

UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Curso de Bacharelado em Engenharia de Computação



Trabalho de Conclusão de Curso

**VideoLearnAI: LLM Powered Web Application para aprendizagem ativa com
vídeos do Youtube**

Kevin Castro Weitgenant

Pelotas, 2025

Kevin Castro Weitgenant

**VideoLearnAI: LLM Powered Web Application para aprendizagem ativa com
vídeos do Youtube**

Trabalho de Conclusão de Curso apresentado
ao Centro de Desenvolvimento Tecnológico da
Universidade Federal de Pelotas, como requisito
parcial à obtenção do título de Bacharel em En-
genharia de Computação.

Orientador: Prof. Dr. Tiago Primo
Coorientador: Prof. Dr. Marilton Sanchotene de Aguiar

Pelotas, 2025

**Insira AQUI a ficha catalográfica
Quando finalizado o trabalho, deve ser
solicitada através do Sistema Cobalto
Biblioteca – Cadastro – Ficha catalográfica.**

Kevin Castro Weitgenant

**VideoLearnAI: LLM Powered Web Application para aprendizagem ativa com
vídeos do Youtube**

Trabalho de Conclusão de Curso aprovado, como requisito parcial, para obtenção do grau de Bacharel em Engenharia de Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 16 de março de 2025

Banca Examinadora:

Prof. Dr. Marilton Sanchotene de Aguiar (orientador)
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Paulo Roberto Ferreira Jr.
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Ricardo Matsumura Araujo
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Luciano da Silva Pinto
Doutor em Biotecnologia pela Universidade Federal de Pelotas.

Dedico...

AGRADECIMENTOS

Agradeço...

What would life be if we had no courage to attempt anything?
— VINCENT VAN GOGH

RESUMO

WEITGENANT, Kevin Castro. **VideoLearnAI: LLM Powered Web Application para aprendizagem ativa com vídeos do Youtube.** Orientador: Tiago Primo. 2025. 78 f. Trabalho de Conclusão de Curso (Engenharia de Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2025.

Este trabalho apresenta o desenvolvimento de um Software como Serviço (SaaS) potencializado por Large Language Models (LLMs) para transformar o consumo passivo de vídeos educacionais em experiências de aprendizagem ativa. Apesar da abundância de conteúdo educacional de alta qualidade disponível online, em geral, as pessoas assistem a esses vídeos de forma passiva, o que limita a compreensão e a retenção do conhecimento, uma vez que a aprendizagem mais efetiva ocorre através do engajamento ativo com o material. Para resolver este problema, o sistema VideoLearnAI implementa cinco funcionalidades principais: melhoria automática da legibilidade de legendas utilizando o modelo SaT (Segment Any Text), geração de capítulos para navegação estruturada, transcrição sincronizada de alta qualidade, quizzes interativos personalizáveis e um sistema de bate-papo contextual baseado no conteúdo do vídeo. A implementação focou em escalabilidade e desempenho, utilizando processamento GPU e técnicas modernas de desenvolvimento de software. Os resultados demonstram o potencial da solução para tornar vídeos educacionais mais engajantes e eficazes, criando experiências de aprendizagem personalizadas e interativas que promovem maior retenção de conhecimento e compreensão mais profunda do conteúdo.

Palavras-chave: aprendizagem ativa; large language models; processamento de linguagem natural; educação online.

ABSTRACT

WEITGENANT, Kevin Castro. **VideoLearnAI: LLM-Powered Web Application for Active Learning with YouTube Videos.** Orientador: Tiago Primo. 2025. 78 f. Trabalho de Conclusão de Curso (Engenharia de Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2025.

This work presents the development of an LLM-powered Software as a Service (SaaS) that transforms passive video consumption into active learning experiences. Despite the abundance of high-quality educational content available online, people generally watch these videos passively, which limits comprehension and knowledge retention, as more effective learning occurs through active engagement with the material. To address this challenge, the VideoLearnAI system implements five main functionalities: automatic subtitle readability improvement using the SaT (Segment Any Text) model, chapter generation for structured navigation, high-quality synchronized transcription, customizable interactive quizzes, and a contextual chat system based on video content. The implementation focused on scalability and performance, utilizing GPU processing and modern software development techniques. The results demonstrate the solution's potential to make educational videos more engaging and effective, creating personalized and interactive learning experiences that promote greater knowledge retention and deeper understanding of content.

Keywords: active learning; large language models; natural language processing; online education.

LISTA DE FIGURAS

Figura 1	Diagrama de arquitetura do sistema VideoLearnAI	48
Figura 2	Interface para seleção do modo de leitura da transcrição	53
Figura 3	Comparação da legibilidade antes e depois do processamento com o modelo SAT	54
Figura 4	Logs de execução no Beam Serverless GPU demonstrando o tempo de cold start (18s 573ms) na primeira requisição e tempos de resposta reduzidos (aproximadamente 50-100ms) nas requisições subsequentes	54
Figura 5	Interface para iniciar o processo de geração automática de capítulos, mostrando o botão "Generate Chapters" que aciona o processamento	56
Figura 6	Interface após a geração de capítulos, os capítulos são listados no contexto do Chat e também na tabela de conteúdos que será apresentada na próxima seção	57
Figura 7	Interface do Sumário de Conteúdo, mostrando a lista de capítulos com indicadores de progresso e opções para marcar capítulos como concluídos	58
Figura 8	Tela de Histórico de Visualização, exibindo os vídeos assistidos pelo usuário com seus respectivos percentuais de progresso	59
Figura 9	Botão "CC" para iniciar o processo de transcrição	60
Figura 10	Modal de confirmação mostrando créditos disponíveis e duração do vídeo	61
Figura 11	Etapa de validação da duração do vídeo via API do YouTube	62
Figura 12	Progresso do download do áudio do vídeo com atualização percentual	62
Figura 13	Processo de transcrição em andamento com estimativa de tempo restante	63
Figura 14	Tela de conclusão do processo de transcrição	63
Figura 15	Exemplo de interface para perguntas e respostas	65
Figura 16	Exemplo de interface para questões de verdadeiro ou falso	66
Figura 17	Botão ao final de cada capítulo, oferecendo diferentes opções de interação com o conteúdo	67
Figura 18	Botão de três pontos para acesso a customização do prompt, nesse caso específico para a opção de geração de questões de Verdadeiro ou Falso	67
Figura 19	Popover para customização do prompt utilizado na geração de questões	68
Figura 20	Questões de Verdadeiro ou Falso geradas em espanhol após customização do prompt	68

Figura 21 Interface do componente de bate-papo com vídeo, mostrando painel de seleção de contexto e área de conversação 69

LISTA DE TABELAS

Tabela 1 Comparativo entre YouLearn.ai, StudyFetch e VideoLearnAI 23

LISTA DE ABREVIATURAS E SIGLAS

SaaS	Software as a Service
API	Application Programming Interface
ASR	Automatic Speech Recognition
GPU	Graphics Processing Unit
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
JWT	JSON Web Token
LLM	Large Language Model
ORM	Object-Relational Mapping
RAG	Retrieval-Augmented Generation
REST	Representational State Transfer
SaT	Segment Any Text
SSE	Server-Sent Events
UI	User Interface
UX	User Experience

