes de Pesquisa:** Focados em auxiliar estudantes a encontrar e compreender informações relevantes para suas pesquisas.
- **Companheiros de Estudo:** Simulam um parceiro de estudo, promovendo discussões e debates sobre o material de aprendizado.

- **Auxiliares de Resolução de Problemas:** Especializados em ajudar estudantes a resolver problemas e exercícios, fornecendo dicas progressivas.
- **Facilitadores de Reflexão:** Estimulam o pensamento metacognitivo, ajudando estudantes a refletir sobre seu próprio aprendizado.

2.6.3 Impacto dos LLMs na Educação

A integração de LLMs avançados em assistentes educacionais tem transformado significativamente o cenário educacional (JUMAR et al., 2024) (HAN et al., 2024):

- **Acessibilidade Ampliada:** Democratização do acesso a tutoria personalizada.
- **Suporte Contínuo:** Disponibilidade 24/7 para esclarecer dúvidas e fornecer orientações.
- **Personalização em Escala:** Adaptação a diferentes estilos de aprendizado e níveis de conhecimento.
- **Redução de Barreiras Psicológicas:** Estudantes podem fazer perguntas sem receio de julgamento.
- **Desenvolvimento de Habilidades Metacognitivas:** Promoção da reflexão sobre o próprio processo de aprendizado.

2.6.4 Desafios na Implementação de Assistentes Educacionais

Apesar dos avanços, a implementação de assistentes educacionais baseados em IA enfrenta desafios significativos:

- **Precisão e Confiabilidade:** Necessidade de minimizar informações incorretas ou enganosas.
- **Adaptação Pedagógica:** Alinhamento com objetivos educacionais e metodologias pedagógicas eficazes.
- **Avaliação de Eficácia:** Dificuldade em medir o impacto real no aprendizado.
- **Dependência Excessiva:** Risco de estudantes desenvolverem dependência do assistente em detrimento de habilidades autônomas.
- **Equidade e Acesso:** Garantia de que todos os estudantes possam se beneficiar igualmente da tecnologia.
- **Barreiras Linguísticas:** Desempenho desigual em diferentes idiomas, como destacado por (SHEN et al., 2024).

2.7 Engenharia de *Prompt*

A engenharia de *prompts* é um componente crítico para sistemas RAG, determinando como o LLM utiliza o contexto recuperado para gerar respostas relevantes e úteis.

2.7.1 Fundamentos da Engenharia de *Prompts*

A engenharia de *prompts* para LLMs envolve a estruturação cuidadosa de instruções e contexto para obter resultados desejados como mostrado em 2.4. Para sistemas RAG, como mostrado em LIU et al. (2023a), isso inclui:

- **Instruções Claras:** Orientações explícitas sobre como interpretar e utilizar o contexto recuperado.
- **Formatação Estruturada:** Apresentação do contexto de forma a facilitar sua compreensão pelo modelo.

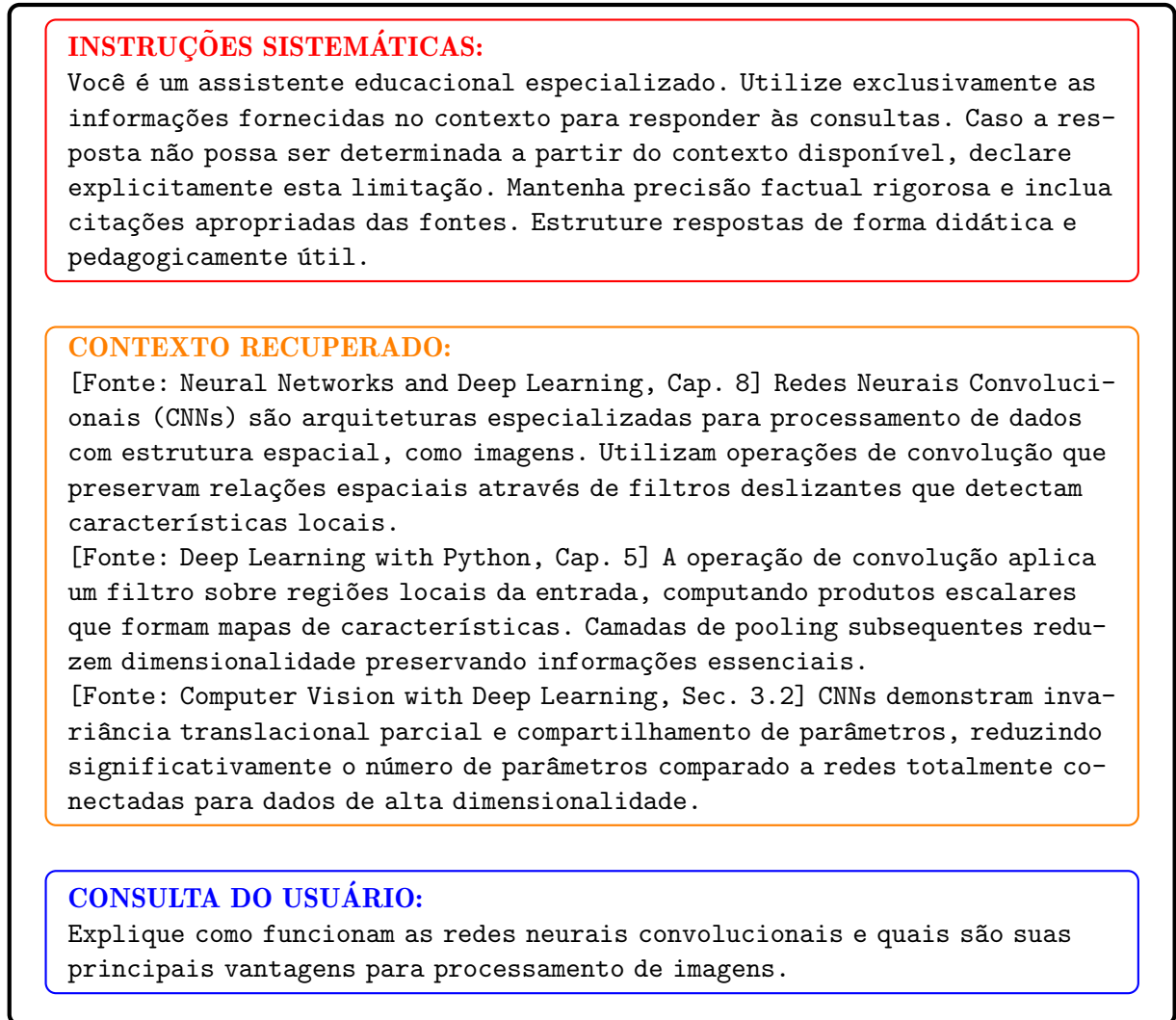


Figura 2.4 – Estrutura de um *prompt* RAG educacional, demonstrando a integração de instruções sistemáticas, contexto recuperado estruturado e consulta do usuário para otimização da qualidade da resposta.

- **Especificação de Saída:** Definição do formato e conteúdo esperados na resposta.
- **Controle de Comportamento:** Instruções para garantir precisão, citação adequada e reconhecimento de limitações.

2.7.2 Técnicas Avançadas de Engenharia de *Prompts*

Diversas técnicas têm sido desenvolvidas para melhorar a eficácia da engenharia de *prompts*:

2.7.2.1 *Prompts* em Poucas Etapas (*Few-Shot Prompting*)

Esta técnica envolve a inclusão de exemplos do comportamento desejado no *prompt*:

Você é um assistente educacional especializado em matemática.
 Dada uma pergunta, forneça uma explicação clara e passo a passo.

Exemplo 1:

Pergunta: Como resolver $x^2 - 5x + 6 = 0$?

Resposta: Para resolver esta equação quadrática, vamos fatorar o lado esquerdo. Notamos que $x^2 - 5x + 6 = (x-2)(x-3)$. Igualando cada fator a zero, obtemos $x=2$ ou $x=3$. Portanto, as soluções são $x=2$ e $x=3$.

Agora, responda à seguinte pergunta:

{consulta_do_usuario}

2.7.2.2 Prompts em Cadeia de Pensamento (*Chain-of-Thought Prompting*)

Esta técnica incentiva o modelo a decompor problemas complexos em passos intermediários, como mostrado em WEI et al. (2022):

Para responder à pergunta do usuário, siga estes passos:

1. Identifique os conceitos-chave mencionados na pergunta
2. Consulte as informações relevantes no contexto fornecido
3. Sintetize uma explicação baseada nessas informações
4. Verifique se a explicação está completa e precisa
5. Forneça uma resposta final com as citações apropriadas

2.7.2.3 Prompts Adaptados à Intenção

Prompts específicos para diferentes intenções do usuário podem melhorar significativamente a qualidade das respostas:

- **Para Resposta a Perguntas:** Enfatiza precisão, abrangência e citação adequada.
- **Para Sumarização:** Foca em concisão, identificação de pontos-chave e estrutura clara.
- **Para Programação:** Prioriza clareza, correção e explicação do código.
- **Para Busca de Informações:** Destaca a organização e a relevância das referências.

2.7.3 Otimização de Consultas para Recuperação Eficaz

Uma técnica avançada em sistemas RAG é a otimização de consultas, onde a consulta original do usuário é transformada em consultas de recuperação mais eficazes:

- **Expansão de Consultas:** Adição de termos relacionados para aumentar a cobertura da recuperação.
- **Reformulação:** Transformação da consulta em formato mais adequado para recuperação.
- **Decomposição:** Divisão de consultas complexas em subconsultas mais específicas.
- **Especificação:** Adição de termos técnicos ou acadêmicos para melhorar a precisão.

Esta otimização pode ser realizada utilizando o próprio LLM como um agente de reformulação, seguindo um *prompt* como:

Sua tarefa é transformar a consulta do usuário em consultas de recuperação eficazes para um sistema RAG educacional.

Consulta original: "Como funciona aprendizado por reforço?"

Gere:

1. Uma versão aprimorada da consulta original

2. Três consultas de recuperação focadas, usando terminologia técnica

Formato de saída:

```
<enhanced>versão aprimorada da consulta</enhanced>
<retrieval1>consulta 1</retrieval1>
<retrieval2>consulta 2</retrieval2>
<retrieval3>consulta 3</retrieval3>
```

2.8 Considerações Éticas e Limitações

O desenvolvimento e implementação de assistentes educacionais baseados em LLMs e RAG levantam importantes considerações éticas e apresentam limitações significativas.

2.8.1 Considerações Éticas

- **Vieses e Preconceitos:** LLMs podem refletir e amplificar vieses presentes nos dados de treinamento, potencialmente perpetuando estereótipos e desigualdades (SHEN et al., 2024).
- **Privacidade:** Consultas dos estudantes podem revelar informações sensíveis sobre seu conhecimento, dificuldades e interesses.
- **Autonomia do Aprendizado:** Dependência excessiva do assistente pode comprometer o desenvolvimento de habilidades críticas de pensamento independente.
- **Transparência:** Estudantes devem compreender quando estão interagindo com um sistema automatizado e suas limitações.
- **Acessibilidade:** Garantia de que a tecnologia seja acessível a estudantes de diferentes contextos socioeconômicos e habilidades.

2.8.2 Limitações Técnicas

- **Alucinações:** LLMs podem gerar informações incorretas com alta confiança, especialmente quando o contexto recuperado é insuficiente.
- **Raciocínio Matemático:** Dificuldade em resolver problemas matemáticos complexos que exigem raciocínio multietapa.
- **Compreensão Profunda:** Limitações na compreensão genuína de conceitos abstratos e suas interrelações.
- **Adaptabilidade:** Desafios em ajustar dinamicamente o nível de explicação ao conhecimento prévio do estudante.
- **Multilinguismo:** Desempenho desigual em diferentes idiomas, com menor qualidade em línguas com menos recursos.

2.8.3 Mitigação e Boas Práticas

Diversas estratégias podem ser adotadas para mitigar estas limitações:

- **Controle de Precisão:** Implementação de sistemas RAG robustos para fundamentar as respostas em fontes confiáveis.
- **Transparência de Limitações:** Comunicação clara das limitações do sistema aos estudantes.
- **Revisão Humana:** Supervisão por educadores para conteúdos sensíveis ou complexos.
- **Design Centrado no Estudante:** Desenvolvimento focado nas necessidades reais e experiência dos estudantes.

- **Avaliação Contínua:** Monitoramento e aprimoramento constante para detectar e corrigir problemas.

2.9 Conclusão

Este capítulo apresentou os fundamentos teóricos essenciais para a compreensão do Assistente de Estudos baseado em IA. Foram abordados os princípios e técnicas que sustentam sistemas modernos baseados em LLMs e RAG, desde a arquitetura *Transformer* até métodos avançados de engenharia de *prompts* e avaliação.

A compreensão destes fundamentos é crucial para o desenvolvimento de sistemas educacionais que sejam não apenas tecnicamente robustos, mas também pedagogicamente eficazes. Os próximos capítulos detalharão a metodologia específica e a implementação do sistema proposto, construindo sobre estas bases teóricas.

Os avanços recentes em LLMs, técnicas de RAG, processamento de documentos e classificação de intenções abrem possibilidades sem precedentes para assistentes educacionais personalizados e escaláveis (LIAO et al., 2024). No entanto, é fundamental abordar estes desenvolvimentos com uma compreensão clara tanto de suas capacidades quanto de suas limitações, garantindo que a tecnologia sirva efetivamente aos objetivos educacionais e ao desenvolvimento do estudante.

3 IMPLEMENTAÇÃO

Este capítulo apresenta a metodologia utilizada para o desenvolvimento do assistente educacional baseado em Inteligência Artificial, detalhando as técnicas e procedimentos empregados em cada etapa do processo. O sistema proposto integra múltiplas tecnologias, incluindo processamento de documentos PDF, geração de *embeddings*, classificação de intenções e RAG, para criar uma solução adaptativa às necessidades específicas dos estudantes.

3.1 Processamento de Documentos PDF

O processamento eficiente e estruturado de documentos PDF constitui a base fundamental para o sistema RAG desenvolvido neste trabalho. Esta etapa é crucial pois determina a qualidade e a estrutura das informações disponíveis para recuperação durante as interações com o usuário. Para atingir resultados otimizados, foi desenvolvido um *pipeline* de processamento que aborda os diversos desafios inerentes à extração de informações de documentos acadêmicos e didáticos.

3.1.1 Estrutura Geral do *Pipeline* de Processamento

O *pipeline* de processamento de documentos PDF foi projetado para maximizar tanto a preservação da estrutura original do documento quanto a qualidade da recuperação subsequente. A abordagem implementada segue as seguintes etapas principais:

1. Extração e análise da tabela de conteúdo (TOC) do documento
2. Identificação automática de capítulos e seções relevantes
3. Extração de texto com preservação do contexto estrutural
4. Segmentação do texto em *chunks* semanticamente coerentes
5. Filtragem e limpeza dos *chunks* para remoção de conteúdo irrelevante
6. Associação de metadados estruturais a cada *chunk*

Esta abordagem foi inspirada em metodologias estabelecidas na literatura para processamento de documentos acadêmicos, particularmente nas técnicas de extração estruturada propostas por YU, ZHANG e WANG (2020), mas com adaptações específicas para o contexto educacional.