SUMÁRIO

1 INTRODUÇÃO	17
1.1 Objetivo Geral	18
1.2 Objetivos Específicos	18
1.3 Estrutura do Trabalho	19
2 SOLUÇÕES RELACIONADAS	20
2.1 Ferramentas Educacionais Potencializadas por IA	20
2.1.1 YouLearn.ai	20
2.1.2 StudyFetch	21
2.2 VideoLearnAI: Uma Abordagem Especializada	21
2.3 Comparativo entre as Soluções	22
3 FUNDAMENTAÇÃO TEÓRICA	24
3.1 Aprendizagem ativa	24
3.1.1 Fundamentos Teóricos	24
3.1.2 Princípios Cognitivos	24
3.1.3 Desafios da Aprendizagem com Vídeos	25
3.1.4 VideoLearnAI como Ferramenta de Aprendizagem Ativa	25
3.2 Large Language Models (LLMs)	26
3.2.1 Conceitos Fundamentais	27
3.2.2 Arquitetura e Funcionamento	28
3.2.3 Prompt Engineering	29
3.2.4 Limitações	30
3.3 LLM Powered Applications	33
3.3.1 Arquiteturas de Aplicações com LLMs	33
3.3.2 Function Calling	34
3.3.3 Agentes Autônomos Baseados em LLMs	35
3.3.4 Retrieval-Augmented Generation (RAG) e RAG Agêntico	36
3.4 Conclusão do capítulo	38
4 TECNOLOGIAS UTILIZADAS	40
4.1 Frameworks e Bibliotecas	40
4.1.1 FastAPI	40
4.1.2 Next.js	40
4.1.3 Vercel AI SDK	41
4.1.4 Drizzle ORM	41
4.1.5 ShadCN	42
4.1.6 Zustand	42

4.1.7	Stripe	42
4.2	Modelos de Machine Learning	43
4.2.1	API Deepgram	43
4.2.2	API GPT	43
4.2.3	Segment Anything (SaT)	43
4.3	Infraestrutura e Serviços em Nuvem	44
4.3.1	Open Next.js	44
4.3.2	Google Cloud	44
4.3.3	Beam Serverless GPU	44
4.3.4	Supabase	45
4.3.5	Sentry	45
4.4	Ferramentas de Desenvolvimento	45
4.4.1	Cursor	45
4.4.2	v0.dev	46
4.5	Conclusão do capítulo	46
5	DESENVOLVIMENTO DA APLICAÇÃO	48
5.1	Visão Geral da Arquitetura	48
5.1.1	Componentes Principais	48
5.1.2	Interações Externas	49
5.1.3	Fluxos de Comunicação	50
5.2	Melhoria da Legibilidade das Legendas	50
5.2.1	O Problema da Legibilidade	51
5.2.2	Primeira Abordagem com LLMs	51
5.2.3	Implementação com SaT (Segment Anything Text)	52
5.2.4	Interface e Experiência do Usuário	53
5.3	Geração de Capítulos	54
5.3.1	Arquitetura do Sistema de Geração de Capítulos	55
5.3.2	Fluxo de Processamento	55
5.3.3	Integração com LLMs via Function Calling	56
5.3.4	Tabela de Conteúdos e Histórico de Visualização	57
5.4	Geração de Transcrição	59
5.4.1	Primeira implementação com Whisper	60
5.4.2	Migração para o Deepgram	60
5.4.3	Interface e Experiência do Usuário	60
5.5	Geração de Quizzes	63
5.5.1	Questões discursivas	65
5.5.2	Questões de Verdadeiro ou Falso	66
5.5.3	Geração e Personalização de Quizzes	66
5.6	Bate-Papo com Vídeo	69
5.6.1	Interface do Usuário	69
5.6.2	Gerenciamento de Contexto	70
5.6.3	Benefícios e Características	70
5.7	Discussão	71
6	CONCLUSÃO	72
6.1	Objetivos Alcançados	72
6.2	Trabalhos Futuros	73

1 INTRODUÇÃO

Nos últimos anos, o consumo de conteúdo educacional em vídeo tem crescido exponencialmente, impulsionado por plataformas como YouTube¹, Coursera² e Udemy³. Hoje, é possível encontrar aulas completas de universidades de altíssimo nível, como MIT, Harvard e Stanford, gratuitamente disponíveis online. No entanto, apesar da abundância de material de qualidade, muitos usuários enfrentam dificuldades em absorver e reter conhecimento de forma eficiente. A maioria das pessoas consome esses conteúdos de maneira passiva, apenas assistindo aos vídeos sem um envolvimento ativo com o material. Isso limita a retenção e a compreensão das informações (Guo; Kim; Rubin, 2014).

A aprendizagem ativa, por outro lado, é um modelo comprovadamente mais eficaz, pois envolve o estudante em processos como resumo, questionamento, reorganização do conteúdo e interação com o material. Pesquisas mostram que métodos ativos de estudo, como fazer perguntas sobre o conteúdo, testar-se frequentemente e organizar a informação de forma estruturada, levam a um aprendizado mais profundo e duradouro (Freeman; Eddy; McDonough; Smith; Okoroafor; Jordt; Wenderoth, 2014).

Diante desse cenário, este trabalho apresenta o desenvolvimento de um Software as a Service (SaaS) voltado para transformar o consumo passivo de vídeos educacionais em um processo de aprendizagem ativa. A solução emprega diferentes arquiteturas de modelos de IA, incluindo Large Language Models (LLMs) e modelos especializados em processamento de texto, para reestruturar legendas em formatos mais legíveis, gerar capítulos automaticamente, fornecer resumos e permitir interações como perguntas e respostas sobre o conteúdo. Além disso, o sistema oferece quizzes dinâmicos para reforçar o aprendizado e um mecanismo para salvar o progresso dos usuários, incentivando um envolvimento mais estruturado com os vídeos (Schwan; Riempp, 2004).

O desenvolvimento da plataforma seguiu uma abordagem iterativa, priorizando a implementação incremental das funcionalidades e a adaptação contínua da arquite-

¹<https://www.youtube.com>

²<https://www.coursera.org>

³<https://www.udemy.com>

tura. A solução técnica integra um frontend baseado em Next.js com dois backends complementares: um principal também em Next.js (utilizando API Routes e Server Components) e um secundário em FastAPI para operações específicas que exigem processamento em Python, como a execução do modelo SaT (Frohmann; Sterner; Vulic; Minixhofer; Schedl, 2024) e extração de dados do YouTube.

Esta arquitetura híbrida permite aproveitar as vantagens de cada tecnologia: a renderização otimizada e o desenvolvimento tipo-seguro do ecossistema React/Next.js, combinados com a eficiência do Python para processamento de linguagem natural e manipulação de modelos de machine learning. Para o serviço de transcrição automática de áudio (ASR), foi utilizado o Deepgram, que oferece alta precisão e suporte a múltiplos idiomas. O armazenamento e gerenciamento dos dados foi implementado utilizando o Supabase, uma plataforma de banco de dados moderna e escalável.

Para garantir eficiência e escalabilidade, foram implementadas estratégias de otimização como serverless GPU para modelos que exigem aceleração por hardware. Adicionalmente, técnicas de streaming de respostas foram utilizadas para melhorar a experiência do usuário, permitindo visualização progressiva de conteúdos gerados por IA sem necessidade de aguardar o processamento completo.

Este projeto experimental revelou potenciais significativos na convergência entre tecnologias de Inteligência Artificial e metodologias de estudo ativo. A plataforma desenvolvida, embora em estágio inicial de implementação, demonstra resultados promissores em sua proposta de potencializar a experiência de aprendizagem dos estudantes durante a utilização de conteúdos audiovisuais educacionais.