3.1.2 Extração e Análise da Tabela de Conteúdo

A primeira etapa do processamento consiste na extração e análise da tabela de conteúdo (TOC) do documento PDF. Esta etapa é fundamental pois a TOC fornece informações valiosas sobre a estrutura hierárquica do documento, facilitando a navegação semântica pelo conteúdo. Para realizar esta extração, utiliza-se a biblioteca PyMuPDF (ARTIFEX SOFTWARE, Inc., 2025), que permite acessar diretamente a estrutura interna do documento:

```
def get_table_of_contents_from_PDF(path: str):
    if not os.path.exists(path):
        raise FileNotFoundError(f"O arquivo {path} não existe.")
    doc = fitz.open(path)
    toc = doc.get_toc()
```

```
doc.close()
return toc
```

Tabela 3.1 – Extrato do sumário do livro “Hands-On Machine Learning” (GÉRON, 2019)

Nível	Título	Página
1	Cover	1
1	Copyright	4
1	Table of Contents	5
1	Preface	13
2	The Machine Learning Tsunami	13
2	Machine Learning in Your Projects	13
2	Objective and Approach	14
2	Prerequisites	15
2	Roadmap	15
2	Other Resources	17
2	Conventions Used in This Book	18
2	Code Examples	19
2	Using Code Examples	20
2	O’Reilly Safari	20
2	How to Contact Us	20
2	Changes in the Second Edition	21
2	Acknowledgments	25
1	Part I. The Fundamentals of Machine Learning	27
2	Chapter 1. The Machine Learning Landscape	29
3	What Is Machine Learning?	30
3	Why Use Machine Learning?	30

Nota: Esta tabela representa apenas uma parte do sumário original do livro “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow” (2ª edição) de Aurélien Géron, extraída e organizada pelo autor para fins de demonstração.

O TOC extraído, como mostrado na Tabela 3.1, é uma lista de tuplas contendo três elementos: nível hierárquico, título da seção e número da página. Esta estrutura é então processada para identificar os elementos estruturais relevantes do documento.

3.1.3 Identificação Automatizada de Capítulos

A identificação de capítulos e seções relevantes é realizada através de uma abordagem híbrida que combina análise heurística e processamento de linguagem natural. Para superar as limitações das abordagens puramente baseadas em regras, foi implementada uma técnica de classificação assistida por um modelo de linguagem (gpt-4o-mini):

```
def get_model_answer_of_chapters(toc_prompt: str, model: str = None):
    # Instruções para o modelo identificar capítulos e apêndices

    prompt_engineering = """Given the table of contents of this book
    ↳ containing the importance index, topic name and topic page, provide me
    ↳ with the list of chapters and apendixes in the book.
    Do not include figures, tables, preface, index, bibliography, or any other
    ↳ non-chapter or non-appendix sections.
```

```

If an appendix is present with sub-sections, the sub-sections should not
↳ be included in the list, only the appendix name. For example, if the
↳ appendix is "Appendix A" and it has sub-sections "A.1", "A.2", "A.3",
↳ only "Appendix A" should be included in the list.
If the book is separated into parts, the parts should not be included in
↳ the list.
This list must be contain only the Name of the sections.
The Names must be exactly as they appear in the table of contents.
The response must have only the list in python format. For example, if the
↳ list is ['a', 'b', 'c'], the response must be ['a', 'b', 'c']. It cannot
↳ have any other text. If the list is empty, the response must be []."""

# Combinação do TOC e instruções
full_prompt = toc_prompt + prompt_engineering

# Chamada ao modelo de linguagem
response = generate_inference(
    [{"role": "user", "content": full_prompt}],
)

```

Esta abordagem permite identificar de forma robusta os capítulos relevantes mesmo em documentos com estruturas não padronizadas, superando as limitações dos métodos tradicionais baseados exclusivamente em regras heurísticas.

3.1.4 Extração de Texto com Preservação de Contexto

Após a identificação dos elementos estruturais relevantes, procede-se à extração do conteúdo textual. Esta etapa foi implementada de forma a preservar o contexto estrutural, distinguindo entre diferentes níveis hierárquicos do documento (introduções de capítulos, seções, subseções, etc.).

```

Exemplo simplificado para extração de introdução de capítulo
if topics: # Se o capítulo tem tópicos
    pre_topic_text = ""
    for i in range(chapter_page, topics[0]["Page"]):
        page_text = reader.pages[i].extract_text()
        title_test_text = topics[0]["Topic"] + "\n"
        if title_test_text in page_text:
            page_text = page_text.split(title_test_text)[0]
        pre_topic_text += page_text
else: # Se o capítulo não tem tópicos identificados
    # Determinar onde o capítulo termina
    if index == len(summary_list) - 1: # Último capítulo
        next_chapter_page = len(reader.pages)
    else:
        next_chapter_page = summary_list[index + 1]["Page"]
    # Extrair todo o conteúdo do capítulo
    pre_topic_text = ""
    for i in range(chapter_page, next_chapter_page):
        pre_topic_text += reader.pages[i].extract_text()

```

Este processo é repetido para cada elemento estrutural identificado, resultando em uma extração que preserva a relação hierárquica entre os diferentes componentes do documento.

3.1.5 Segmentação em *Chunks* Semanticamente Coerentes

Um dos aspectos mais críticos para o desempenho de sistemas RAG é a estratégia de segmentação (*chunking*) do texto extraído. Neste trabalho, adotou-se uma abordagem

híbrida de *chunking* que combina divisão por tamanho fixo com considerações semânticas e estruturais.

```

text_splitter = NLTKTextSplitter(
    chunk_size=chunk_size,
    separator="\n",
    chunk_overlap=1000
)
for text in text_splitter.split_text(pre_topic_text):
    text = text.encode("utf-8", errors="ignore").decode("utf-8")
    # Filtros de qualidade aplicados ao chunk
    if len(text) < 300: # Skip chunks muito pequenos
        continue
    # Filtro para textos com excesso de pontuação
    normalized_text = re.sub(r"\.{2,}", "", text)
    filtered_text = re.sub(
        r"\d\.\d",
        lambda m: m.group().replace(".", ""),
        normalized_text
    )
    if filtered_text.count(".") / len(text) > 0.02:
        continue

    # Armazenamento do chunk processado
    full_df = full_df._append(
        {
            "Book": book_name,
            "Chapter": title,
            "Text": text,
            "Topic": "Chapter Introduction",
            "is_introduction": True,
        },
        ignore_index=True,
    )

```

A estratégia de divisão implementada utiliza a biblioteca NLTK para realizar cortes que respeitam fronteiras linguísticas naturais, enquanto também mantém uma sobreposição significativa (1000 caracteres no exemplo) entre *chunks* consecutivos para preservar contexto. Adicionalmente, foram implementados filtros de qualidade que removem *chunks* muito pequenos ou com características que indicam baixa qualidade textual (como excesso de pontuação).

3.1.6 Metadados e Enriquecimento de *Chunks*

Cada *chunk* gerado no processo anterior é enriquecido com metadados estruturais que facilitam sua posterior recuperação contextual. Estes metadados incluem:

- Nome do livro ou documento fonte
- Título do capítulo
- Tópico ou seção específica
- *Flag* indicando se o *chunk* pertence a uma introdução
- Informações de posicionamento relativo no documento

Este enriquecimento com metadados é crucial para a fase de recuperação, pois permite que o sistema não apenas recupere *chunks* relevantes, mas também compreenda seu contexto estrutural dentro do documento original.

3.1.7 Desafios e Soluções Implementadas

O processamento de documentos PDF apresenta diversos desafios que foram abordados com soluções específicas:

3.1.7.1 Desafio de *Layout* Variável

Documentos acadêmicos e didáticos frequentemente apresentam *layouts* complexos com múltiplas colunas, figuras incorporadas, tabelas e elementos flutuantes. Para abordar este desafio, foi adotada uma estratégia de processamento baseada na tabela de conteúdo (TOC) em vez de técnicas de análise de *layout* visual. Esta abordagem permite contornar as complexidades de *layout*, focando na estrutura semântica do documento conforme representada na TOC.

3.1.7.2 Desafio de Fronteiras Semânticas

A divisão de texto em *chunks* pode frequentemente romper unidades semânticas coerentes, como explicações de conceitos ou passos de um raciocínio. Para mitigar este problema, foi implementada uma sobreposição significativa entre *chunks* consecutivos (*chunk_overlap* = 1000) com um tamanho de *chunk* de 3000 caracteres (*chunk_size* = 3000). Esses valores foram obtidos experimentalmente através da análise de uma amostra aleatória de *chunks*, identificando um tamanho que comportasse um parágrafo médio, já que a segmentação é baseada nessa estrutura. Isso garante que conceitos que possam ter sido divididos em dois *chunks* ainda possam ser recuperados integralmente. Adicionalmente, a utilização do NLTKTextSplitter com separador baseado em quebras de linha ajuda a respeitar fronteiras naturais de parágrafos.

3.1.8 Processamento em Lote e Gerenciamento de Memória

Para permitir o processamento eficiente de grandes volumes de documentos, foi implementada uma abordagem de processamento em lote com gerenciamento cuidadoso de memória:

```
def process_and_save_multiple_pdfs(
    self,
    pdf_paths: list,
    book_names: list = None,
    chunk_size: int = 3000,
    output_path: str = "embeddings.db",
):
    # Ensure output_path has .db extension
    db_path = (
        output_path.replace(".json", ".db").replace(".pkl", ".db")
        if output_path.endswith((".json", ".pkl"))
        else output_path
    )
    # Process each PDF file separately
    for i, path in tqdm(
        enumerate(pdf_paths), desc="Processing PDF files", total=len(pdf_paths)
    ):
        book_name = book_names[i] if book_names and i < len(book_names) else None
        print(f"\nProcessing PDF: {book_name or path}")

        try:
            self.process_and_save_pdf_by_chapter(
```

```

        path, db_path, book_name, chunk_size
    )
    print(f"Completed processing {book_name or path}")

    # Force garbage collection between files
    import gc
    gc.collect()

except Exception as e:
    print(f"Error processing {book_name or path}: {e}")
    continue

```

Esta abordagem processa cada documento individualmente, salvando os resultados diretamente em um banco de dados SQLite em vez de manter todos os dados em memória. Adicionalmente, o código explicitamente força a coleta de lixo (*garbage collection*) entre documentos para evitar vazamentos de memória durante o processamento de grandes conjuntos de PDFs.

3.2 Classificação de Intenções

A classificação precisa das intenções do usuário representa um componente fundamental do sistema educacional desenvolvido, permitindo personalizar a estratégia de recuperação e geração de respostas de acordo com objetivos específicos de aprendizagem. Esta seção detalha a metodologia implementada para o desenvolvimento, treinamento e integração do classificador de intenções, destacando as escolhas técnicas e suas justificativas.

3.2.1 Taxonomia de Intenções no Contexto Educacional

O primeiro passo para o desenvolvimento do classificador foi estabelecer uma taxonomia adequada de intenções relevantes para o contexto educacional. Após análise da literatura e observação de interações típicas em ambientes de aprendizagem, foram definidas as seguintes categorias de intenção:

- **Resposta a Perguntas (*Question Answering*)**: Consultas que buscam informações factuais específicas ou explicações diretas sobre conceitos.
- **Sumarização (*Summarization*)**: Solicitações para condensar ou resumir um tópico ou conceito complexo.
- **Programação (*Coding*)**: Consultas relacionadas à implementação ou explicação de código.
- **Busca por Informação (*Searching for Information*)**: Consultas exploratórias que buscam localizar informações específicas em documentos.

Esta taxonomia foi desenvolvida considerando os cenários mais comuns de interação entre estudantes e materiais didáticos.

3.2.2 Conjunto de Dados para Treinamento

Para treinar um classificador robusto e preciso, foi necessário desenvolver um conjunto de dados representativo que abrangesse as diversas categorias de intenção e contemplasse a variabilidade linguística natural. O processo de construção deste conjunto envolveu as seguintes etapas:

3.2.2.1 Construção de exemplos iniciais

Foi criado com base nas necessidades do projeto um *dataset* que tenta representar o máximo de possibilidades que o projeto possa demandar.

3.2.2.2 Aumento de Dados via LLMs

A estratégia adotada neste trabalho foi a utilização de LLMs para o aumento de dados. Partindo do conjunto inicial de exemplos, utilizou-se LLMs para gerar variações e exemplos adicionais, seguindo técnicas similares às propostas por LI et al. (2023b). Este processo envolveu:

- Fornecimento de exemplos base aos LLMs como poucas amostras (*few-shot*)
- Instruções explícitas para geração de variações preservando a intenção original
- Diversificação de formulações, níveis de formalidade e domínios de conhecimento
- Revisão manual de amostras para garantir qualidade e adequação

Esta abordagem permitiu a expansão significativa do conjunto de dados de treinamento, garantindo maior diversidade e robustez ao classificador treinado.

3.2.2.3 Balanceamento do Conjunto de Dados

Para garantir que o modelo não desenvolvesse viés em favor das categorias mais representadas, foi realizado um balanceamento cuidadoso do conjunto de dados, assegurando que cada categoria de intenção tivesse representação aproximadamente equivalente no conjunto de treinamento.

3.2.2.4 Validação Multilíngue

Considerando a natureza multilíngue do sistema desenvolvido, o conjunto de dados foi expandido para incluir exemplos tanto em português quanto em inglês. Este passo foi crucial para garantir o desempenho adequado do classificador em ambos os idiomas. O processo de geração via LLMs foi conduzido separadamente para cada idioma, assegurando que as nuances linguísticas específicas fossem adequadamente representadas. O conjunto de dados final está disponível publicamente no *Huggingface* no seguinte URL:

<https://huggingface.co/datasets/claudiomello/AcademiCK-intent-data>

3.2.3 Arquitetura do Modelo de Classificação

Após experimentação com diversas arquiteturas, optou-se por utilizar uma abordagem baseada em *fine-tuning* do modelo Snowflake/snowflake-arctic-embed-l-v2.0 (PUXUAN et al., 2024). Esta escolha foi motivada por diversos fatores:

- Desempenho superior em tarefas de compreensão de linguagem natural
- Capacidades multilíngues robustas, particularmente em português e inglês
- Eficiência computacional, permitindo inferência rápida no contexto de um sistema interativo
- Compatibilidade com a infraestrutura de *embeddings* utilizada no componente RAG

3.2.4 Processo de Treinamento via *Huggingface AutoTrain*

Foi utilizada a plataforma *Huggingface AutoTrain* para o treinamento do classificador de intenções. Esta abordagem ofereceu diversas vantagens:

- Simplificação do processo de *fine-tuning*
- Paralelização eficiente do treinamento em infraestrutura otimizada
- Integração direta com o ecossistema *Huggingface* para armazenamento e distribuição do modelo
- Reprodutibilidade e documentação transparente do processo de treinamento

O processo de treinamento seguiu o fluxo padrão do *AutoTrain*, incluindo:

1. Carregamento do conjunto de dados previamente preparado
2. Divisão automática em conjuntos de treinamento e validação (85%/15%)
3. Seleção do modelo base (Snowflake/snowflake-arctic-embed-l-v2.0)
4. Configuração da tarefa como classificação de texto
5. Execução do treinamento

Os hiperparâmetros utilizados no treinamento foram:

```
{
  "data_path": "Classifier-Intent-snowflake/autotrain-data",
  "model": "Snowflake/snowflake-arctic-embed-l-v2.0",
  "lr": 5e-05,
  "epochs": 20,
  "max_seq_length": 8192,
  "batch_size": 8,
  "warmup_ratio": 0.1,
  "gradient_accumulation": 1,
  "optimizer": "adamw_torch",
  "scheduler": "linear",
  "weight_decay": 0.0,
  "max_grad_norm": 1.0,
  "seed": 42,
  "train_split": "train",
  "valid_split": "validation",
  "text_column": "autotrain_text",
  "target_column": "autotrain_label",
  "logging_steps": 1,
  "project_name": "AcademiCK-intent-classifier",
  "auto_find_batch_size": true,
  "mixed_precision": "fp16",
  "save_total_limit": 1,
  "push_to_hub": true,
  "eval_strategy": "epoch",
  "username": "claudiomello",
  "log": "tensorboard",
  "early_stopping_patience": 5,
  "early_stopping_threshold": 0.01
}
```

O modelo treinado está disponível publicamente no *Huggingface* e pode ser acessado no seguinte URL:

<https://huggingface.co/claudiomello/AcademiCK-intent-classifier>

3.2.5 Implementação do Classificador no Sistema

A implementação do classificador no sistema final foi realizada através da integração com o *ResourceManager*, que gerencia os recursos compartilhados do sistema, incluindo modelos carregados. O código a seguir demonstra como o classificador é inicializado e utilizado:

```

def initialize_resources(self, embeddings_model: str = "BAAI/bge-m3"):
    """Initialize all shared resources including models and embeddings"""
    if self.embeddings_model is None:
        print("Loading BGE-M3 embeddings model...")
        self.embeddings_model = BGEM3FlagModel(
            embeddings_model,
            device=self.device,
            query_max_length=8192,
            use_fp16=True,
        )
    if self.prompt_classifier is None:
        print("Loading prompt classifier...")
        self.prompt_classifier = pipeline(
            "text-classification",
            model="claudiomello/Classifier-Intent-snowflake",
            device=self.device,
        )

    # Load embeddings automatically
    if self.embedding_df is None:
        self._load_embeddings()

```

A função de classificação é então utilizada de forma simples, retornando a intenção prevista para uma determinada consulta:

```

def get_intent(self, query: str):
    return self.resource_manager.prompt_classifier(query)[0]["label"]

```

3.2.6 Integração com o Sistema RAG

A integração do classificador de intenções com o sistema RAG constitui um dos aspectos mais inovadores da abordagem implementada. Esta integração permite adaptar todo o *pipeline* de recuperação e geração com base na intenção identificada:

```

def generate_response_conversation(self, query: str, model: str | None =
    ↪ None):
    """
    Generates a response to the query using the language model, RAG context, and
    ↪ conversation history.
    Args:
        query (str): The query to generate a response for.
        model (str): The language model to use for generating the response.
    Returns:
        str: The response generated by the language model.
    """
    # If the model is not provided, use the default model
    if model is None:
        model = self.model
    # Get the intent of the query
    intention = self.get_intent(query)

    # If the intent is searching for information, use a different top_k value to
    ↪ generate the prompt
    if intention == "searching_for_information":
        messages = self.generate_LLM_prompt(
            query, intention, conversation=True, top_k=12
        )
    else:
        messages = self.generate_LLM_prompt(

```

```

        query, intention, conversation=True, top_k=6
    )

    # Store only the user query in message history (system message with context is
    ↪ not stored)
    self.messages.append({"role": "user", "content": query})

    # Use unified inference function
    response_content = generate_inference(messages, model)

    # Add the assistant response to the messages
    self.messages.append({"role": "assistant", "content": response_content})

    return response_content

```

Como demonstrado no código acima, a intenção identificada influencia diretamente parâmetros críticos do sistema RAG. Especificamente, a quantidade de documentos recuperados (*top_k*) é ajustada conforme a natureza da consulta: consultas de busca por informação (*searching_for_information*) utilizam (*top_k* = 12) para garantir cobertura abrangente, enquanto outras intenções empregam (*top_k* = 6), priorizando precisão sobre quantidade.

Além do ajuste na recuperação de documentos, como mostrado no código a seguir, cada intenção identificada ativa estratégias específicas de engenharia de *prompts*, otimizadas para o tipo particular de consulta. Essas estratégias personalizadas determinam tanto a formulação das instruções para o LLM quanto os formatos de apresentação da resposta, garantindo que cada tipo de interação seja tratado com a abordagem mais adequada às suas características específicas.

```

def get_rag_prompt_engineering(self, intention: str):
    # If there is no subject, set the subject to an empty string
    if not hasattr(self, "subject"):
        self.subject = ""
    # Get the RAG prompt engineering based on the intention
    if intention == "question_answering":
        print("Question Answering")
        return self.get_questions_rag_prompt(self.subject)
    elif intention == "summarization":
        print("Summarization")
        return self.get_summary_rag_prompt(self.subject)
    elif intention == "coding":
        print("Coding")
        return self.get_coding_rag_prompt(self.subject)
    elif intention == "searching_for_information":
        print("Searching for Information")
        return self.get_search_rag_prompt(self.subject)

```

Esta adaptação dinâmica baseada na intenção permite que o sistema ofereça respostas mais alinhadas com as necessidades específicas do estudante em cada interação, contribuindo para a eficácia da solução.

3.3 Geração Aumentada por Recuperação (RAG)

A implementação do sistema RAG constitui o núcleo metodológico deste trabalho, representando a abordagem principal para fornecer respostas precisas e contextualizadas aos estudantes. Esta seção detalha a metodologia desenvolvida para a construção do

sistema RAG adaptativo, abrangendo desde a geração de *embeddings* até a formulação de *prompts* especializados, com foco particular na adaptação dinâmica baseada em intenções.

3.3.1 Arquitetura Geral do Sistema RAG

O sistema RAG implementado segue uma arquitetura modular que integra diferentes componentes para criar um *pipeline* coeso de recuperação e geração. A estrutura foi projetada para maximizar a precisão das respostas enquanto mantém a eficiência computacional necessária para interações em tempo real.

O componente central desta arquitetura é a classe `StudentHelper`, que orquestra todo o processo de interação, desde a recepção da consulta do usuário até a entrega da resposta final. Esta classe atua como uma fachada para os componentes mais especializados, incluindo o gerenciador de recursos (`ResourceManager`), o processador de documentos PDF e o provedor de inferência para modelos de linguagem.

Como mostrado em 2.3, o fluxo de processamento segue uma sequência bem definida: (1) identificação da intenção da consulta, (2) geração de consultas aprimoradas para recuperação, (3) recuperação de contexto relevante, (4) formulação de *prompts* especializados, e (5) geração da resposta final. Cada uma destas etapas foi implementada com atenção às peculiaridades do domínio educacional, priorizando respostas que sejam não apenas precisas, mas também pedagogicamente úteis.

3.3.2 Geração e Indexação de *Embeddings* com BGE-M3

A geração de *embeddings* de alta qualidade é fundamental para a eficácia do sistema RAG. Após avaliação de diferentes modelos disponíveis, optou-se pelo modelo BGE-M3 (BAAI *General Embedding Model*, versão M3) desenvolvido pela *Beijing Academy of Artificial Intelligence*, devido ao seu desempenho superior em tarefas de recuperação semântica, particularmente em contextos multilíngues (CHEN et al., 2024).

O modelo BGE-M3 representa uma evolução significativa em relação a modelos anteriores de *embeddings*, oferecendo representações vetoriais que capturam nuances semânticas com maior precisão. Uma característica distintiva deste modelo é sua capacidade de gerar três tipos complementares de *embeddings*: densos, esparsos (lexicais) e ColBERT. Esta abordagem híbrida permite combinar as vantagens de diferentes paradigmas de recuperação, resultando em maior precisão e robustez.

A implementação da geração de *embeddings* foi realizada através da biblioteca `FlagEmbedding`, que oferece uma interface otimizada para o modelo BGE-M3. O processo começa após a fase de processamento de documentos PDF e segue estas etapas metodológicas: Inicialmente, o modelo BGE-M3 é carregado com configurações que priorizam o equilíbrio entre precisão e eficiência. A configuração inclui a definição de um comprimento máximo de consulta generoso (8192 *tokens*) para acomodar consultas complexas, e a utilização de precisão de ponto flutuante de 16 bits (FP16) para otimização de desempenho em hardware com suporte a operações tensoriais.

Para cada *chunk* de texto extraído dos documentos PDF, gera-se um *embedding* denso, que serve como a representação primária para fins de indexação e recuperação. Este *embedding* é uma representação vetorial de alta dimensionalidade (1024 dimensões) que captura o significado semântico global do texto. Os *embeddings* gerados são então armazenados em um banco de dados SQLite, juntamente com os metadados estruturais associados a cada *chunk*. Esta abordagem permite persistência eficiente e recuperação

rápida dos dados, além de facilitar a portabilidade do índice entre diferentes ambientes de execução.

Uma decisão metodológica importante foi a de não pré-calcular *embeddings* esparsos e ColBERT durante a fase de indexação, mas sim gerá-los dinamicamente durante a fase de recuperação apenas para os *chunks* mais promissores identificados pelo *embedding* denso. Esta abordagem representa um compromisso entre precisão e eficiência computacional, permitindo que o sistema utilize as capacidades avançadas do modelo BGE-M3 sem incorrer em custos proibitivos de armazenamento e processamento.

3.3.3 Estratégia de Recuperação Ponderada

Um aspecto metodológico distintivo do sistema implementado é a utilização de uma estratégia de recuperação ponderada que combina diferentes tipos de similaridade. Esta abordagem foi adaptada para aproveitar ao máximo as capacidades do modelo BGE-M3.

A função central para esta estratégia é a `compute_weighted_similarity`, que calcula a similaridade entre uma consulta q e os *chunks* de documentos d armazenados utilizando uma combinação ponderada de diferentes métricas:

$$\text{score}(q, d) = w_{\text{dense}} \cdot \text{simdense}(q, d) + w_{\text{sparse}} \cdot \text{simsparse}(q, d) + w_{\text{colbert}} \cdot \text{simcolbert}(q, d) \quad (3.1)$$

onde w_{dense} , w_{sparse} e w_{colbert} são pesos que somam 1.0, e as funções de similaridade correspondentes são:

$$\text{simdense}(q, d) = \frac{\vec{q}_{\text{dense}} \cdot \vec{d}_{\text{dense}}}{|\vec{q}_{\text{dense}}| |\vec{d}_{\text{dense}}|} \quad (3.2)$$

$$\text{simsparse}(q, d) = \sum_{k \in K_q \cap K_d} \text{sparse}_q(k) \cdot \text{sparse}_d(k) \quad (3.3)$$

$$\text{simcolbert}(q, d) = \sum_i i = 1^{|q|} \max_{j=1}^{|d|} \frac{\vec{q}_i \cdot \vec{d}_j}{|\vec{q}_i| |\vec{d}_j|} \quad (3.4)$$

onde K_q e K_d são os conjuntos de chaves nos *embeddings* esparsos da consulta e do documento, respectivamente.

Um aspecto inovador da implementação foi a adaptação dos pesos de similaridade com base na intenção identificada para a consulta. Especificamente:

- Para intenções de programação (*coding*), maior peso é atribuído à similaridade esparsa ($w_{\text{sparse}} = 0.4$), priorizando correspondências lexicais exatas que são cruciais para identificar exemplos de código e explicações de APIs.
- Para intenções de sumarização (*summarization*), maior peso é atribuído à similaridade densa ($w_{\text{dense}} = 0.5$), favorecendo a compreensão semântica global que é essencial para identificar conteúdo relevante para síntese.
- Para outras intenções, uma distribuição balanceada é utilizada ($w_{\text{dense}} = 0.4$, $w_{\text{sparse}} = 0.2$, $w_{\text{colbert}} = 0.4$), combinando as vantagens de diferentes tipos de similaridade.

Esta abordagem adaptativa representa uma contribuição metodológica significativa, permitindo que o sistema RAG otimize sua estratégia de recuperação de acordo com o objetivo específico da consulta do estudante.

3.3.4 Otimização de Consultas para Recuperação

Um desafio frequente em sistemas RAG educacionais é a discrepância entre a linguagem natural utilizada pelos estudantes em suas consultas e a terminologia técnica presente nos materiais didáticos. Para mitigar este problema, foi implementada uma metodologia de otimização de consultas que utiliza o próprio LLM para gerar consultas de recuperação aprimoradas.

Esta abordagem se baseia no conceito de reformulação de consultas utilizado em sistemas de recuperação de informação avançados, mas adaptado para o contexto educacional. O processo é iniciado pela função `generate_enhanced_query`, que recebe a consulta original do estudante e a transforma em um conjunto de consultas de recuperação otimizadas:

O LLM recebe instruções específicas para gerar até três consultas diferentes, cada uma focada em aspectos complementares do tópico abordado. As instruções enfatizam:

- A utilização de terminologia técnica e acadêmica formal, alinhada com a linguagem típica dos materiais didáticos.
- A inclusão de conceitos fundamentais, teoremas, leis e princípios relacionados ao tópico.
- A decomposição de consultas complexas em componentes mais simples e direcionados.
- A maximização da cobertura temática para aumentar a probabilidade de recuperar passagens relevantes.
- A geração de consultas diversificadas que abordam diferentes aspectos do mesmo tópico.

A implementação deste processo segue uma abordagem estruturada, onde as consultas geradas são formatadas com *tags* XML específicas para facilitar a extração programática. Cada consulta também pode incluir um direcionamento específico para determinado livro, quando apropriado.

As consultas otimizadas são então utilizadas para realizar buscas independentes, cujos resultados são combinados para formar um conjunto diversificado de passagens relevantes. Esta abordagem de buscas de várias consultas permite superar limitações associadas à utilização de uma única consulta, aumentando significativamente a revocação (*recall*) sem comprometer excessivamente a precisão.

Um aspecto metodológico importante é a capacidade do sistema de adaptar esta otimização de consultas com base no contexto conversacional. Quando o parâmetro `conversation=True` é utilizado, o sistema considera mensagens anteriores da conversa na reformulação, permitindo que a recuperação seja contextualmente relevante à discussão em andamento.

3.3.5 Geração de *Prompts* Adaptados por Intenção

A formulação de *prompts* para modelos de linguagem representa um componente crítico em sistemas RAG, influenciando diretamente a qualidade e relevância das respostas geradas. Neste trabalho, foi desenvolvida uma metodologia de engenharia de *prompts* adaptativa que varia de acordo com a intenção identificada para a consulta do estudante.

Esta abordagem parte do princípio de que diferentes objetivos educacionais requerem diferentes estratégias de apresentação e estruturação da informação. Para implementar esta visão, foram desenvolvidos *templates* de *prompts* especializados para cada categoria de intenção identificada pelo classificador.

O sistema implementa quatro funções distintas de geração de *prompts*, cada uma otimizada para uma intenção específica:

A função `get_questions_rag_prompt` gera *prompts* para intenções de resposta a perguntas. Estes *prompts* enfatizam a precisão factual, a citação adequada de fontes e a contextualização da resposta no material didático. As instruções incluem diretrizes específicas para lidar com consultas matemáticas, priorizando a explanação do método de solução em detrimento da simples apresentação do resultado.

A função `get_coding_rag_prompt` é especializada para consultas relacionadas à programação. Os *prompts* gerados enfatizam a apresentação de código funcional e completo, acompanhado de explicações detalhadas sobre sua lógica e funcionamento. Há também instruções específicas para priorizar códigos encontrados nos materiais didáticos quando disponíveis.

Para intenções de busca por informação, a função `get_search_rag_prompt` gera *prompts* que priorizam a localização precisa da informação nos materiais, incluindo referências a livros, capítulos, seções e números de página. Esta abordagem é particularmente útil quando o estudante busca navegar pelos materiais físicos ou digitais.

Finalmente, a função `get_summary_rag_prompt` cria *prompts* para tarefas de sumarização, enfatizando a concisão e clareza na apresentação de conceitos complexos. As instruções direcionam o modelo a produzir resumos que capturem os pontos essenciais do tópico enquanto mantêm a precisão técnica.

Uma característica compartilhada por todos estes templates é a instrução para utilizar formatação LaTeX para equações e variáveis matemáticas, garantindo que o conteúdo técnico seja apresentado com clareza e precisão. Esta atenção à formatação matemática é particularmente importante no contexto educacional, onde a representação adequada de conceitos matemáticos é crucial para a compreensão.

A função `get_rag_prompt_engineering` é responsável pela seleção do template apropriado, recebendo a intenção identificada e retornando o *prompt* especializado correspondente. Esta abordagem modular facilita a manutenção e expansão do sistema, permitindo a adição de novos templates para intenções adicionais no futuro.

3.3.6 Fluxo de Processamento Integrado

Um aspecto metodológico crucial do sistema desenvolvido é a integração coesa de todos os componentes em um fluxo de processamento unificado. Esta integração é implementada primariamente através da função `generate_response_conversation`, que orquestra todo o processo desde a recepção da consulta até a entrega da resposta final. O fluxo de processamento integrado segue uma sequência bem definida:

1. Inicialmente, a consulta do estudante é analisada pelo classificador de intenções, que determina seu objetivo primário (resposta a pergunta, sumarização, programação ou busca por informação).
2. Com base na intenção identificada, o sistema ajusta dinamicamente seus parâmetros de recuperação, selecionando o número apropriado de documentos a recuperar (maior para buscas por informação, menor para outras intenções).
3. Em seguida, o sistema gera consultas otimizadas para recuperação, considerando tanto a intenção identificada quanto o contexto conversacional quando disponível.
4. As consultas otimizadas são utilizadas para recuperar passagens relevantes dos materiais didáticos indexados, utilizando a estratégia de similaridade ponderada descrita anteriormente.

5. As passagens recuperadas são então combinadas com a consulta original e o *prompt* especializado para a intenção identificada, formando um *prompt* completo para o modelo de linguagem.
6. Este *prompt* é enviado ao provedor de inferência, que utiliza o modelo de linguagem especificado (padrão: GPT-4o-mini) para gerar a resposta final.
7. Finalmente, a resposta gerada e a consulta original são adicionadas ao histórico de conversação, permitindo a continuidade contextual da interação.

A abordagem integrada permite que cada componente do sistema contribua de forma sinérgica para o resultado final, adaptando-se dinamicamente às necessidades específicas de cada interação. Esta flexibilidade representou um dos objetivos centrais da metodologia desenvolvida, refletindo a natureza diversificada das necessidades de aprendizado dos estudantes.

3.3.7 Gestão de Contexto Conversacional

A capacidade de manter e utilizar contexto conversacional representa um aspecto metodológico importante do sistema desenvolvido, distinguindo-o de abordagens RAG mais básicas que tratam cada consulta de forma isolada. Esta funcionalidade é particularmente valiosa no contexto educacional, onde discussões frequentemente se desenvolvem de forma incremental, com consultas posteriores construindo sobre conceitos e explicações anteriores.

A gestão de contexto conversacional foi implementada através de um mecanismo de armazenamento de mensagens na classe `StudentHelper`. Este mecanismo mantém um histórico ordenado de mensagens trocadas entre o estudante e o sistema, preservando tanto as consultas quanto as respostas correspondentes.

O histórico conversacional é utilizado em dois pontos principais do fluxo de processamento:

- Durante a geração de consultas otimizadas, quando o parâmetro `conversation=True` é especificado. Neste caso, as mensagens anteriores são fornecidas ao LLM que gera as consultas de recuperação, permitindo que este considere o contexto da discussão em andamento.
- Na formulação do *prompt* final para geração de resposta, onde as mensagens anteriores são inseridas entre o *prompt* especializado (que contém instruções e contexto recuperado) e a consulta atual. Isto permite que o modelo de linguagem acesse toda a história da conversa ao formular sua resposta.

Este *design* modular permite que o sistema funcione tanto em modo “stateless” (tratando cada consulta isoladamente) quanto em modo conversacional, dependendo das necessidades da aplicação. A capacidade de alternar entre estes modos oferece flexibilidade para diferentes cenários de uso, desde consultas pontuais até sessões de estudo prolongadas.

Uma consideração metodológica importante foi a decisão de não incluir o *prompt* especializado completo (que contém instruções detalhadas e contexto recuperado) no histórico conversacional. Esta abordagem evita a acumulação de contexto redundante que poderia eventualmente exceder os limites de contexto do modelo de linguagem em conversas prolongadas.

3.3.8 Considerações sobre Escalabilidade e Desempenho

A metodologia desenvolvida considerou cuidadosamente aspectos de escalabilidade e desempenho, visando criar um sistema que pudesse funcionar eficientemente com volumes crescentes de dados e número de usuários.

Para assegurar a escalabilidade no processamento de documentos, a abordagem implementada separa claramente as fases de processamento e consulta. O processamento de documentos, que é computacionalmente intensivo, é realizado de forma assíncrona e seus resultados são persistidos em um banco de dados SQLite. Esta separação permite que o processamento seja realizado em lotes, em períodos de baixa demanda, sem impactar o desempenho do sistema durante a operação normal.

Na fase de consulta, várias otimizações foram implementadas para manter tempos de resposta aceitáveis mesmo com grandes volumes de dados. A estratégia de recuperação em duas fases representa um componente chave desta abordagem. Adicionalmente, o armazenamento de *embeddings* pré-computados e a seleção cuidadosa de índices no banco de dados SQLite contribuem para manter o desempenho em níveis adequados.

Para situações que requerem escalabilidade ainda maior, o sistema foi projetado para ser compatível com tecnologias de indexação vetorial mais avançadas, como FAISS (JOHNSON; DOUZE; JÉGOU, 2019) ou Milvus (WANG et al., 2021). Embora estas tecnologias não tenham sido implementadas na versão atual, a arquitetura modular do sistema permite sua incorporação futura com alterações mínimas no código.

No que diz respeito ao desempenho em termos de economia monetária e tempo de resposta, a metodologia implementada busca um equilíbrio entre qualidade, rapidez e custo. A seleção de modelos como GPT-4o-mini como padrão representa uma escolha consciente que prioriza tempos de resposta razoáveis e custo reduzido, mantendo um nível aceitável de qualidade. Para cenários onde maior qualidade é necessária, o sistema suporta a utilização de modelos mais avançados como GPT-4o ou Claude 3.5 Sonnet, com o conseqüente aumento no custo e tempo de processamento.

3.3.9 Arquitetura do Servidor e Interface de Usuário

A implementação prática do sistema RAG educacional foi estruturada como uma aplicação cliente-servidor, com uma clara separação entre o *backend*, responsável pela lógica de processamento, e o *frontend*, que gerencia a interface de usuário. Esta arquitetura facilita a manutenção, escalabilidade e adaptação do sistema para diferentes contextos de uso.

O servidor foi implementado utilizando o framework FastAPI (RAMÍREZ, 2018), escolhido por sua eficiência, suporte a operações assíncronas e facilidade de documentação automática da API. A estrutura do servidor segue o paradigma RESTful, com *endpoints* bem definidos para autenticação, processamento de consultas e gestão de dados.

O sistema de autenticação implementa um mecanismo de sessões baseado em *tokens* UUID, permitindo autenticação persistente sem a necessidade de revalidar credenciais a cada interação. Este mecanismo inclui funcionalidades de *timeout* e invalidação automática de sessões inativas, garantindo segurança sem comprometer a experiência do usuário.

Para garantir o isolamento de contexto entre diferentes usuários, o servidor mantém instâncias separadas da classe `StudentHelper` para cada sessão ativa. Estas instâncias são gerenciadas pelo componente `SessionManager`, que também é responsável por limpar instâncias associadas a sessões expiradas, prevenindo vazamentos de memória durante operação prolongada.

A interface de usuário foi desenvolvida utilizando React e Next.js, com componentes estilizados usando a biblioteca Tailwind CSS. Esta abordagem permitiu a criação de uma interface responsiva e acessível, que funciona adequadamente em diferentes dispositivos e tamanhos de tela.

Um aspecto distintivo da interface é sua adaptação para o contexto educacional, com funcionalidades específicas como:

- A possibilidade de definir o assunto de estudo, permitindo que o sistema adapte suas respostas ao domínio específico de interesse.
- Mecanismo de menção direta de livros, permitindo que o estudante referencie explicitamente materiais específicos em suas consultas.
- Visualização formatada das respostas, incluindo suporte a equações matemáticas (LaTeX) e código de programação com *syntax highlighting*.
- Histórico de conversação persistente, permitindo que estudantes retomem sessões de estudo anteriores.

A comunicação entre cliente e servidor é realizada através de requisições HTTP assíncronas, com formato JSON para troca de dados. Esta abordagem standard facilita a integração com diferentes clientes e a extensão do sistema para novos casos de uso.

Como demonstrado na Figura 3.1, a interface principal apresenta recursos de formatação e referência. O sistema destaca automaticamente as menções a livros específicos (como mostrado em verde na consulta sobre o livro de Aurélien Géron), renderiza equações matemáticas usando notação LaTeX ($\sigma(z) = \frac{1}{1+e^{-z}}$), aplica formatação em negrito para realces, e fornece referências precisas a capítulos e seções dos materiais didáticos. Esta integração de recursos visuais e estruturais facilita significativamente a compreensão e navegação.

3.3.10 Tratamento de Multilinguismo

Um aspecto metodológico importante do sistema desenvolvido é seu suporte nativo a operações multilíngues, com foco particular nos idiomas português e inglês. Esta capacidade é fundamental no contexto educacional brasileiro, onde materiais didáticos frequentemente transitam entre estes dois idiomas.

O suporte multilíngue foi implementado de forma integrada em todos os componentes do sistema:

- A geração de *embeddings* utiliza o modelo BGE-M3, que possui capacidades multilíngues robustas, permitindo representações vetoriais semanticamente equivalentes entre diferentes idiomas. Esta característica é particularmente valiosa para consultas que podem combinar terminologia em português e inglês, comum em áreas técnicas como computação e engenharia.
- O classificador de intenções foi treinado explicitamente com dados balanceados em ambos os idiomas, garantindo desempenho consistente independentemente do idioma utilizado pelo estudante em suas consultas.
- Os *templates* de *prompts* foram desenvolvidos com suporte explícito a respostas multilíngues, instruindo o modelo de linguagem a responder no mesmo idioma utilizado na consulta. Esta abordagem natural preserva o fluxo da conversação sem necessidade de tradução explícita.
- Para geração de respostas, foram selecionados modelos de linguagem com reconhecidas capacidades multilíngues, como GPT-4o, Claude 3 e DeepSeek, que demonstram desempenho robusto em português brasileiro.

Esta abordagem integrada para multilinguismo representa uma consideração metodológica essencial, reconhecendo a realidade do ambiente educacional onde este sistema será utilizado. Em vez de tratar o suporte a múltiplos idiomas como uma funcionalidade adicional, ele foi incorporado como um requisito fundamental desde as fases iniciais de *design* do sistema.

AcademiCK
 Auxiliando seus estudos
 Desenvolvido por Cláudio Klautau Mello

Assunto: Machine Learning

[\[> Sair\]](#)

Logado como:
claudiomello

[Deletar Chat](#)

AcademiCK

Bem-vindo ao AcademiCK!

Estou aqui para ajudá-lo a estudar baseado nos materiais fornecidos.

Posso auxiliar você a encontrar conteúdos específicos dentro do livro, explicar conceitos, fornecer detalhes de implementação e criar exemplos de código. Fique à vontade para fazer qualquer pergunta sobre o conteúdo!

Você

AcademiCK

A função de ativação sigmoide, também conhecida como função logística, é uma função que transforma a entrada em um valor entre 0 e 1. Esta função tem a seguinte forma:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Quando a entrada z se torna muito grande (positiva ou negativa), a função se aproxima de 1 ou 0, respectivamente. Isso significa que a derivada da função se torna muito próxima de zero, resultando em dificuldades para o algoritmo de retropropagação, pois não haverá gradiente suficiente para atualizar os pesos das camadas da rede neural, o que é conhecido como o problema dos gradientes que desaparecem.

No livro, Aurélien Géron menciona que a função sigmoide "tem um derivado bem definido e não nulo em todos os pontos, permitindo que o método de descida do gradiente avance a cada passo". No entanto, essa mesma função pode causar problemas em redes neurais profundas, pois a variação dos dados se torna mais elevada à medida que se avança nas camadas da rede, levando à saturação da função.

Você pode aprender mais sobre a função de ativação sigmoide no seguinte trecho do livro:

- Livro: 2022_Aurélien Géron - Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems - O'Reilly Media (2022)
- Capítulo: Capítulo 10. Introdução às Redes Neurais Artificiais com Keras
- Seção: A Multilayer Perceptron e a Retropropagação

Além disso, o problema dos gradientes que desaparecem e a saturação da função sigmoide também são discutidos no:

- Capítulo: Capítulo 11. Treinando Redes Neurais Profundas
- Seção: Os Problemas dos Gradientes que Desaparecem/Explodem

Você

Faça uma pergunta... (use @ para mencionar um livro)

GPT-4o Mini

Figura 3.1 – Interface principal do AcademiCK

3.3.11 Adaptabilidade a Diferentes Domínios de Conhecimento

Um aspecto distintivo da metodologia implementada é a adaptabilidade do sistema a diferentes domínios de conhecimento. Esta característica foi considerada essencial para um assistente educacional, dada a diversidade de disciplinas e tópicos que estudantes podem precisar explorar.

A adaptabilidade a diferentes domínios é implementada através de vários mecanismos: A classe `StudentHelper` mantém um atributo `subject` que pode ser definido explicitamente pelo usuário ou inferido do contexto da conversação. Este atributo é incorporado nos *prompts* enviados ao modelo de linguagem, permitindo respostas contextualizadas ao domínio específico de estudo.

A arquitetura de armazenamento permite a indexação de materiais didáticos de diferentes disciplinas, mantendo metadados que identificam o livro e o tópico específico de cada passagem. Durante a recuperação, esta informação pode ser utilizada para priorizar materiais relevantes ao domínio atual de estudo.

A geração de consultas otimizadas considera o domínio específico, incorporando terminologia técnica apropriada para aumentar a precisão da recuperação. Esta adaptação é particularmente valiosa para domínios altamente especializados, onde a terminologia técnica difere significativamente da linguagem cotidiana.

Os *templates* de *prompts* incluem o domínio atual como contexto, orientando o modelo de linguagem a adaptar suas respostas ao nível de tecnicidade e formalismo apropriado para a disciplina em questão.

Esta adaptabilidade permite que o mesmo sistema base seja utilizado efetivamente para assistência em diversas áreas, desde disciplinas técnicas como programação e matemática até campos mais conceituais como filosofia ou literatura. A abordagem implementada não requer retreinamento ou reconfiguração significativa para transitar entre diferentes domínios, dependendo primariamente dos materiais didáticos disponíveis para indexação.

3.3.12 Integração com Interface Administrativa

Para facilitar o gerenciamento do sistema em ambientes educacionais, a metodologia implementada incluiu o desenvolvimento de uma interface administrativa específica. Esta interface permite operações de manutenção e monitoramento essenciais para a operação contínua do sistema em contexto de produção.

A interface administrativa foi implementada como um módulo separado da interface principal do estudante, com controles de acesso restritos a usuários com privilégios administrativos. Esta separação clara de responsabilidades segue princípios estabelecidos de segurança e gerenciamento de sistema.

As funcionalidades principais implementadas na interface administrativa incluem:

- **Gestão de conteúdo:** *Upload* e processamento de novos materiais didáticos, visualização e gerenciamento dos materiais existentes, e estatísticas detalhadas sobre o *corpus* indexado.
- **Gestão de usuários:** Criação, edição e desativação de contas de usuário, definição de níveis de acesso, e monitoramento de atividade.
- **Monitoramento de sistema:** Visualização de métricas de uso, desempenho e carga do sistema, facilitando a identificação proativa de possíveis gargalos ou problemas.