1.1 Objetivo Geral

O objetivo geral deste trabalho é desenvolver um Software as a Service (SaaS) que transforme o consumo passivo de vídeos educacionais em um processo de aprendizagem ativa. Para isso, a plataforma utilizará inteligência artificial para melhorar a legibilidade das legendas, gerar resumos, estruturar conteúdos em capítulos, criar quizzes interativos e permitir interações diretas com o conteúdo por meio de perguntas e respostas. O foco é tornar o aprendizado com vídeos mais eficiente, estruturado e acessível, permitindo aproveitar melhor o vasto acervo educacional disponível online.

1.2 Objetivos Específicos

Para atingir o objetivo geral, este trabalho busca:

- Desenvolver um sistema que reestruture legendas de vídeos em textos mais legíveis e organizados, facilitando a compreensão.

- Implementar um mecanismo para geração automática de capítulos e resumos, permitindo uma navegação mais eficiente pelo conteúdo.
- Gerar legendas de alta qualidade utilizando modelos avançados de reconhecimento automático de fala (ASR), superando as limitações das legendas geradas automaticamente pelo YouTube.
- Criar um chatbot contextual que possibilite interações e perguntas diretamente relacionadas ao conteúdo do vídeo.
- Desenvolver um sistema de quizzes automáticos baseados no conteúdo dos vídeos, reforçando o aprendizado ativo.
- Implementar um sistema de salvamento de progresso para permitir que usuários retomem facilmente seus estudos.

1.3 Estrutura do Trabalho

Este trabalho está organizado da seguinte forma:

- **Capítulo 2 – Soluções Relacionadas:** apresenta algumas ferramentas educacionais baseadas em IA com propósitos similares, analisando suas abordagens e funcionalidades, e contextualizando a contribuição específica da solução desenvolvida.
- **Capítulo 3 – Fundamentação Teórica:** explora os conceitos fundamentais de aprendizagem ativa, Large Language Models (LLMs) e aplicações potencializadas por LLMs, que embasam o desenvolvimento da solução proposta.
- **Capítulo 4 – Tecnologias utilizadas:** detalha os frameworks, bibliotecas, modelos de machine learning, infraestrutura e ferramentas de desenvolvimento empregados na construção da aplicação.
- **Capítulo 5 – Desenvolvimento da Aplicação:** apresenta a visão geral da arquitetura e detalha a implementação das principais funcionalidades do sistema, incluindo melhoria de legibilidade das legendas, geração de capítulos, transcrição, geração de quizzes e bate-papo com vídeo, além dos desafios enfrentados e soluções adotadas.
- **Capítulo 6 – Conclusão:** discute os objetivos alcançados, as contribuições do trabalho e apresenta perspectivas para trabalhos futuros.

2 SOLUÇÕES RELACIONADAS

Esta seção apresenta algumas ferramentas educacionais baseadas em IA que compartilham objetivos similares aos do VideoLearnAI. O foco está em duas plataformas específicas - YouLearn.ai e StudyFetch - que ilustram abordagens alternativas para o uso de inteligência artificial no aprimoramento da experiência de aprendizagem.

2.1 Ferramentas Educacionais Potencializadas por IA

O avanço das tecnologias de inteligência artificial, especialmente dos modelos de linguagem de grande escala (LLMs), tem possibilitado o surgimento de novas abordagens para o aprendizado digital. Diversas ferramentas têm sido desenvolvidas com o objetivo de enriquecer a experiência educacional, tornando-a mais interativa, personalizada e eficaz.

2.1.1 YouLearn.ai

O YouLearn.ai posiciona-se como um tutor de IA personalizado, projetado para transformar diversos formatos de conteúdo educacional em experiências de aprendizado interativas. Esta plataforma se destaca pelos seguintes recursos:

- **Processamento multimodal:** Capacidade de processar e compreender diversos formatos de conteúdo (PDFs, vídeos, apresentações de slides), permitindo que os usuários interajam com materiais educacionais através de uma interface inteligente.
- **Supporte multilíngue:** Funcionalidade de traduzir e explicar conceitos complexos no idioma nativo do usuário, tornando o aprendizado mais acessível para falantes não nativos de inglês e removendo barreiras linguísticas.
- **Assistente de IA conversacional:** Sistema de chat integrado que permite aos usuários fazer perguntas sobre o material estudado, com indicação das fontes utilizadas para aumentar a credibilidade e transparência das respostas.

- **Espaços de aprendizado personalizáveis:** Ferramentas para organizar e categorizar materiais de estudo com base em disciplinas, cursos ou preferências pessoais, criando um ambiente de aprendizado mais coerente e gerenciável.

2.1.2 StudyFetch

O StudyFetch apresenta-se como um ecossistema completo de ferramentas de aprendizado potencializadas por IA, oferecendo diversos recursos educacionais:

- **Geração automática de conteúdo:** Conjunto abrangente de ferramentas para criação de materiais de estudo, incluindo Notes AI (geração de anotações concisas), Flashcards AI (criação de cartões de memorização) e Quizzes AI (desenvolvimento de questões práticas adaptadas ao material específico).
- **Spark.E - Tutor de IA avançado:** Assistente virtual que oferece suporte em tempo real durante as sessões de estudo, com capacidade de discutir materiais, responder perguntas e fornecer assistência imediata 24/7. Inclui recursos como "Call with Spark.E"(interações por chamada de voz) e "Tutor Me"(aulas completas lideradas por IA).
- **Ferramentas multimídia:** Recursos como Record Live Lecture (captura de aulas ao vivo com reconhecimento de voz), Audio Recap (geração de podcasts e resumos em áudio) e Explainer Video (criação de vídeos educacionais usando IA).
- **Recursos para educadores:** Funcionalidades específicas para professores, como Teacher Assistant (criação de assistentes de ensino personalizados) e Analytics (análise de tópicos que os estudantes estão tendo dificuldade).

2.2 VideoLearnAI: Uma Abordagem Especializada

A VideoLearnAI representa uma proposta diferenciada no cenário de ferramentas educacionais potencializadas por IA. Enquanto plataformas como YouLearn.ai e StudyFetch oferecem ecossistemas abrangentes e versáteis para diversos formatos de conteúdo, a VideoLearnAI adota uma estratégia de especialização, concentrando-se exclusivamente na experiência com vídeos.

As plataformas YouLearn.ai e StudyFetch destacam-se pela amplitude de suas funcionalidades, com processamento multimodal, suporte multilíngue e ferramentas avançadas para educadores. A VideoLearnAI, em contraste, diferencia-se pela granularidade do processamento de conteúdo audiovisual, permitindo interações seletivas e personalizadas com segmentos específicos do vídeo, evitando a sobrecarga cognitiva associada à geração indiscriminada de centenas de quizzes e flashcards quando todo o conteúdo do vídeo é processado de uma só vez.

Esta especialização permite o desenvolvimento de funcionalidades integradas à experiência audiovisual, com segmentação precisa do conteúdo em capítulos significativos. O vídeo permanece como elemento central, enquanto ferramentas complementares transformam o consumo passivo em aprendizagem ativa.

A plataforma implementa uma filosofia de design que prioriza:

- **Processamento granular:** Trabalho com segmentos específicos do vídeo, evitando sobrecarga cognitiva.
- **Integração audiovisual:** Sincronização entre player e transcrição com navegação bidirecional.
- **Personalização via prompts:** Controle do usuário sobre o tipo e dificuldade das questões geradas.
- **Estruturação de texto:** Transformação de legendas em texto coeso e legível.
- **Transcrição de alta fidelidade:** Geração de transcrições precisas com modelos avançados.
- **Progresso contextualizado:** Rastreamento de progresso baseado em capítulos.