Conforme ilustrado na Figura 3.2, o painel administrativo oferece uma visão abrangente do estado do sistema, incluindo métricas quantitativas sobre o conteúdo processado e ferramentas para gestão da biblioteca de materiais. A interface apresenta

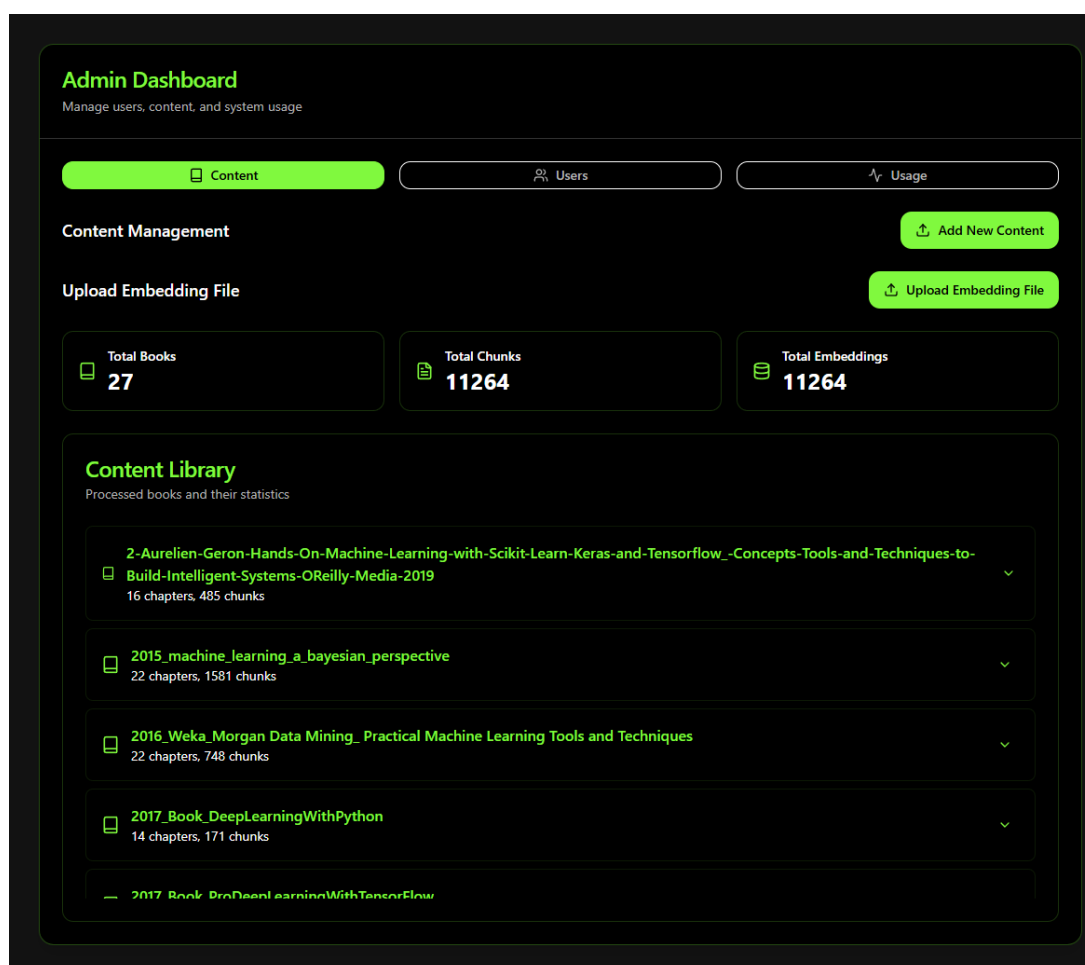


Figura 3.2 – Interface administrativa do sistema AcademiCK

de forma clara as estatísticas fundamentais do sistema, como o número total de livros indexados (27), *chunks* de texto processados (11.264) e *embeddings* gerados, facilitando o monitoramento da capacidade e utilização do sistema.

A implementação técnica da interface administrativa segue a mesma *stack* tecnológica da interface principal (React, Next.js, Tailwind CSS), mantendo consistência no desenvolvimento e facilitando a manutenção. No entanto, ela incorpora componentes específicos para visualização de dados administrativos, como gráficos de uso e tabelas de gestão.

A comunicação entre a interface administrativa e o servidor utiliza *endpoints* API dedicados, com verificações adicionais de autorização para garantir que apenas usuários com privilégios apropriados possam executar operações sensíveis. Esta camada adicional de segurança é essencial para proteger a integridade do sistema e a privacidade dos dados dos usuários.

Em termos metodológicos, a inclusão de uma interface administrativa robusta representa o reconhecimento da importância de ferramentas de gestão adequadas para sistemas educacionais. Esta abordagem facilita não apenas a manutenção técnica do sistema, mas também sua adaptação contínua às necessidades pedagógicas da instituição onde é implantado.

3.3.13 Considerações Éticas na Implementação

O desenvolvimento do sistema RAG educacional foi guiado por considerações éticas explícitas, reconhecendo as responsabilidades especiais associadas a tecnologias aplicadas em contextos de aprendizagem. Estas considerações informaram decisões metodológicas em todas as fases do desenvolvimento.

Um aspecto ético fundamental abordado foi a transparência do sistema. Ao contrário de abordagens que podem ocultar a natureza automatizada das respostas, o sistema foi projetado para ser explícito sobre sua natureza como assistente baseado em IA. Esta transparência se estende também às limitações do sistema, com respostas que reconhecem explicitamente situações onde a informação disponível é insuficiente ou onde há incerteza significativa.

A questão da precisão factual recebeu atenção metodológica particular, dado seu impacto potencial no processo de aprendizagem. A abordagem RAG foi selecionada especificamente por sua capacidade de ancorar respostas em fontes confiáveis, reduzindo o risco de alucinações que poderiam induzir os estudantes a erro. Adicionalmente, o sistema foi projetado para sempre citar suas fontes, permitindo que estudantes verifiquem as informações apresentadas diretamente nos materiais didáticos.

O sistema foi implementado para armazenar apenas os dados mínimos necessários para sua funcionalidade. Todas as análises de uso são realizadas em dados agregados e anonimizados, protegendo a privacidade individual.

A acessibilidade também representou uma consideração ética importante, reconhecendo a diversidade de necessidades e habilidades dos estudantes. O suporte multilíngue nativo (português e inglês) representa um esforço consciente para ampliar o acesso a estudantes com diferentes bagagens linguísticas.

Finalmente, o sistema foi projetado para complementar, não substituir, outras modalidades de aprendizagem. Esta abordagem reconhece os limites fundamentais de sistemas automatizados no processo educacional e a importância contínua de interações humanas significativas. As respostas geradas frequentemente incluem sugestões para aprofundamento através de discussões com professores ou colegas, reconhecendo o valor destes intercâmbios para a consolidação da aprendizagem.

3.3.14 Síntese Metodológica

Em síntese, a metodologia implementada para o sistema RAG educacional representa uma abordagem integrada que combina técnicas avançadas de processamento de linguagem natural com considerações pedagógicas específicas. O sistema incorpora inovações metodológicas em múltiplos níveis:

- Na fase de processamento de documentos, através de uma abordagem de extração estruturada que preserva o contexto hierárquico e semântico dos materiais didáticos.
- Na geração de *embeddings*, através da utilização do modelo BGE-M3 que oferece representações vetoriais ricas que capturam nuances semânticas em múltiplos idiomas.
- Na classificação de intenções, através de um modelo especializado treinado com dados educacionais que permite adaptar todo o *pipeline* subsequente ao objetivo específico do estudante.
- Na estratégia de recuperação, através de uma abordagem ponderada que adapta dinamicamente a importância relativa de diferentes tipos de similaridade conforme a intenção identificada.

- Na engenharia de *prompts*, através de *templates* especializados que otimizam a apresentação da informação para diferentes objetivos de aprendizagem.
- Na arquitetura de sistema, através de um *design* modular que facilita manutenção, extensão e adaptação para diferentes contextos educacionais.

Esta abordagem metodológica holística resultou em um sistema que não apenas recupera informações relevantes, mas o faz de forma contextualizada e pedagogicamente eficaz. A adaptação dinâmica baseada na intenção do usuário representa uma contribuição particularmente significativa, permitindo que o sistema responda de maneira diferenciada a diversas necessidades educacionais.

As decisões metodológicas descritas neste capítulo foram informadas tanto por considerações técnicas quanto pedagógicas, buscando um equilíbrio que maximize o valor educacional do sistema enquanto mantém viabilidade prática de implementação e operação. O resultado é uma abordagem que demonstra o potencial de sistemas RAG adaptados especificamente para o domínio educacional, apontando caminhos promissores para o desenvolvimento futuro de assistentes de estudo baseados em inteligência artificial.

4 RESULTADOS

Este capítulo apresenta os resultados obtidos a partir da implementação e avaliação do sistema de *chatbot* educacional baseado em RAG. Os resultados compreendem análises quantitativas e qualitativas do desempenho do sistema, com foco particular na eficácia do mecanismo de recuperação de informações, na qualidade das respostas geradas e na capacidade do sistema de se adaptar às diferentes intenções dos usuários identificadas pelo classificador.

4.1 Avaliação do Sistema RAG

A avaliação do sistema foi realizada utilizando a plataforma DeepEval (IP; AI, 2024), que implementa a metodologia “LLM as a Judge” (LLM como juiz) para avaliação automatizada de sistemas RAG. Esta abordagem foi selecionada devido à impraticabilidade de conduzir avaliações humanas extensivas no escopo temporal deste trabalho. Foram utilizados três modelos distintos: GPT-4.1-nano para recuperação, GPT-4o-mini para geração de respostas e GPT-4o como modelo juiz avaliador. Esta configuração implica uma certa imparcialidade avaliativa, uma vez que o modelo avaliador possui capacidades superiores aos modelos do *pipeline* operacional.

O DeepEval implementa um processo estruturado onde o modelo juiz recebe contexto específico sobre os critérios de avaliação e executa uma análise sistemática para cada métrica. Embora esta metodologia ofereça avaliação consistente, padronizada e escalável, eliminando variabilidade inter-avaliador comum em avaliações humanas, reconhece-se que a avaliação automática pode não capturar completamente nuances pedagógicas ou aspectos subjetivos da qualidade educacional que seriam evidentes para avaliadores humanos especializados. Os resultados devem ser interpretados como indicadores robustos de desempenho técnico, complementando a necessidade de validação pedagógica em contextos educacionais reais.

4.2 Métricas de Avaliação

O processo de avaliação foi estruturado utilizando um conjunto de 99 pares pergunta-resposta no domínio de aprendizado de máquina, extraídos do dataset “prsdm/ Machine-Learning-QA-dataset” (MAHAMULKAR, 2024), onde para cada consulta o sistema AcademiCK executou seu *pipeline* completo enquanto o modelo GPT-4o atuou como juiz avaliador.

- **Relevância de Resposta (*Answer Relevancy*):** avalia o grau de alinhamento entre a resposta gerada e a pergunta do usuário, independentemente do contexto recuperado. O modelo juiz analisa a correspondência semântica entre pergunta e resposta, considerando completude, precisão e adequação da informação fornecida. Esta métrica indica a capacidade do sistema de gerar respostas que efetivamente abordam a consulta do estudante.
- **Relevância Contextual (*Contextual Relevancy*):** mensura a adequação do contexto recuperado em relação à consulta do usuário. O processo de avaliação examina se o contexto recuperado pelo sistema RAG é apropriado para responder à consulta específica, analisando a conexão temática entre a pergunta e as passagens recuperadas. Esta métrica é crucial para avaliar a eficácia do componente de recuperação do

sistema RAG, determinando se o sistema está selecionando passagens apropriadas dos materiais didáticos disponíveis.

- **Fidelidade (*Faithfulness*)**: avalia o grau em que a resposta gerada se mantém fiel às informações presentes no contexto recuperado, sem adicionar informações não fundamentadas ou contraditórias. O modelo juiz verifica se a resposta permanece consistente com as informações do contexto recuperado, identificando possíveis adições de informações não fundamentadas, contradições ou distorções do conteúdo original. Esta métrica é essencial para garantir a confiabilidade educacional do sistema.

O modelo juiz avaliador produz pontuações em uma escala de 0 a 1 para cada métrica, onde valores mais altos indicam melhor desempenho. Foram definidos limiares (*thresholds*) específicos considerando práticas da literatura de avaliação RAG e as demandas do contexto educacional: 0,5 para Relevância Contextual, mantendo o padrão do *framework* DeepEval dado que o sistema visa apresentar fundamentação contextual além de respostas diretas, justificando menor rigor na inclusão de contexto potencialmente útil; 0,7 para Fidelidade, baseando-se na metodologia estatisticamente rigorosa de (BHASKARJIT et al., 2024), que propõe técnicas de validação cruzada e análise de confiança para determinação ótima de *thresholds* em métricas de LLM; e 0,7 para Relevância de Resposta, adotando critério mais rigoroso que o padrão de 0,5 do DeepEval devido às demandas específicas do ambiente educacional, onde respostas inadequadas podem comprometer o processo de aprendizagem.

4.3 Resultados Quantitativos

Esta seção apresenta os resultados quantitativos obtidos durante a avaliação do sistema desenvolvido, abrangendo o desempenho do classificador de intenções e a eficácia geral do sistema.

4.3.1 Resultados do Classificador de Intenções

Um aspecto distintivo do sistema desenvolvido é sua capacidade de adaptar o comportamento de recuperação e geração de respostas com base na intenção identificada da consulta do usuário. Embora, devido à dificuldade de avaliação, não tenham sido realizados testes quantitativos que discriminem o desempenho por categoria de intenção, a implementação mostrou-se robusta e eficiente nos casos testados.

O funcionamento eficaz da adaptação baseada em intenções depende fundamentalmente da precisão do classificador responsável por identificar a categoria de cada consulta. O classificador desenvolvido demonstrou bons resultados durante a avaliação no conjunto de teste, conforme apresentado na Tabela 4.1.

As métricas de avaliação foram calculadas em três variações distintas para fornecer uma análise abrangente do desempenho do classificador. O *score* macro calcula a média aritmética simples das métricas individuais de cada classe, tratando todas as categorias de forma igualitária independentemente de sua frequência no conjunto de dados. O *score* micro considera a contribuição de cada instância individual, calculando as métricas globalmente através da soma dos verdadeiros positivos, falsos positivos e falsos negativos de todas as classes. Já o *score* weighted pondera as métricas de cada classe pelo número de instâncias verdadeiras dessa classe, fornecendo uma média que reflete a distribuição real dos dados.

Os resultados evidenciam um bom desempenho do classificador, alcançando uma acurácia de 97,61% no conjunto de teste. As métricas F1-score apresentaram valores consistentemente superiores a 97,6% em todas as variações (macro, micro e weighted),

Tabela 4.1 – Métricas de desempenho do classificador de intenções no conjunto de teste

Métrica	Valor
Acurácia	0,9761
Loss	0,2448
F1-Score (Macro)	0,9768
F1-Score (Micro)	0,9761
F1-Score (Weighted)	0,9760
Precisão (Macro)	0,9784
Precisão (Micro)	0,9761
Precisão (Weighted)	0,9781
Recall (Macro)	0,9773
Recall (Micro)	0,9761
Recall (Weighted)	0,9761

demonstrando a capacidade do modelo de distinguir eficazmente entre as quatro categorias de intenção: busca por informações, resposta a perguntas, programação e sumarização.

A consistência entre as diferentes métricas indica que o classificador apresenta desempenho equilibrado para todas as classes de intenção, sem viés significativo para categorias específicas. Particularmente, a proximidade entre os valores de F1-score macro (0,9768) e micro (0,9761) sugere que o modelo mantém alta precisão mesmo considerando possíveis variações na representatividade das classes no conjunto de dados.

A alta precisão alcançada pelo classificador de intenções constitui um fundamento sólido para o funcionamento do sistema RAG adaptativo. Durante os casos testados, o classificador demonstrou capacidade consistente de identificar corretamente a intenção subjacente às consultas dos usuários, habilitando as adaptações específicas implementadas para cada categoria.

Esta precisão na classificação traduz-se diretamente na qualidade das adaptações realizadas pelo sistema, uma vez que cada categoria de intenção aciona estratégias diferenciadas de recuperação de contexto e geração de respostas, otimizadas para as necessidades específicas de cada tipo de consulta educacional.

É importante ressaltar que, embora estas adaptações tenham sido implementadas com base em princípios estabelecidos na literatura sobre sistemas RAG e considerações pedagógicas específicas, a avaliação formal do impacto diferencial destas adaptações por categoria de intenção representa uma direção promissora para pesquisas futuras.

4.3.2 Resultados do sistema

Os resultados da avaliação revelaram um desempenho satisfatório do sistema em todas as métricas avaliadas. A Figura 4.1 apresenta as médias gerais obtidas para cada métrica.

O sistema alcançou uma pontuação média de 0,99 em Relevância de Resposta, superando significativamente o limiar estabelecido de 0,7. Este resultado indica que as respostas geradas pelo sistema estão consistentemente alinhadas com as consultas, demonstrando uma excelente compreensão das necessidades informacionais expressas nas perguntas.