A interface equilibra automação e controle manual, oferecendo funcionalidades sofisticadas sem complexidade desnecessária. Esta abordagem especializada resulta em uma ferramenta que, embora mais focada em escopo, proporciona uma experiência de aprendizado eficaz para conteúdo audiovisual, complementando as plataformas mais abrangentes.

2.3 Comparativo entre as Soluções

A Tabela 1 apresenta um comparativo entre as três soluções analisadas, destacando suas principais características, abordagens e diferenciais.

Tabela 1 – Comparativo entre YouLearn.ai, StudyFetch e VideoLearnAI

Característica	YouLearn.ai	StudyFetch	VideoLearnAI
Tipos de conteúdo	Múltiplos (PDFs, vídeos, slides)	Múltiplos (textos, áudios, vídeos)	Exclusivamente vídeos
Processamento de conteúdo	Abrangente com suporte multimodal	Geração automática de diversos materiais	Granular com foco em segmentos específicos
Recursos de acessibilidade	Suporte multilíngue integrado	Áudio Recap e recursos de conversão de formato	Melhoria de legibilidade de legendas e transcrição de alta qualidade
Ferramentas de avaliação	Quizzes gerais sobre o material	Flashcards AI e Quizzes AI abrangentes	Quizzes personalizáveis por capítulo com prompts editáveis
Integração audiovisual	Básica	Recursos como Record Live Lecture e Explainer Video	Sincronização bidirecional e navegação por capítulos
Diferencial principal	Versatilidade multimodal e multilíngue	Amplitude de ferramentas e recursos para educadores	Especialização em vídeos com processamento granular e possibilidade de customização dos prompts

Como evidenciado na tabela comparativa, enquanto YouLearn.ai e StudyFetch oferecem ecossistemas abrangentes para diversos formatos de conteúdo, o VideoLearnAI adota uma abordagem de especialização focada exclusivamente na experiência com vídeos. Esta especialização permite um nível de refinamento nas funcionalidades específicas para conteúdo audiovisual que seria difícil de alcançar em plataformas mais generalistas. Cada solução apresenta vantagens distintas dependendo do contexto de uso e das necessidades específicas do usuário.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Aprendizagem ativa

A aprendizagem ativa representa um paradigma educacional que contrasta significativamente com métodos tradicionais de ensino passivo. Enquanto abordagens passivas posicionam o estudante como mero receptor de informações, a aprendizagem ativa coloca-o como protagonista do processo educacional, exigindo engajamento cognitivo direto e participação constante (Freeman; Eddy; Mcdonough; Smith; Okoroafor; Jordt; Wenderoth, 2014). Este modelo pedagógico fundamenta-se na premissa de que o conhecimento é construídoativamente pelo aprendiz, não simplesmente transmitido pelo educador ou material didático.

3.1.1 Fundamentos Teóricos

Bonwell e Eison (Bonwell; Eison, 1991) definiram aprendizagem ativa como "qualquer coisa que envolva os estudantes em fazer coisas e pensar sobre as coisas que estão fazendo". Esta definição enfatiza dois componentes essenciais: a ação (fazer) e a reflexão sobre essa ação (metacognição).

A eficácia da aprendizagem ativa tem sido amplamente documentada na literatura científica. Uma meta-análise abrangente conduzida por Freeman et al. (Freeman; Eddy; Mcdonough; Smith; Okoroafor; Jordt; Wenderoth, 2014) examinou 225 estudos comparando métodos ativos com aulas expositivas tradicionais. Os resultados foram contundentes: a aprendizagem ativa aumentou as notas dos alunos em média 6% e reduziu as taxas de reprovação em aproximadamente 1,5 vezes em comparação com métodos passivos.

3.1.2 Princípios Cognitivos

Do ponto de vista cognitivo, a aprendizagem ativa fundamenta-se em princípios bem estabelecidos de como o cérebro processa e retém informações. Pesquisas na área identificaram várias técnicas de estudo baseadas em evidências que promovem aprendizagem eficaz (Dunlosky; Rawson; Marsh; Nathan; Willingham, 2013):

- **Prática de recuperação:** O ato de recuperar informações da memória fortalece as conexões neurais associadas a esse conhecimento.
- **Prática distribuída:** Distribuir o estudo ao longo do tempo melhora significativamente a retenção de longo prazo.
- **Aprendizagem intercalada:** Alternar entre diferentes tipos de problemas ou tópicos durante o estudo promove maior transferência de conhecimento.
- **Elaboração e autoexplicação:** Explicar conceitos e conectá-los ao conhecimento existente cria redes neurais mais robustas.

Estes princípios são particularmente relevantes no contexto do consumo de conteúdo em vídeo, que tradicionalmente tende a ser uma experiência passiva.

3.1.3 Desafios da Aprendizagem com Vídeos

O consumo de vídeos educacionais, embora conveniente, apresenta desafios específicos para a aprendizagem efetiva. Estudos mostram que a atenção dos estudantes diminui drasticamente após 6-9 minutos de vídeo (Guo; Kim; Rubin, 2014). Além disso, existe o fenômeno da "ilusão de compreensão- a tendência dos estudantes de superestimar seu entendimento após assistir a um vídeo explicativo (Schwan; Riempp, 2004).

3.1.4 VideoLearnAI como Ferramenta de Aprendizagem Ativa

Para superar os desafios da aprendizagem passiva, a VideoLearnAI implementa diversas estratégias baseadas em princípios de aprendizagem ativa, cada uma materializada em funcionalidades específicas da aplicação:

- **Segmentação e estruturação:** Dividir vídeos longos em segmentos menores e conceitualmente coerentes melhora a retenção e reduz a carga cognitiva. Na aplicação, esta estratégia é implementada através da **geração automática de capítulos**, que analisa o conteúdo do vídeo e identifica pontos de transição entre tópicos, criando uma estrutura navegável.
- **Questionamento e testes:** Incorporar perguntas e quizzes durante ou após a visualização do vídeo ativa o processo de recuperação e identifica lacunas de compreensão. A aplicação implementa esta estratégia através dos **quizzes interativos personalizados** para cada capítulo, oferecendo tanto questões de verdadeiro/falso quanto perguntas discursivas com feedback imediato.
- **Anotação e interação com o conteúdo:** Permitir que os estudantes interajam diretamente com o conteúdo do vídeo promove processamento mais profundo do

material. Esta estratégia é concretizada no **sistema de bate-papo contextual**, que permite ao usuário fazer perguntas específicas sobre o conteúdo e receber respostas contextualizadas baseadas nas informações do vídeo.

- **Transcrições interativas:** Fornecer transcrições navegáveis permite que os estudantes revisitem conceitos específicos e processem o material no seu próprio ritmo. A aplicação implementa esta estratégia através da **melhoria de legibilidade das legendas**, que transforma legendas desconexas em texto estruturado e coeso, e da **transcrição de alta qualidade** que supera as limitações das legendas automáticas do YouTube.
- **Metacognição e reflexão:** Promover a reflexão sobre o conteúdo aprendido fortalece a compreensão e retenção. Esta estratégia é implementada através do **sistema de perguntas e respostas** e da possibilidade de **personalização dos prompts** para geração de quizzes, incentivando o usuário a refletir sobre o tipo de questões mais relevantes para seu aprendizado.

Cada uma dessas funcionalidades foi projetada não apenas como uma ferramenta isolada, mas como parte de um ecossistema integrado que transforma a experiência de assistir a um vídeo em um processo de aprendizagem ativo e estruturado.