Em relação à Relevância Contextual, o sistema obteve uma pontuação média de 0,58, superando o limiar de 0,5 estabelecido para esta métrica. Este valor mais baixo em comparação às outras métricas decorre da estratégia de recuperação implementada,

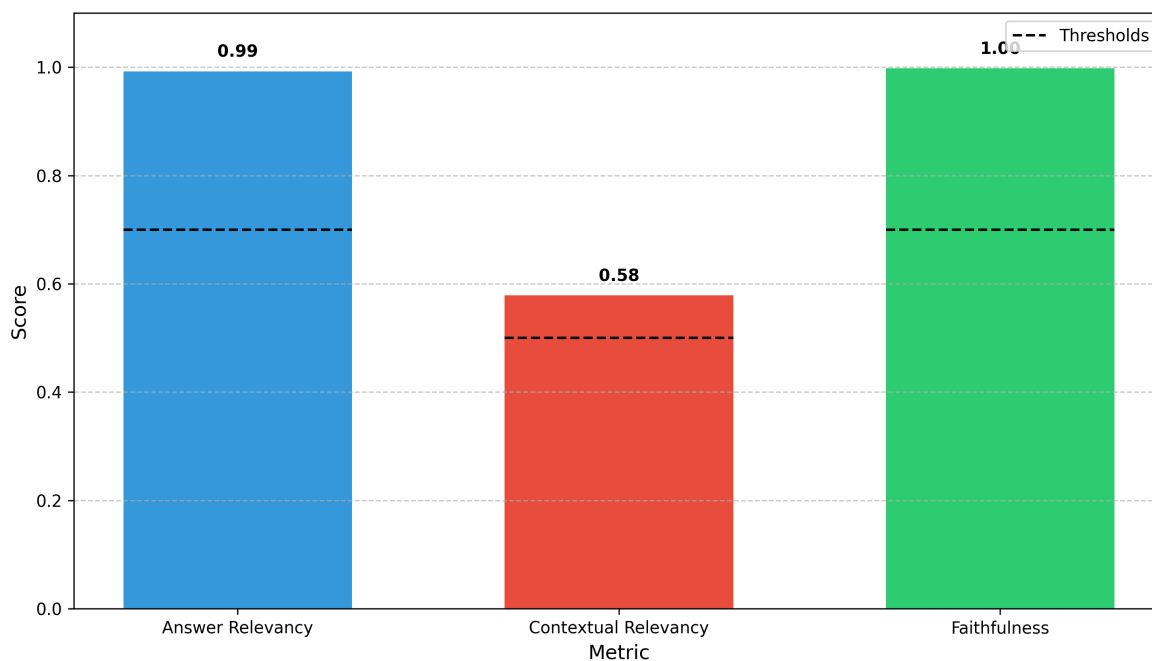


Figura 4.1 – Médias das métricas de avaliação do sistema RAG

que prioriza a diversidade e abrangência do contexto recuperado. A decisão metodológica de recuperar um conjunto mais amplo de informações, incluindo conteúdos adjacentes e complementares para facilitar a compreensão holística do tópico pelo estudante, resulta na inclusão de passagens que, embora pedagogicamente valiosas, são avaliadas como menos diretamente relevantes para a consulta específica.

A métrica de Fidelidade apresentou o resultado mais expressivo, com uma pontuação média de 1,0, indicando que as respostas geradas mantêm-se consistentemente fiéis às informações presentes no contexto recuperado. Este resultado é particularmente importante no contexto educacional, onde a precisão factual é essencial para a credibilidade e utilidade do sistema como ferramenta de apoio ao aprendizado.

A Figura 4.2 apresenta a distribuição das pontuações em relação aos limiares estabelecidos, revelando que, de um total de 99 consultas avaliadas, 98 (99%) atingiram ou superaram o limiar para Relevância de Resposta, 66 (67%) para Relevância Contextual e 99 (100%) para Fidelidade.

A análise detalhada da distribuição das pontuações, ilustrada na Figura 4.3, revela padrões interessantes. Enquanto as métricas de Relevância de Resposta e Fidelidade apresentam distribuições concentradas em valores muito altos (próximos a 1,0), a Relevância Contextual mostra uma distribuição mais dispersa, com uma concentração significativa entre 0,4 e 0,7. Esta variação na Relevância Contextual é também explicada pela natureza do mecanismo de recuperação implementado, que deliberadamente prioriza a recuperação de um conjunto mais amplo e diversificado de informações.

4.4 Resultados Qualitativos

Além das métricas quantitativas apresentadas, foram conduzidas análises qualitativas das respostas geradas pelo sistema para diferentes tipos de consultas educacionais relacionadas ao domínio de aprendizado de máquina. Estas análises fornecem informações sobre o comportamento do sistema em cenários reais de uso e demonstram a eficácia da

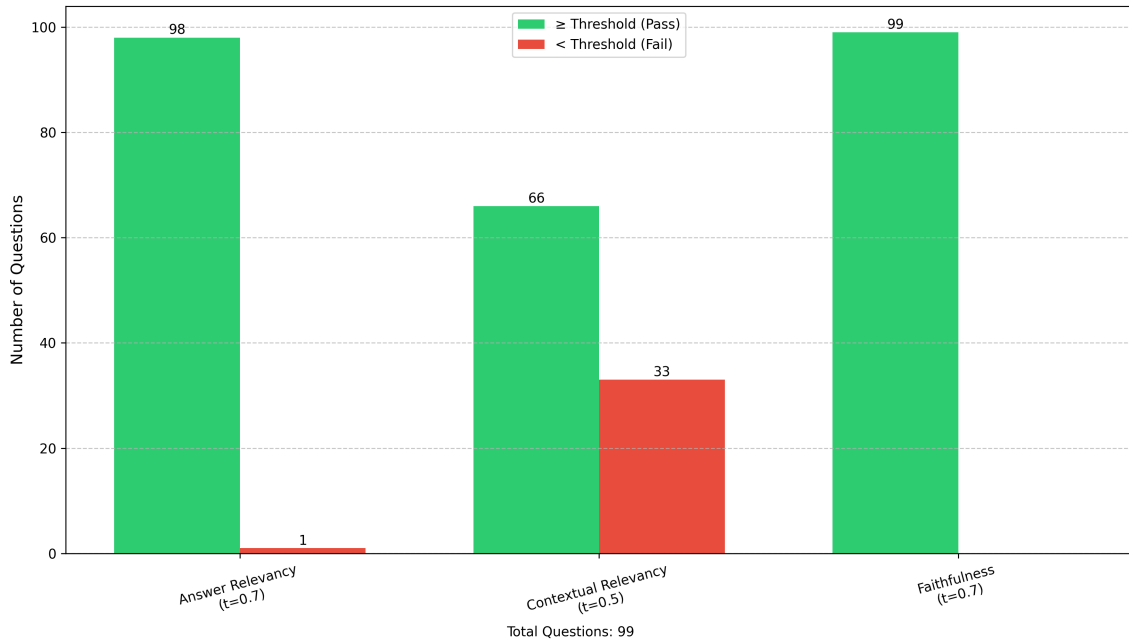


Figura 4.2 – Distribuição de consultas acima e abaixo dos limiares estabelecidos

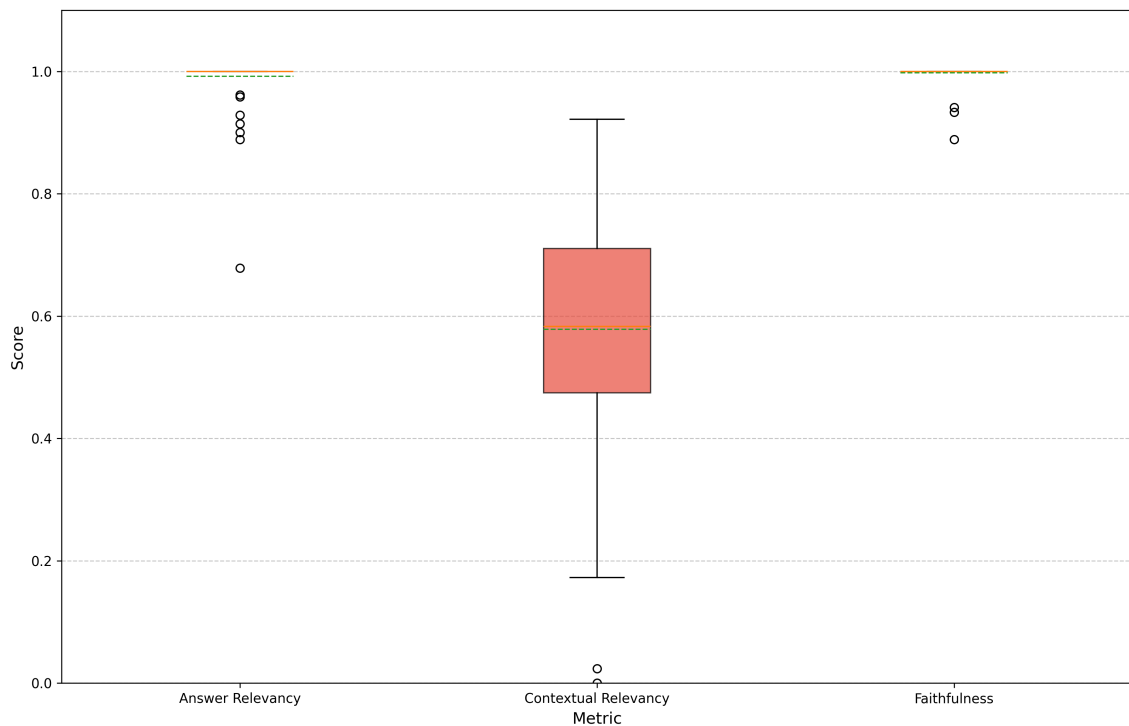


Figura 4.3 – Distribuição das pontuações por métrica de avaliação

adaptação baseada em intenções implementada.

4.4.1 Desempenho por Categoria de Intenção

O sistema demonstrou comportamentos distintos e apropriados para cada uma das categorias de intenção identificadas pelo classificador treinado. A seguir, são apresentados exemplos representativos para cada categoria, ilustrando como o sistema adapta suas estratégias de recuperação e geração de respostas.

4.4.1.1 Busca por Informações (*Searching for Information*)

Para consultas classificadas como busca por informações, o sistema prioriza a recuperação abrangente de materiais relevantes e fornece referências detalhadas aos estudantes. Um exemplo representativo foi a consulta: “*Estou com dificuldades de entender Redes neurais convolucionais. Me ajude a achar material sobre esse tópico*”.

Como mostrado na Figura 4.4, o sistema respondeu com uma lista estruturada de cinco fontes diferentes, cada uma com referências precisas incluindo livro, capítulo, seção e número da página. Esta resposta demonstra a capacidade do sistema de não apenas localizar materiais relevantes, mas também de fornecer resumos contextualizados que orientam o estudante sobre o conteúdo de cada referência, facilitando a seleção do material mais apropriado para suas necessidades específicas.

4.4.1.2 Resposta a Perguntas (*Question Answering*)

Para consultas de resposta a perguntas, o sistema foca em fornecer explicações diretas e didáticas, mantendo as citações das fontes utilizadas. Um exemplo foi a pergunta: “*Como funcionam as camadas convolucionais?*”.

Como mostrado na Figura 4.5, o sistema gerou uma resposta estruturada que incluiu:

1. Uma explicação conceitual clara da operação de convolução
2. Descrição da aprendizagem de recursos por cada filtro
3. Explicação de funções de ativação e *pooling*
4. Indicações de material para aprofundamento

Esta abordagem demonstra como o sistema equilibra a necessidade de fornecer uma explicação completa e autocontida com a importância de dirigir o estudante às fontes originais para aprofundamento.

4.4.1.3 Programação (*Coding*)

Para consultas relacionadas à implementação de código, o sistema adapta sua resposta para incluir exemplos práticos funcionais, acompanhados de explicações detalhadas e referências aos materiais didáticos utilizados. Um exemplo foi a solicitação: “*Faça um código de rede neural para classificação*”.

Como mostrado na Figura 4.6, o sistema respondeu com um código completo e funcional. A resposta incluiu não apenas o código, mas também uma explicação da implementação e uma citação de fontes que podem auxiliar na criação.

4.4.1.4 Sumarização (*Summarization*)

Para consultas que requerem respostas amplas, o sistema demonstrou capacidade de fornecer informações de forma sintética. Um exemplo foi a pergunta: “*Resuma redes neurais convolucionais*”.

Como mostrado na Figura 4.7, o sistema respondeu de forma completa e coesa.

4.4.2 Capacidade de Contextualização e Aprofundamento

Uma característica observada nas análises qualitativas foi a capacidade do sistema de fornecer contextualizações apropriadas baseadas na natureza da consulta. Para consultas

conceituais mais complexas, o sistema demonstrou habilidade de engajar em diálogos contínuos, fornecendo esclarecimentos adicionais quando solicitado.

Por exemplo, após a explicação inicial sobre redes neurais convolucionais, quando questionado especificamente sobre “*Como o pooling é feito?*”, o sistema foi capaz de aprofundar o tópico com uma explicação mais detalhada e técnica, mantendo a coerência com a explicação anterior e fornecendo novas referências bibliográficas relevantes, como mostrado na Figura 4.8.

4.4.3 Qualidade das Citações e Referenciamento

Um aspecto consistentemente observado em todas as categorias de intenção foi a precisão e completude das citações fornecidas pelo sistema. Todas as respostas analisadas incluíram referências estruturadas contendo:

- Nome completo do livro ou material didático
- Capítulo específico
- Seção ou subseção relevante
- Números de página quando disponíveis
- Resumos contextualizados do conteúdo referenciado

Esta consistência no referenciamento demonstra a eficácia do processamento estruturado de documentos PDF implementado e a capacidade do sistema de manter a rastreabilidade das informações utilizadas, aspecto fundamental para aplicações educacionais onde a verificabilidade das fontes é crucial.

4.4.4 Adaptação Linguística e Terminológica

O sistema demonstrou capacidade consistente de utilizar terminologia técnica apropriada ao domínio de aprendizado de máquina, mantendo um nível de formalidade adequado ao contexto educacional. A linguagem utilizada mostrou-se acessível para estudantes, evitando jargões desnecessários enquanto preservava a precisão técnica das explicações.

Claro! Aqui estão alguns livros e seções que você pode consultar para entender melhor as Redes Neurais Convolucionais (CNNs):

1. Livro: Deep Learning for Computer Vision with Python

- **Capítulo:** 11 Convolutional Neural Networks
- **Seção:** 11.3 Common Architectures and Training Patterns
- **Página:** 126
- **Resumo:** Este capítulo discute a estrutura básica das CNNs, incluindo as principais camadas (CONV, POOL, RELU e FC) e como elas se empilham para formar uma arquitetura CNN.

2. Livro: Deep Learning with Python

- **Capítulo:** 5 Convolutional Neural Networks
- **Seção:** Intuition behind CNNs
- **Página:** 78
- **Resumo:** Esta seção fornece uma visão geral intuitiva das CNNs, abordando conceitos fundamentais como operação de convolução, operação de pooling e a hierarquia que as CNNs são capazes de aprender.

3. Livro: Neural Networks and Deep Learning

- **Capítulo:** 8 Convolutional Neural Networks
- **Seção:** 8.5.2 Convolutional Autoencoders
- **Página:** 358
- **Resumo:** Esta seção discute a operação de convolução e apresenta as CNNs numa perspectiva de aprendizado não supervisionado, incluindo a visualização e o entendimento dos diferentes níveis de abstração nas redes.

4. Livro: Pro Deep Learning with TensorFlow

- **Capítulo:** 3 Convolutional Neural Networks
- **Seção:** Convolution Neural Networks
- **Página:** 30
- **Resumo:** O capítulo detalha a arquitetura típica das CNNs, a importância das camadas de convolução e como os filtros são aprendidos através do treinamento.