A integração dessas funcionalidades cria um fluxo de aprendizagem que supera as limitações do consumo passivo de vídeos. Um usuário típico pode, por exemplo, inicialmente dividir o conteúdo em pequenos segmentos (gerando capítulos), assistir a um segmento do vídeo, consultar a transcrição melhorada para revisar conceitos complexos, testar sua compreensão com quizzes personalizados e, finalmente, aprofundar seu entendimento através de perguntas específicas no chat contextual.

Esta abordagem integrada busca transformar a experiência de aprendizagem com vídeos, elevando-a do nível passivo para níveis mais profundos de engajamento cognitivo. Ao combinar diferentes estratégias de interação com o conteúdo, a aplicação desenvolvida nesse trabalho cria um ambiente de aprendizado mais eficaz, onde o estudante não apenas consome informação, masativamente processa, questiona e aplica o conhecimento adquirido, resultando em melhor compreensão e retenção a longo prazo (Prince, 2004).

3.2 Large Language Models (LLMs)

Os Large Language Models (LLMs) representam um avanço significativo no campo da Inteligência Artificial, especificamente no Processamento de Linguagem Natural (PLN). Estes modelos têm revolucionado diversas áreas, desde assistentes virtuais até sistemas educacionais, como o proposto neste trabalho. Esta seção explora

os conceitos fundamentais, arquitetura, técnicas de prompt engineering e aspectos relacionados a limitações desses modelos.

3.2.1 Conceitos Fundamentais

Large Language Models são sistemas de inteligência artificial treinados em vastos corpora de texto para aprender padrões estatísticos da linguagem humana. Diferentemente dos sistemas tradicionais de PLN, que frequentemente dependiam de regras pré-definidas ou características manualmente extraídas, os LLMs aprendem representações contextuais ricas diretamente dos dados, como demonstrado no trabalho "Language Models are Few-Shot Learners" (Brown et al., 2020) (Brown; Mann; Ryder; Subbiah; Kaplan; Dhariwal; Neelakantan; Shyam; Sastry; Askell et al., 2020).

O desenvolvimento dos LLMs pode ser compreendido como uma evolução natural dos modelos de linguagem estatísticos. Enquanto os modelos tradicionais, como n-gramas, previam a próxima palavra baseando-se apenas em um contexto limitado de palavras anteriores, os LLMs modernos consideram contextos muito mais amplos e capturam dependências de longo alcance no texto, conforme apresentado no artigo "Attention is All You Need" (Vaswani et al., 2017) (Vaswani; Shazeer; Parmar; Uszkoreit; Jones; Gomez; Kaiser; Polosukhin, 2017).

Um aspecto fundamental dos LLMs é sua capacidade de aprendizado de representações contextuais. Modelos como BERT ("BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding") (Devlin et al., 2019) (Devlin; Chang; Lee; Toutanova, 2019) e GPT ("Language Models are Unsupervised Multitask Learners") (Radford et al., 2019) (Radford; Wu; Child; Luan; Amodei; Sutskever, 2019) aprendem representações vetoriais (embeddings) para palavras que variam dependendo do contexto em que aparecem, capturando assim nuances semânticas que modelos anteriores não conseguiam.

O treinamento destes modelos ocorre em duas fases principais: pré-treinamento e ajuste fino (fine-tuning). Durante o pré-treinamento, o modelo é exposto a enormes quantidades de texto não rotulado, aprendendo padrões linguísticos gerais. No ajuste fino, o modelo é especializado para tarefas específicas usando conjuntos de dados menores e rotulados.

Um conceito crucial para entender os LLMs modernos é o de aprendizado auto-supervisionado. Nesta abordagem, o próprio texto serve como supervisão, com o modelo sendo treinado para prever partes mascaradas do texto (como no BERT) ou a próxima palavra na sequência (como no GPT). Esta capacidade de extrair sinais de supervisão dos próprios dados permitiu o treinamento em escala sem precedentes.

3.2.2 Arquitetura e Funcionamento

A arquitetura predominante nos LLMs modernos é baseada no Transformer, introduzido no trabalho seminal "Attention is All You Need" (Vaswani et al., 2017) (Vaswani; Shazeer; Parmar; Uszkoreit; Jones; Gomez; Kaiser; Polosukhin, 2017). Esta arquitetura revolucionou o PLN ao substituir as redes recorrentes (RNNs) e convolucionais (CNNs) por um mecanismo de atenção que permite processar todas as palavras de uma sequência simultaneamente, em vez de sequencialmente.

O componente central da arquitetura Transformer é o mecanismo de auto-atenção (self-attention), que permite ao modelo ponderar a importância de diferentes palavras em uma sentença ao processar cada palavra. Isso possibilita a captura de dependências de longo alcance no texto, independentemente da distância entre as palavras relacionadas.

Os LLMs modernos podem ser classificados em três categorias principais baseadas na arquitetura Transformer:

- **Encoder-only:** Modelos como BERT ("BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding") (Devlin et al., 2019) (Devlin; Chang; Lee; Toutanova, 2019) utilizam apenas a parte do codificador do Transformer. São bidirecionais, considerando tanto o contexto à esquerda quanto à direita de cada palavra, e são particularmente eficazes em tarefas de compreensão de linguagem.
- **Decoder-only:** Modelos como GPT ("Language Models are Unsupervised Multitask Learners") (Radford et al., 2019) (Radford; Wu; Child; Luan; Amodei; Sutskever, 2019) utilizam apenas a parte do decodificador. São unidirecionais, considerando apenas o contexto à esquerda (palavras anteriores) e são especializados em geração de texto.
- **Encoder-decoder:** Modelos como T5 ("Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer") (Raffel et al., 2020) (Raffel; Shazeer; Roberts; Lee; Narang; Matena; Zhou; Li; Liu, 2020) utilizam ambas as partes, sendo eficazes tanto para compreensão quanto para geração de texto, especialmente em tarefas de transformação de texto como tradução e resumo.

O escalonamento dos modelos tem sido um fator crucial para o avanço dos LLMs. Pesquisas demonstraram que aumentar o número de parâmetros, a quantidade de dados de treinamento e o poder computacional leva a melhorias consistentes no desempenho, como evidenciado no estudo "Scaling Laws for Neural Language Models" (Kaplan et al., 2020) (Kaplan; Mccandlish; Henighan; Brown; Chess; Child; Gray; Radford; Wu; Amodei, 2020). Este fenômeno, conhecido como "scaling laws",

tem guiado o desenvolvimento de modelos cada vez maiores, como o GPT-3 com 175 bilhões de parâmetros e o GPT-4, cujo número exato de parâmetros não foi divulgado, mas estima-se que seja significativamente maior ("GPT-4 Technical Report") (OpenAI, 2023) (OpenAI, 2023a).

3.2.3 Prompt Engineering

Prompt engineering refere-se à prática de formular instruções ou consultas (prompts) para LLMs de maneira a obter respostas mais precisas, relevantes e úteis. Com o advento de modelos como GPT-3 e GPT-4, capazes de realizar diversas tarefas sem fine-tuning específico, a qualidade do prompt tornou-se um fator determinante para o desempenho do modelo, como discutido em "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing"(Liu et al., 2023) (Liu; Yuan; Fu; Jiang; Hayashi; Neubig, 2023).

A eficácia do prompt engineering baseia-se no fenômeno de "in-context learning", onde os LLMs podem adaptar seu comportamento com base apenas nos exemplos ou instruções fornecidos no prompt, sem modificação de seus parâmetros, como demonstrado em "Language Models are Few-Shot Learners"(Brown et al., 2020) (Brown; Mann; Ryder; Subbiah; Kaplan; Dhariwal; Neelakantan; Shyam; Sastry; Askell et al., 2020). Este comportamento emergente permite que os modelos realizem tarefas para as quais não foram explicitamente treinados.

Várias técnicas de prompt engineering têm sido desenvolvidas e estudadas:

- **Zero-shot prompting:** Fornecer apenas a instrução, sem exemplos, confiando na capacidade do modelo de entender a tarefa, como explorado em "Large Language Models are Zero-Shot Reasoners"(Kojima et al., 2022) (Kojima; Gu; Reid; Matsuo; Iwasawa, 2022).
- **Few-shot prompting:** Incluir alguns exemplos de entrada-saída no prompt para guiar o modelo, técnica apresentada em "Language Models are Few-Shot Learners"(Brown et al., 2020) (Brown; Mann; Ryder; Subbiah; Kaplan; Dhariwal; Neelakantan; Shyam; Sastry; Askell et al., 2020).
- **Chain-of-thought prompting:** Solicitar ao modelo que explique seu raciocínio passo a passo, melhorando significativamente o desempenho em tarefas de raciocínio complexo, como demonstrado em "Chain of Thought Prompting Elicits Reasoning in Large Language Models"(Wei et al., 2022) (Wei; Wang; Schuurmans; Bosma; Ichter; Xia; Chi; Le; Zhou, 2022).
- **ReAct:** Combinar raciocínio e ações, permitindo que o modelo interaja com ferramentas externas para resolver problemas, abordagem introduzida em "ReAct:

"Synergizing Reasoning and Acting in Language Models" (Yao et al., 2023) (Yao; Zhao; Yu; Du; Shafran; Narasimhan; Cao, 2023).

A estruturação do prompt também é crucial. Pesquisas mostram que a ordem dos exemplos, a formatação do texto e até mesmo a escolha de palavras específicas podem afetar significativamente o desempenho do modelo. Além disso, a inclusão de metadados como o papel que o modelo deve assumir (por exemplo, "Você é um assistente útil") pode influenciar o estilo e a qualidade das respostas.

Esta técnica é especialmente valiosa na geração de quizzes personalizados, permitindo que os usuários customizem o tipo, dificuldade e formato das perguntas geradas. Ao ajustar cuidadosamente os prompts, é possível direcionar o modelo para criar questões que enfatizem conceitos específicos, adotem determinadas abordagens pedagógicas ou se adequem a diferentes níveis de conhecimento, tornando a experiência de aprendizado mais eficaz e personalizada.

3.2.4 Limitações

Apesar de seu impressionante desempenho em diversas tarefas, os LLMs apresentam limitações fundamentais que impactam significativamente sua aplicação prática. Estas restrições podem ser categorizadas em limitações de janela de contexto e limitações arquiteturais intrínsecas.

3.2.4.1 Janela de Contexto

A janela de contexto refere-se à quantidade máxima de texto que um LLM pode processar em uma única operação. Esta limitação é uma consideração crítica no design de aplicações baseadas em LLMs, especialmente aquelas que lidam com documentos longos ou conversas extensas, como investigado no estudo "Lost in the Middle: How Language Models Use Long Contexts" (Liu et al., 2023) (Liu; Bosselut; Srinivasan; Choi; Hajishirzi; Khashabi, 2023).

Os LLMs processam texto dividindo-o em unidades chamadas tokens, que podem representar palavras, partes de palavras ou caracteres individuais, dependendo do algoritmo de tokenização utilizado. O número de tokens que um modelo pode processar simultaneamente é determinado por limitações de hardware (principalmente memória GPU) e pela arquitetura do modelo, como explicado em "Attention is All You Need" (Vaswani et al., 2017) (Vaswani; Shazeer; Parmar; Uszkoreit; Jones; Gomez; Kaiser; Polosukhin, 2017).

A evolução dos LLMs tem sido marcada por um aumento constante no tamanho da janela de contexto. Modelos iniciais como o GPT-2 ("Language Models are Unsupervised Multitask Learners") (Radford et al., 2019) (Radford; Wu; Child; Luan; Amodei; Sutskever, 2019) tinham uma janela de contexto de 1.024 tokens, enquanto

versões recentes como o GPT-4 Turbo ("GPT-4 Technical Report") (OpenAI, 2023) (OpenAI, 2023a) podem processar até 128.000 tokens. Este aumento permite aplicações mais sofisticadas, como análise de documentos longos, resumo de livros inteiros e manutenção de conversas extensas com histórico completo.

No entanto, mesmo com janelas de contexto expandidas, os LLMs enfrentam desafios ao lidar com contextos longos. Pesquisas recentes identificaram o fenômeno de "lost in the middle", onde informações localizadas no meio de um texto longo têm menor probabilidade de serem utilizadas pelo modelo em suas respostas, como demonstrado em "Lost in the Middle: How Language Models Use Long Contexts" (Liu et al., 2023) (Liu; Bosselut; Srinivasan; Choi; Hajishirzi; Khashabi, 2023). Este efeito pode ser atribuído a limitações no mecanismo de atenção e à forma como os modelos foram treinados.

Para mitigar as limitações da janela de contexto, várias estratégias têm sido desenvolvidas:

- **Chunking:** Dividir documentos longos em segmentos menores que podem ser processados separadamente, como discutido em "Retrieval-Augmented Generation for Large Language Models: A Survey" (Gao et al., 2023) (Gao; Xiong; Gao; Jiang; Shen; Ren; Han, 2023).
- **Sliding window:** Processar o texto em janelas sobrepostas para manter a coerência entre segmentos, abordagem proposta em "Longformer: The Long-Document Transformer" (Beltagy et al., 2020) (Beltagy; Peters; Cohan, 2020).
- **Retrieval-Augmented Generation (RAG):** Combinar LLMs com sistemas de recuperação de informação para acessar conhecimento externo sem sobrecarregar a janela de contexto, método introduzido em "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" (Lewis et al., 2020) (Lewis; Perez; Piktus; Petroni; Karpukhin; Goyal; Küttler; Lewis; Yih; Rocktäschel et al., 2020).
- **Compressão de contexto:** Resumir partes do histórico de conversação para liberar espaço na janela de contexto, técnica explorada em "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" (Xu et al., 2023) (Xu; Sarthi; Agarwal; Gupta; Saxena; Aralikatte; Batra; Parikh; Misra; Awadallah, 2023).

A eficiência no uso da janela de contexto é particularmente importante em aplicações como a proposta neste trabalho, onde é necessário processar transcrições de vídeos potencialmente longas. A implementação de técnicas como chunking inteligente baseado em capítulos ou segmentos temáticos pode melhorar significativamente a qualidade das respostas geradas pelo modelo, como sugerido em "Retrieval-Augmented Generation for Large Language Models: A Survey" (Gao et al., 2023) (Gao; Xiong; Gao; Jiang; Shen; Ren; Han, 2023).

Além disso, a otimização do uso de tokens é crucial para controlar custos em aplicações comerciais, já que os serviços de API de LLMs geralmente cobram com base no número de tokens processados (OpenAI, 2023) (OpenAI, 2023b). Estratégias como a remoção de informações redundantes e a priorização de conteúdo relevante podem reduzir significativamente o consumo de tokens sem comprometer a qualidade das respostas.