5. Livro: Python Deep Learning

- **Capítulo:** 3 Deep Learning Fundamentals
- **Seção:** Chapter Introduction
- **Página:** 3
- **Resumo:** Embora seja uma introdução ao deep learning, este capítulo discute as redes neurais profundas e suas múltiplas camadas, estabelecendo a base para compreender como as CNNs se encaixam nessa estrutura.

Explorar esses materiais vai ajudá-lo a compreender a teoria e a prática das Redes Neurais Convolucionais. Se você tiver dúvidas específicas enquanto lê, sinta-se à vontade para perguntar!

Figura 4.4 – Resposta do AcademiCK sobre Redes Neurais Convolucionais

As camadas de convolução em redes neurais convolucionais (CNNs) desempenham um papel crucial na extração de características de dados de entrada, como imagens. Aqui estão os principais aspectos do funcionamento dessas camadas:

1. Operação de Convolução: A camada de convolução utiliza um filtro (ou kernel) que desliza sobre a imagem de entrada. A operação de convolução envolve multiplicar os valores do filtro pelos valores correspondentes da região da imagem que está sendo coberta, somando os resultados para produzir um único número de saída. Este processo é repetido para todas as posições da imagem, resultando em um mapa de recursos que destaca características específicas detectadas pelo filtro.

2. Recursos Aprendidos: Cada filtro na camada de convolução é treinado para detectar certas características, como bordas, texturas ou padrões. Dividindo a imagem em regiões locais, a camada de convolução captura informações espaciais e preserva a estrutura da imagem.

3. Camadas Múltiplas: Geralmente, múltiplas camadas de convolução são empilhadas em uma CNN. À medida que se avança pelas camadas, os filtros tornam-se mais complexos, detectando características de nível mais alto. Por exemplo, as primeiras camadas podem detectar bordas, enquanto as camadas mais profundas podem capturar formas mais complexas, como objetos completos.

4. Ativação e Pooling: Após a operação de convolução, geralmente uma função de ativação (como ReLU) é aplicada para introduzir não-linearidade. Depois, uma camada de pooling é frequentemente utilizada para reduzir a dimensionalidade do mapa de características, retenendo as informações mais importantes e tornando o modelo menos suscetível a variáveis de localização.

Se você quer se aprofundar mais sobre as camadas de convolução, você pode consultar o seguinte trecho:

- **“Neural Networks and Deep Learning” (2018), de A. Aggarwal**
 - **Capítulo 1:** An Introduction to Neural Networks
 - * **Seção 1.6.5:** Convolutional Neural Networks
- **“Neural Networks and Deep Learning” (2018), de A. Aggarwal**
 - **Capítulo 8:** Convolutional Neural Networks
 - * **Seção 8.2:** The Basic Structure of a Convolutional Network
- **“Deep Learning for Computer Vision with Python” (2017), de A. Rosebrock**
 - **Capítulo 11:** Convolutional Neural Networks
 - * **Seção 11.2.1:** Layer Types

Essas seções oferecem uma explicação detalhada sobre o funcionamento e a estrutura das camadas de convolução nas redes neurais convolucionais.

Figura 4.5 – Resposta do AcademiCK sobre Camadas de Convolução em CNNs

Aqui está um código Python para uma rede neural simples usando a biblioteca Keras, que é uma interface de alto nível para construir e treinar modelos de aprendizado profundo. Esta rede é adequada para classificação binária.

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Gerar um conjunto de dados sintético para classificação
X, y = make_classification(n_samples=1000, n_features=20, n_classes=2,
↪ random_state=42)

# Dividir os dados em conjunto de treinamento e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
↪ random_state=42)

# Normalizar os dados
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Criar o modelo da rede neural
model = Sequential()
model.add(Dense(32, input_dim=X_train.shape[1], activation='relu'))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compilar o modelo
model.compile(loss='binary_crossentropy', optimizer='adam',
↪ metrics=['accuracy'])