3.2.4.2 Limitações Arquiteturais

Além das restrições de janela de contexto, os LLMs apresentam limitações fundamentais inerentes à sua arquitetura. Dois fenômenos críticos que afetam significativamente o desempenho destes modelos são o *representational collapse* e o *over-squashing*, como revelado no trabalho “Transformers need glasses!” (Barbero; Banino; Kapturowski; Kumaran; Araújo; Vitvitskyi; Pascanu; Veličković, 2024).

O *representational collapse* ocorre quando sequências de entrada distintas geram representações praticamente idênticas no token final do modelo. Este fenômeno é especialmente problemático em sequências longas, onde a representação do último token tende a convergir mesmo quando há diferenças significativas no conteúdo. O problema é ainda mais acentuado pelo uso de formatos de ponto flutuante de baixa precisão comumente empregados em LLMs modernos.

Já o *over-squashing* refere-se à perda progressiva de sensibilidade do modelo a tokens específicos na entrada. Este fenômeno é uma consequência direta da arquitetura dos transformers decoder-only, onde o fluxo unidirecional de informação converge no token final. Tokens que aparecem no início da sequência têm mais caminhos para propagar sua informação até o token final, enquanto tokens posteriores têm menos caminhos, resultando em perda de informação importante.

Estas limitações são particularmente relevantes em aplicações que exigem processamento de textos longos ou análise detalhada de conteúdo, como a melhoria de legibilidade de legendas ou a geração de resumos de documentos extensos. No desenvolvimento do VideoLearnAI, estas restrições manifestaram-se concretamente durante as tentativas iniciais de utilizar LLMs para melhorar a legibilidade das transcrições de vídeos.

As legendas de vídeos educacionais frequentemente são extensas, podendo ultrapassar facilmente os limites da janela de contexto dos modelos. Além disso, o fenômeno de *representational collapse* resultava em perda de informações importantes nas seções intermediárias das transcrições, enquanto o *over-squashing* comprometia a capacidade do modelo de manter a coerência ao longo de todo o texto processado. Estas limitações levaram a resultados inconsistentes: algumas partes da transcrição eram adequadamente reformatadas, enquanto outras apresentavam problemas como omissão de conteúdo, adição de texto não presente no original ou perda da estrutura

semântica. Compreender estas restrições foi fundamental para desenvolver estratégias que as contornassem ou minimizassem seus efeitos, como a utilização de modelos especializados para tarefas específicas (no caso, o modelo SAT para segmentação de texto, apresentado por Frohmann et al. (2024) (Frohmann; Sterner; Vulic; Minixhofer; Schedl, 2024)).

3.3 LLM Powered Applications

O desenvolvimento de aplicações potencializadas por Large Language Models (LLMs) representa uma nova fronteira na engenharia de software, onde sistemas de inteligência artificial são integrados como componentes centrais de produtos e serviços. Esta seção explora os conceitos fundamentais, arquiteturas e técnicas para construção de aplicações eficazes baseadas em LLMs.

3.3.1 Arquiteturas de Aplicações com LLMs

As aplicações potencializadas por LLMs podem ser categorizadas em diferentes arquiteturas, dependendo do papel que o modelo desempenha no sistema e como ele interage com outros componentes. Segundo Bommasani et al. (2021) (Bommasani; Hudson; Adeli; Altman; Arora; Arx; Bernstein; Bohg; Bosselut; Brunskill et al., 2021), estas arquiteturas podem ser classificadas em:

- **Aplicações de interface direta:** Onde o usuário interage diretamente com o LLM através de uma interface conversacional, como em assistentes virtuais e chatbots. Nestes sistemas, o LLM atua como o principal mecanismo de processamento e geração de respostas.
- **Aplicações aumentadas por LLM:** Onde o LLM complementa funcionalidades existentes, como em editores de texto com sugestões de escrita, ferramentas de análise de dados com geração de insights, ou sistemas de busca com respostas geradas.
- **Aplicações orquestradas por LLM:** Onde o LLM atua como um controlador central que coordena diferentes componentes e serviços, delegando tarefas específicas e integrando os resultados em uma resposta coesa.

A escolha da arquitetura adequada depende dos requisitos específicos da aplicação, considerando fatores como latência, custo, privacidade e complexidade do domínio, como discutido por Liu et al. (2023) (Liu; Sun; Zhong; Han; Sun; Zhang; Peng; Tang; Huang; Lai et al., 2023).

3.3.2 Function Calling

Function Calling representa um avanço significativo na integração de LLMs com sistemas de software, permitindo que os modelos interajam de forma estruturada com funções e APIs externas. Esta capacidade transforma os LLMs de meros geradores de texto em componentes capazes de acionar ações específicas no mundo real, expandindo drasticamente seu potencial de aplicação.

O conceito de Function Calling, também conhecido como "Tool Use" em alguns contextos, permite que um LLM:

- Reconheça quando uma solicitação do usuário requer a execução de uma função específica
- Determine qual função deve ser chamada entre um conjunto de funções disponíveis
- Gere os parâmetros apropriados para essa função em um formato estruturado (geralmente JSON)
- Interprete os resultados da execução da função e os incorpore em sua resposta

Como explicado por White et al. (2023) (White; Fu; Hays; Sandborn; Olea; Gilbert; Elnashar; Spencer-smith; Schmidt, 2023), o Function Calling é implementado através de um processo de duas etapas: primeiro, o modelo determina se uma função deve ser chamada e, em caso afirmativo, gera uma representação estruturada dos argumentos; em seguida, o sistema executa a função com esses argumentos e fornece o resultado de volta ao modelo para incorporação na resposta final.

Esta capacidade é particularmente valiosa para superar limitações inerentes aos LLMs, como:

- **Acesso a informações atualizadas:** Permitindo consultas a APIs externas para obter dados em tempo real que não estavam disponíveis durante o treinamento do modelo.
- **Execução de cálculos complexos:** Delegando operações matemáticas ou computacionais a sistemas especializados, evitando os erros comuns dos LLMs nessas tarefas.
- **Interação com sistemas externos:** Possibilitando ações como envio de e-mails, agendamento de compromissos ou consulta a bancos de dados.
- **Garantia de saídas estruturadas:** Assegurando que as respostas sigam um formato específico, crucial para integração com outros sistemas.

Na implementação prática, o Function Calling geralmente envolve a definição de um schema que especifica o nome da função, sua descrição e a estrutura esperada dos parâmetros, frequentemente utilizando formatos como JSON Schema. Este schema é fornecido ao modelo junto com o prompt do usuário, permitindo que o LLM determine quando e como chamar a função apropriada, como demonstrado por Vemprala et al. (2023) (Vemprala; Bonatti; Bucker; Kapoor, 2023).

No contexto da VideoLearnAI, o Function Calling é utilizado em várias funcionalidades críticas, como a geração de capítulos e quizzes, onde o modelo precisa produzir saídas estruturadas que seguem um formato específico para integração com o restante do sistema.

3.3.3 Agentes Autônomos Baseados em LLMs

Uma tendência emergente no desenvolvimento de aplicações potencializadas por LLMs é a criação de agentes autônomos - sistemas que podem executar tarefas complexas com mínima supervisão humana, tomando decisões e adaptando-se a diferentes contextos. Como explorado por Yao et al. (2023) (Yao; Zhao; Yu; Du; Shafran; Narasimhan; Cao, 2023), estes agentes combinam LLMs com capacidades de planejamento, raciocínio e interação com o ambiente.