# Treinar o modelo
model.fit(X_train, y_train, epochs=50, batch_size=10, verbose=1)

# Avaliar o modelo
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Loss: {loss}, Accuracy: {accuracy}')
```

Explicação do Código

- **Importação de Bibliotecas:** Importamos as bibliotecas necessárias, incluindo Keras, NumPy e scikit-learn.
- **Geração de Dados:** Usamos `make_classification` para criar um conjunto de dados sintético para facilitar a demonstração.
- **Divisão de Dados:** Os dados são divididos em conjuntos de treinamento e teste usando `train_test_split`.
- **Normalização:** Os dados são normalizados para melhorar o desempenho do modelo.
- **Construção do Modelo:** Usamos uma rede neural sequencial com duas camadas ocultas.
- **Compilação:** O modelo é compilado com a função de perda `binary_crossentropy` e o otimizador `adam`.
- **Treinamento:** O modelo é treinado por 50 épocas.
- **Avaliação:** O desempenho do modelo é avaliado no conjunto de teste.

Esse código é um exemplo básico de como criar uma rede neural para tarefas de classificação binária.

Referência

As ideias centrais para a construção de redes neurais podem ser extraídas do livro:

- **Título:** Neural Networks and Deep Learning
- **Capítulo:** Machine Learning with Shallow Neural Networks
- **Seção:** Neural Architectures for Binary Classification Models
- **Citação:** 2018_Book_NeuralNetworksAndDeepLearning - Capítulo 2: Machine Learning with Shallow Neural Networks - Seção: 2.2 Neural Architectures for Binary Classification Models.

Figura 4.6 – Resposta do AcademiCK - Código Python para Rede Neural Simples

Redes Neurais Convolucionais (CNNs) são um tipo de rede neural biologicamente inspirada, amplamente utilizadas em visão computacional para tarefas como classificação de imagens e detecção de objetos. Elas foram inicialmente inspiradas pelos estudos de Hubel e Wiesel sobre o córtex visual de gatos, onde se observou que porções específicas do campo visual excitam neurônios particulares.

As CNNs substituem pelo menos uma camada totalmente conectada por camadas convolucionais. A operação de convolução utiliza um filtro que realiza um mapeamento das ativações de uma camada para a próxima, preservando a estrutura espacial dos dados de entrada. Isso permite que a CNN detecte características como bordas e formas em diferentes partes da imagem, sem a necessidade de redesenhar o modelo para cada nova posição de um objeto dentro da imagem.

As CNNs geralmente são compostas por camadas convolucionais seguidas de camadas de pooling (ou subamostragem) que ajudam a reduzir a dimensionalidade das características detectadas, facilitando a generalização. Isso envolve resumir as ativações de regiões vizinhas, capturando a presença de características sem se preocupar com a sua localização exata.

Além disso, as CNNs conseguem construir uma hierarquia de conceitos, onde características mais simples em camadas iniciais se combinam para formar características mais complexas em camadas superiores. Isso resulta em uma abordagem altamente eficiente para o aprendizado de representações, contrastando com modelos tradicionais onde as características eram manuais e previamente definidas.

Em resumo, as CNNs são fundamentais para o processamento eficiente de dados visuais e têm se mostrado muito eficazes em diversos problemas, incluindo a classificação de imagens e análise de vídeo.

Figura 4.7 – Explicação sobre Redes Neurais Convolucionais (CNNs)

O pooling é uma operação que reduz as dimensões de uma imagem ou de um mapa de características, mantendo as informações mais significativas. Existem diferentes tipos de pooling, com o max pooling sendo o mais comum. Aqui está um resumo de como o pooling é realizado:

1. Definição do Filtro: Um filtro de pooling (ou kernel) é definido com um tamanho específico, como 2×2 ou 3×3 . Este filtro é aplicado em cada região da entrada.

2. Deslizamento (Stride): A operação de pooling é realizada movendo o filtro pela imagem de entrada, seguindo um passo definido chamado stride. Por exemplo, se o stride for 2, o filtro é movido duas unidades para cada passo.

3. Agregação: Para cada posição do filtro na imagem, uma função de agregação é aplicada. No max pooling, o valor máximo na área coberta pelo filtro é selecionado. Em outros tipos de pooling, como average pooling, faz-se a média dos valores.

4. Saída: O resultado é um mapa de características reduzido, onde cada valor corresponde à saída da operação de pooling de uma determinada região da entrada.

Por exemplo, ao aplicar um filtro de 2×2 com stride 2, você dividirá a imagem em blocos de 2 pixels por 2 pixels e tomará apenas o maior valor de cada bloco para o mapa de saída.

Para mais detalhes e exemplos sobre como o pooling é realizado, você pode consultar as seguintes seções dos livros:

- “Deep Learning for Computer Vision with Python” de Adrian Rosebrock - **Capítulo 11: Convolutional Neural Networks, Seção 11.2.4: Pooling Layers.**
- “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow” de Aurélien Géron - **Capítulo 14: Deep Computer Vision Using Convolutional Neural Networks, Seção: Pooling Layers.**
- “Deep Learning” de Ian Goodfellow, Yoshua Bengio e Aaron Courville - **Capítulo 8: Convolutional Neural Networks, Seção 8.2.5: Pooling.**

Essas referências fornecerão uma visão mais abrangente sobre os diferentes métodos de pooling e suas aplicações.

Figura 4.8 – Explicação sobre a operação de Pooling em CNNs

5 CONCLUSÃO

Este trabalho apresentou o desenvolvimento e avaliação de um sistema de chatbot educacional baseado em RAG com adaptação dinâmica baseada na intenção do usuário. A pesquisa buscou abordar os desafios fundamentais da aplicação de sistemas RAG no contexto educacional, propondo soluções inovadoras para melhorar a qualidade, relevância e utilidade pedagógica das respostas geradas.

5.1 Contribuições Principais

A principal contribuição desta pesquisa reside na implementação de um sistema RAG que adapta dinamicamente seu comportamento de recuperação e geração de respostas com base na intenção identificada do usuário. Esta abordagem representa um avanço significativo em relação aos sistemas RAG genéricos, que tradicionalmente aplicam estratégias uniformes independentemente do contexto ou objetivo da consulta.

O desenvolvimento de um classificador de intenções específico para o domínio educacional, treinado com um *dataset* bilíngue (português e inglês) e baseado no modelo “Snowflake/snowflake-arctic-embed-l-v2.0” (PUXUAN et al., 2024), alcançou uma acurácia de 97,6% no conjunto de teste. Este classificador demonstrou capacidade de distinguir efetivamente entre quatro categorias de intenção: busca por informações, resposta a perguntas, programação e sumarização, permitindo que o sistema adapte adequadamente suas estratégias de resposta.

A implementação de um algoritmo personalizado de processamento de documentos PDF mostrou-se eficaz na extração estruturada de conteúdo educacional. O algoritmo desenvolvido utiliza análise de tabela de conteúdos e técnicas de segmentação inteligente para preservar a estrutura hierárquica dos materiais didáticos, facilitando a recuperação precisa de informações contextualizadas.

A integração do modelo de *embeddings* BGE-M3, que combina representações densas, esparsas e ColBERT, proporcionou capacidades de recuperação mais robustas e nuançadas em comparação com abordagens tradicionais baseadas em embeddings únicos. Esta estratégia multi-modal de representação vetorial contribuiu significativamente para a qualidade da recuperação de contexto relevante.

5.2 Resultados Alcançados

A avaliação quantitativa do sistema demonstrou desempenho satisfatório em todas as métricas estabelecidas. A Fidelidade perfeita (1,0) indica que o sistema mantém consistentemente a precisão factual em relação às fontes consultadas, aspecto fundamental para aplicações educacionais. A Relevância de Resposta de 0,99 evidencia a capacidade do sistema de gerar respostas altamente alinhadas com as consultas dos estudantes.

A Relevância Contextual de 0,58, embora sendo a métrica com menor pontuação, reflete uma decisão metodológica consciente de priorizar abrangência na recuperação de informações. Esta estratégia mostrou-se pedagogicamente valiosa, permitindo que o sistema forneça não apenas respostas diretas, mas também contexto fundamental e informações adjacentes que enriquecem o processo de aprendizagem.

A análise qualitativa revelou que o sistema adapta efetivamente seu comportamento às diferentes categorias de intenção. Para consultas de busca por informações, o

sistema fornece referências estruturadas e detalhadas; para perguntas conceituais, oferece explicações didáticas com citações apropriadas; para solicitações de programação, gera código funcional com explicações técnicas; e para sumarização, apresenta sínteses abrangentes e coerentes.

A capacidade multilíngue do sistema, evidenciada pela operação eficaz em português e inglês, amplia significativamente sua aplicabilidade em contextos educacionais diversos. O sistema demonstrou consistência na qualidade das respostas independentemente do idioma da consulta, mantendo a precisão das citações e referências bibliográficas.

5.3 Implicações Práticas

Os resultados obtidos têm implicações práticas significativas para a implementação de sistemas de apoio educacional baseados em inteligência artificial. A abordagem desenvolvida oferece uma alternativa viável aos sistemas tradicionais de tutoria, proporcionando suporte personalizado e contextualizado para estudantes em diversos níveis de aprendizagem.

A capacidade do sistema de citar precisamente suas fontes e fornecer referências bibliográficas detalhadas endereça uma preocupação fundamental no uso de sistemas de IA em educação: a verificabilidade e rastreabilidade das informações fornecidas. Esta característica é particularmente relevante em contextos acadêmicos onde a integridade intelectual é prioritária.

A arquitetura modular do sistema permite sua adaptação para diferentes domínios de conhecimento através da substituição dos materiais didáticos processados, sem necessidade de modificações substanciais no código base. Esta flexibilidade facilita a implementação em diversas áreas do conhecimento e instituições educacionais.

O uso de modelos de linguagem de menor escala (GPT-4o-mini) para as operações principais do sistema, mantendo alta qualidade de resposta, tem implicações econômicas positivas para implementações em larga escala. A eficiência computacional alcançada torna o sistema viável para uso em ambientes educacionais com recursos limitados.

5.4 Limitações e Trabalhos Futuros

Apesar dos resultados promissores, este estudo apresenta algumas limitações que devem ser reconhecidas. A avaliação foi realizada utilizando métricas automatizadas com modelos de linguagem como avaliadores, o que pode não capturar completamente as nuances da utilidade educacional percebida por usuários humanos reais. Estudos futuros se beneficiariam de avaliações com estudantes em ambientes educacionais autênticos, incluindo estudos longitudinais sobre o impacto real na aprendizagem.

O conjunto de materiais didáticos utilizado para avaliação, embora diversificado, concentrou-se principalmente no domínio de aprendizado de máquina. A generalização dos resultados para outras áreas do conhecimento requer validação adicional com materiais específicos de diferentes disciplinas. Além disso, uma limitação significativa da implementação atual é sua dependência de materiais didáticos pré-processados, não possuindo capacidade para incorporar dinamicamente novos materiais em tempo real durante interações com estudantes.

A estratégia de recuperação que prioriza abrangência sobre relevância estrita, embora pedagogicamente justificada, pode não ser a melhor para todos os tipos de consultas educacionais. A atual implementação utiliza pesos predefinidos para diferentes tipos de

similaridade, sendo que uma abordagem mais sofisticada poderia implementar aprendizado de *rankings* personalizado baseado no *feedback* dos estudantes.

A granularidade da classificação de intenções implementada, embora eficaz com quatro categorias principais, pode não capturar adequadamente a diversidade completa de objetivos de aprendizagem. As categorias atuais podem não contemplar dimensões adicionais como o nível de conhecimento prévio do estudante ou objetivos pedagógicos específicos.

Finalmente, a abordagem atual para multilinguismo, embora funcional, não aborda completamente as nuances culturais e educacionais associadas a diferentes contextos linguísticos, considerando apenas diferenças linguísticas sem variações em tradições pedagógicas entre diferentes culturas.

Com base nisso, direções promissoras para pesquisas futuras incluem:

1. **Avaliação pedagógica aprofundada:** Desenvolvimento de métricas de avaliação específicas para sistemas educacionais que capturem aspectos pedagógicos além da precisão factual, incluindo estudos longitudinais em ambientes educacionais reais
2. **Personalização avançada:** Investigação de técnicas de personalização baseadas no histórico de aprendizagem individual dos estudantes e implementação de *feedback* dos usuários para aprendizagem contínua do sistema
3. **Refinamento da classificação de intenções:** Exploração de taxonomias mais refinadas de intenção, incorporando dimensões como nível de conhecimento prévio e objetivos pedagógicos específicos
4. **Capacidades multimodais:** Integração de processamento de diagramas, equações e outros elementos visuais comuns em materiais educacionais
5. **Especialização de LLM:** Treinamento de uma LLM especializada para o contexto educacional
6. **Adaptação cultural:** Desenvolvimento de adaptações que considerem não apenas diferenças linguísticas, mas também variações culturais em tradições pedagógicas e estilos de comunicação educacional

5.5 Considerações Finais

O desenvolvimento deste sistema de chatbot educacional baseado em RAG com adaptação por intenções representa um passo significativo na direção de ferramentas de apoio educacional mais inteligentes e contextualmente adequadas. Os resultados obtidos demonstram que é possível combinar as capacidades de recuperação de informações estruturadas com a flexibilidade de modelos de linguagem para criar sistemas que não apenas fornecem respostas precisas, mas que também se adaptam às necessidades específicas de diferentes contextos de aprendizagem.

A abordagem metodológica desenvolvida, que integra processamento estruturado de documentos, classificação de intenções, recuperação adaptativa e geração contextualizada, estabelece um *framework* que pode ser aplicado e adaptado para diversos cenários educacionais. A combinação de técnicas de processamento de linguagem natural, aprendizado de máquina e *design* de sistemas educacionais resulta em uma solução holística que endereça múltiplas dimensões do desafio de criar assistentes virtuais verdadeiramente úteis para educação.

Os excelentes resultados quantitativos, complementados por evidências qualitativas de adaptação efetiva às diferentes intenções dos usuários, sugerem que sistemas RAG especializados podem superar significativamente abordagens genéricas quando aplicados a domínios específicos. Esta especialização, quando combinada com estratégias adaptativas

baseadas na intenção do usuário, oferece um caminho promissor para o desenvolvimento de ferramentas educacionais mais eficazes.

Este trabalho contribui não apenas com uma implementação funcional de um sistema educacional avançado, mas também com insights metodológicos sobre como combinar diferentes tecnologias de IA para criar soluções educacionais mais eficazes. Os princípios e técnicas desenvolvidos podem ser aplicados em contextos mais amplos, apoiando o desenvolvimento de uma nova geração de ferramentas educacionais inteligentes que verdadeiramente complementam e enriquecem o processo de aprendizagem humana.

A validação empírica da efetividade da adaptação baseada em intenções abre caminho para pesquisas futuras que podem explorar formas ainda mais sofisticadas de personalização e adaptação em sistemas educacionais baseados em IA. À medida que estas tecnologias continuam evoluindo, a abordagem apresentada neste trabalho oferece uma base sólida para o desenvolvimento de sistemas educacionais cada vez mais inteligentes, úteis e pedagogicamente eficazes.

REFERÊNCIAS

- ACHIAM, J. et al. Gpt-4 technical report. **arXiv preprint arXiv:2303.08774**, 2023. Citado na página 10.
- ARTETXE, M.; SCHWENK, H. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. **Transactions of the Association for Computational Linguistics**, v. 7, p. 597–610, 2019. Citado 2 vezes nas páginas 25 e 27.
- ARTIFEX SOFTWARE, Inc. **PyMuPDF: A high performance Python library for data extraction, analysis, conversion & manipulation of PDF (and other) documents**. 2025. Python Package Index. Disponível em: <https://pypi.org/project/PyMuPDF/>. Disponível em: <https://github.com/pymupdf/PyMuPDF>. Citado na página 41.
- BAST, H.; KORZEN, C. A benchmark and evaluation for text extraction from pdf. In: **2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)**. Toronto, ON, Canada: [s.n.], 2017. p. 1–10. Citado na página 30.
- BHASKARJIT, S. et al. **How to Choose a Threshold for an Evaluation Metric for Large Language Models**. 2024. Disponível em: <https://arxiv.org/abs/2412.12148>. Citado na página 64.
- BROWN, T. et al. Language models are few-shot learners. **Advances in neural information processing systems**, v. 33, p. 1877–1901, 2020. Citado 3 vezes nas páginas 10, 15 e 17.
- CHEN, J. et al. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In: KU, L.; MARTINS, A.; SRIKUMAR, V. (Ed.). **Findings of the Association for Computational Linguistics: ACL 2024**. Bangkok, Thailand: Association for Computational Linguistics, 2024. p. 2318–2335. Disponível em: <https://aclanthology.org/2024.findings-acl.137/>. Citado 2 vezes nas páginas 26 e 51.
- DELVIN, j. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. Disponível em: <https://aclanthology.org/N19-1423/>. Citado 2 vezes nas páginas 10 e 26.
- FENG, F. et al. Language-agnostic BERT sentence embedding. In: MURESAN, S.; NAKOV, P.; VILLAVICENCIO, A. (Ed.). **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. Dublin, Ireland: Association for Computational Linguistics, 2022. p. 878–891. Disponível em: <https://aclanthology.org/2022.acl-long.62/>. Citado 2 vezes nas páginas 25 e 27.
- FENG, S. et al. Leveraging the power of large language models to drive progress in the manufacturing industry. In: **Proceedings of the 9th International Conference on Financial Innovation and Economic Development (ICFIED 2024)**. Atlantis Press, 2024. p. 125–133. ISBN 978-94-6463-408-2. ISSN 2352-5428. Disponível em: https://doi.org/10.2991/978-94-6463-408-2_15. Citado 2 vezes nas páginas 11 e 19.

GAO, Y. et al. Retrieval-augmented generation for large language models: A survey. **arXiv preprint arXiv:2312.10997**, v. 2, n. 1, 2023. Citado 2 vezes nas páginas 20 e 21.

GIDIOTIS, A.; TSOUMAKAS, G. A divide-and-conquer approach to the summarization of long documents. **IEEE/ACM Transactions on Audio, Speech, and Language Processing**, v. 28, p. 3029–3040, 2020. Citado na página 32.

GUO, D. et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. **arXiv preprint arXiv:2501.12948**, 2025. Citado 2 vezes nas páginas 10 e 18.

GÉRON, A. **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. 2. ed. Sebastopol, CA: O'Reilly Media, Inc., 2019. 600 p. ISBN 978-1-492-03264-9. Disponível em: <<https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/>>. Citado na página 42.

HAN, J. et al. LLM-as-a-tutor in EFL writing education: Focusing on evaluation of student-LLM interaction. In: KUMAR, S. et al. (Ed.). **Proceedings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U)**. Miami, Florida, USA: Association for Computational Linguistics, 2024. p. 284–293. Disponível em: <<https://aclanthology.org/2024.customnlp4u-1.21/>>. Citado 2 vezes nas páginas 12 e 36.

HAQUE, M. A.; LI, S. Exploring chatgpt and its impact on society. **AI and Ethics**, Springer, p. 1–13, 2024. Citado na página 10.

HUANG, Y. et al. Advancing transformer architecture in long-context large language models: A comprehensive survey. **arXiv preprint arXiv:2311.12351**, 2023. Citado na página 10.

HWANG, G. et al. Vision, challenges, roles and research issues of artificial intelligence in education. **Computers and Education: Artificial Intelligence**, v. 1, p. 100001, 2020. Citado na página 12.

IP, J.; AI, C. **DeepEval: The Open-Source LLM Evaluation Framework**. 2024. Python package version 2.9.3. Disponível em: <<https://github.com/confident-ai/deepeval>>. Citado 2 vezes nas páginas 22 e 63.

JI, Z. et al. Survey of hallucination in natural language generation. **ACM computing surveys**, ACM New York, NY, v. 55, n. 12, p. 1–38, 2023. Citado na página 11.

JOHNSON, J.; DOUZE, M.; JÉGOU, H. Billion-scale similarity search with gpus. **IEEE Transactions on Big Data**, IEEE, v. 7, n. 3, p. 535–547, 2019. Citado na página 56.

JUMAR, H. et al. Guiding students in using llms in supported learning environments: Effects on interaction dynamics, learner performance, confidence, and trust. **Proc. ACM Hum.-Comput. Interact.**, Association for Computing Machinery, New York, NY, USA, v. 8, n. CSCW2, nov. 2024. Disponível em: <<https://doi.org/10.1145/3687038>>. Citado 2 vezes nas páginas 12 e 36.

KARPUKHIN, V. et al. Dense passage retrieval for open-domain question answering. In: WEBBER, B. et al. (Ed.). **Proceedings of the 2020 Conference on Empirical Methods in**

Natural Language Processing (EMNLP). Online: Association for Computational Linguistics, 2020. p. 6769–6781. Disponível em: <<https://aclanthology.org/2020.emnlp-main.550/>>. Citado na página 28.

LAI, J. et al. Large language models in law: A survey. **AI Open**, Elsevier, 2024. Citado 2 vezes nas páginas 12 e 19.

LEWIS, P. et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In: **Proceedings of the 34th International Conference on Neural Information Processing Systems**. Red Hook, NY, USA: Curran Associates Inc., 2020. (NIPS '20). ISBN 9781713829546. Citado 3 vezes nas páginas 12, 13 e 20.

LI, Y. et al. Large language models in finance: A survey. In: **Proceedings of the Fourth ACM International Conference on AI in Finance**. New York, NY, USA: Association for Computing Machinery, 2023. (ICAIF '23), p. 374–382. ISBN 9798400702402. Disponível em: <<https://doi.org/10.1145/3604237.3626869>>. Citado 2 vezes nas páginas 11 e 19.

LI, Z. et al. Synthetic data generation with large language models for text classification: Potential and limitations. In: BOUAMOR, H.; PINO, J.; BALI, K. (Ed.). **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**. Singapore: Association for Computational Linguistics, 2023. p. 10443–10461. Disponível em: <<https://aclanthology.org/2023.emnlp-main.647/>>. Citado na página 47.

LIAO, Z. et al. Llms as research tools: A large scale survey of researchers' usage and perceptions. **arXiv preprint arXiv:2411.05025**, 2024. Citado 3 vezes nas páginas 11, 19 e 40.

LIU, P. et al. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 55, n. 9, jan. 2023. ISSN 0360-0300. Disponível em: <<https://doi.org/10.1145/3560815>>. Citado na página 36.

LIU, Z. et al. Evaluating the knowledge dependency of questions in conversational question answering. **arXiv preprint arXiv:2308.13710**, 2023. Citado na página 23.

MAHAMULKAR, P. **Machine-Learning-QA-dataset**. Hugging Face, 2024. Hugging Face Datasets. A collection of machine learning questions and answers for educational and training purposes. Disponível em: <<https://huggingface.co/datasets/prsdm/Machine-Learning-QA-dataset>>. Citado na página 63.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: BURGESS, C.; BOTTOU, L.; WELLING, M.; GHAMRANI, Z.; WEINBERGER, K. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2013. v. 26. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf>. Citado na página 25.

MINAEE, S. et al. Large language models: A survey. **arXiv preprint arXiv:2402.06196**, 2024. Citado 2 vezes nas páginas 10 e 15.

NI, J. et al. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In: MURESAN, S.; NAKOV, P.; VILLAVICENCIO, A. (Ed.). **Findings of the Association**

for **Computational Linguistics: ACL 2022**. Dublin, Ireland: Association for Computational Linguistics, 2022. p. 1864–1874. Disponível em: <<https://aclanthology.org/2022.findings-acl.146/>>. Citado na página 26.

OPENAI. **Learning to reason with LLMs**. 2024. <<https://openai.com/index/learning-to-reason-with-llms/>>. Acesso em: 12 fev. 2025. Citado 2 vezes nas páginas 10 e 18.

OUYANG, L. et al. Training language models to follow instructions with human feedback. **Advances in neural information processing systems**, v. 35, p. 27730–27744, 2022. Citado na página 18.

PUXUAN, Y. et al. **Arctic-Embed 2.0: Multilingual Retrieval Without Compromise**. 2024. Disponível em: <<https://arxiv.org/abs/2412.04506>>. Citado 2 vezes nas páginas 47 e 75.

R., S. **ChatGPT Architecture Explained**. 2023. <<https://medium.com/@sreedevr/chatgpt-architecture-explained-7fc0cbb7426c>>. Medium article. Citado na página 17.

RAMÍREZ, S. **FastAPI: modern, fast (high-performance), web framework for building APIs with Python**. 2018. <<https://github.com/fastapi/fastapi>>. Acessado em: Julho 2025. Citado na página 56.

REIMERS, N.; GUREVYCH, I. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: INUI, K. et al. (Ed.). **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**. Hong Kong, China: Association for Computational Linguistics, 2019. p. 3982–3992. Disponível em: <<https://aclanthology.org/D19-1410/>>. Citado 2 vezes nas páginas 24 e 26.

SHEN, L. et al. The language barrier: Dissecting safety challenges of LLMs in multilingual contexts. In: KU, L.; MARTINS, A.; SRIKUMAR, V. (Ed.). **Findings of the Association for Computational Linguistics: ACL 2024**. Bangkok, Thailand: Association for Computational Linguistics, 2024. p. 2668–2680. Disponível em: <<https://aclanthology.org/2024.findings-acl.156/>>. Citado 4 vezes nas páginas 12, 19, 36 e 39.

SOLOMIN, A.; IVANOVA, Y. Modern approaches to multiclass intent classification based on pre-trained transformers. **Journal Scientific and Technical Of Information Technologies, Mechanics and Optics**, v. 128, n. 4, p. 532–538, 2020. Citado 3 vezes nas páginas 13, 34 e 35.

VASWANI, A. et al. Attention is all you need. In: GUYON, I. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2017. v. 30, p. 5998–6008. Disponível em: <<https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>>. Citado 3 vezes nas páginas 10, 15 e 16.

WANG, J. et al. Milvus: A purpose-built vector data management system. In: **Proceedings of the 2021 International Conference on Management of Data**. New York, NY, USA: Association for Computing Machinery, 2021. (SIGMOD '21), p. 2614–2627. Citado na página 56.

WEI, J. et al. Chain-of-thought prompting elicits reasoning in large language models. In: KOYEJO, S.; MOHAMED, S.; AGARWAL, A.; BELGRAVE, D.; CHO, K.; OH, A. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2022. v. 35, p. 24824–24837. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf>. Citado na página 38.

WOLLNY, S. et al. Are we there yet? a systematic literature review on chatbots in education. **Frontiers in Artificial Intelligence**, v. 4, p. 654924, 2021. Citado na página 35.

XIONG, L.; XIONG, C.; LI, Y.; TANG, K.-F.; LIU, J.; BENNETT, P. N.; AHMED, J.; OVERWIJK, A. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: **9th International Conference on Learning Representations**. Virtual Event, Austria: OpenReview.net, 2021. Disponível em: <<https://openreview.net/forum?id=zeFrfgYzln>>. Citado na página 28.

XU, Y. et al. Layoutlm: Pre-training of text and layout for document image understanding. In: **Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. New York, NY, USA: Association for Computing Machinery, 2020. (KDD '20), p. 1192–1200. ISBN 9781450379984. Disponível em: <<https://doi.org/10.1145/3394486.3403172>>. Citado na página 31.

XU, Y. et al. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In: ZONG, C. et al. (Ed.). **Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. Online: Association for Computational Linguistics, 2021. p. 2579–2591. Disponível em: <<https://aclanthology.org/2021.acl-long.201/>>. Citado na página 31.

YANG, R. et al. Large language models in health care: Development, applications, and challenges. **Health Care Science**, Wiley Online Library, v. 2, n. 4, p. 255–263, 2023. Citado 2 vezes nas páginas 11 e 19.

YU, C.; ZHANG, C.; WANG, J. Extracting body text from academic PDF documents for text mining. **CoRR**, abs/2010.12647, 2020. Disponível em: <<https://arxiv.org/abs/2010.12647>>. Citado na página 41.