Os agentes baseados em LLMs geralmente implementam um ciclo de "pensamento-ação-observação", onde o modelo:

- Analisa a situação atual e planeja os próximos passos (pensamento)
- Executa ações através de chamadas a funções ou APIs (ação)
- Processa o resultado dessas ações e atualiza seu entendimento (observação)

Esta abordagem, conhecida como ReAct (Reasoning and Acting), permite que os agentes decomponham problemas complexos em etapas gerenciáveis, adaptem-se a resultados inesperados e mantenham um foco consistente em objetivos de longo prazo.

Weng (2023) (Weng, 2023) identifica vários componentes-chave em arquiteturas de agentes baseados em LLMs:

- **Memória:** Sistemas para armazenar e recuperar informações relevantes ao longo do tempo, incluindo memória de curto prazo (contexto da conversa atual) e longo prazo (conhecimento persistente).
- **Planejamento:** Mecanismos para decompor objetivos complexos em sub-tarefas e determinar a sequência ótima de ações.
- **Ferramentas:** Conjunto de funções e APIs que o agente pode utilizar para interagir com sistemas externos e executar ações concretas.

- **Reflexão:** Capacidade de avaliar o próprio desempenho, identificar erros e ajustar estratégias futuras.

A evolução dos agentes autônomos baseados em LLMs representa um passo significativo na direção de sistemas de IA mais capazes e independentes. À medida que estas tecnologias amadurecem, observa-se uma transição gradual de assistentes passivos que respondem apenas quando solicitados para agentes proativos que antecipam necessidades, sugerem ações e executam tarefas complexas com autonomia crescente.

Esta progressão abre caminho para aplicações educacionais que não apenas respondem às dúvidas dos estudantes, mas ativamente monitoram seu progresso, identificam lacunas de conhecimento e personalizam dinamicamente o material de estudo, criando experiências de aprendizado verdadeiramente adaptativas e centradas no usuário. No futuro próximo, podemos esperar que estes agentes se tornem colaboradores cada vez mais sofisticados no processo educacional, complementando - e não substituindo - o papel dos educadores humanos.

3.3.4 Retrieval-Augmented Generation (RAG) e RAG Agêntico

Retrieval-Augmented Generation (RAG) representa uma abordagem híbrida que combina a capacidade generativa dos LLMs com a precisão de sistemas de recuperação de informação. Introduzida por Lewis et al. (2020) (Lewis; Perez; Piktus; Petroni; Karpukhin; Goyal; Kütller; Lewis; Yih; Rocktäschel et al., 2020), esta técnica permite que os modelos acessem e incorporem conhecimento externo durante a geração de respostas, superando limitações como informações desatualizadas ou conhecimento especializado ausente no treinamento original.

O funcionamento do RAG pode ser dividido em três etapas principais:

- **Indexação:** Documentos ou fontes de conhecimento são processados e transformados em representações vetoriais (embeddings) que capturam seu conteúdo semântico. Estes vetores são armazenados em uma base de dados vetorial para recuperação eficiente.
- **Recuperação:** Quando uma consulta é recebida, ela também é transformada em um vetor e utilizada para buscar os documentos mais relevantes na base indexada, geralmente através de métricas de similaridade como distância cosenoideal.
- **Geração aumentada:** Os documentos recuperados são incorporados ao contexto fornecido ao LLM, que então gera uma resposta que integra tanto seu conhecimento interno quanto as informações externas recuperadas.

Como destacado por Gao et al. (2023) (Gao; Xiong; Gao; Jiang; Shen; Ren; Han, 2023), o RAG oferece várias vantagens significativas para aplicações baseadas em LLMs:

- **Atualização de conhecimento:** Permite que o modelo acesse informações mais recentes sem necessidade de retreinamento.
- **Redução de alucinações:** Ao fundamentar as respostas em fontes externas verificáveis, diminui a tendência dos LLMs de gerar informações incorretas ou fabricadas.
- **Especialização em domínios:** Possibilita a adaptação do modelo para domínios específicos através da indexação de literatura especializada.
- **Transparência e rastreabilidade:** As fontes utilizadas para gerar respostas podem ser citadas, aumentando a confiabilidade e verificabilidade.

No contexto de aplicações educacionais como a VideoLearnAI, o RAG pode ser particularmente valioso para garantir que as respostas geradas pelo sistema sejam precisas e fundamentadas no conteúdo específico do vídeo, evitando a introdução de informações externas que possam confundir o usuário. Esta abordagem seria especialmente útil para processar vídeos muito longos, cujas transcrições excedem a janela de contexto dos LLMs.

Embora a implementação atual da VideoLearnAI contorne essa limitação permitindo que usuários selecionem manualmente capítulos específicos de interesse e fornecendo alertas quando a seleção ultrapassa os limites do contexto, a integração futura de RAG poderia automatizar esse processo, indexando toda a transcrição e recuperando apenas as partes mais relevantes para cada consulta, sem necessidade de intervenção manual do usuário.

Uma evolução significativa neste campo é o conceito de RAG Agêntico (Agentic RAG), como apresentado por Singh et al. (2025) (Singh; Ehtesham; Kumar; Khoei, 2025). Esta abordagem transcende as limitações dos sistemas RAG tradicionais, que frequentemente seguem fluxos de trabalho estáticos, incorporando agentes autônomos de IA na pipeline de RAG. Estes agentes utilizam padrões de design agêntico como reflexão, planejamento, uso de ferramentas e colaboração multi-agente para gerenciar dinamicamente estratégias de recuperação, refinar iterativamente a compreensão contextual e adaptar fluxos de trabalho para atender a requisitos de tarefas complexas.

A implementação de um sistema RAG agêntico no VideoLearnAI poderia não apenas recuperar as partes relevantes da transcrição, mas também tomar decisões sobre quais informações são mais importantes para o contexto atual, realizar análises comparativas entre diferentes seções do vídeo, e até mesmo buscar informações complementares

em outros vídeos relacionados quando apropriado, tudo isso mantendo uma interação natural e contextualizada com o usuário. Esta abordagem proporcionaria flexibilidade, escalabilidade e consciência contextual, superando as limitações da seleção manual de contexto atualmente implementada na plataforma.

3.4 Conclusão do capítulo

A fundamentação teórica apresentada neste capítulo fornece a base conceitual que orientou o desenvolvimento do VideoLearnAI. Esta seção sintetiza os principais conceitos e explicita as decisões estratégicas tomadas durante o projeto.

Os princípios de aprendizagem ativa demonstram que o engajamento cognitivo produz resultados educacionais superiores ao consumo passivo de conteúdo (Freeman; Eddy; McDonough; Smith; Okoroafor; Jordt; Wenderoth, 2014). Simultaneamente, os avanços em Large Language Models (LLMs) oferecem novas possibilidades para transformar experiências educacionais, apesar de suas limitações inerentes como janelas de contexto restritas (Liu; Bosselut; Srinivasan; Choi; Hajishirzi; Khashabi, 2023) e desafios arquiteturais como representational collapse (Barbero; Banino; Kapturowski; Kumaran; Araújo; Vitvitskyi; Pascanu; Veličković, 2024).

A convergência destes campos teóricos levou a decisões fundamentais que moldaram a arquitetura e funcionalidades do VideoLearnAI:

- **Arquitetura híbrida e especializada:** Em vez de depender exclusivamente de LLMs para todas as funcionalidades, optou-se por uma abordagem que combina modelos especializados para tarefas específicas. Esta decisão reconhece que, apesar da versatilidade dos LLMs, modelos dedicados frequentemente oferecem melhor desempenho, eficiência e previsibilidade para funções específicas, como exemplificado pela adoção do modelo SaT para segmentação de texto.
- **Estruturação semântica do conteúdo:** A organização do conteúdo em capítulos semânticos, em vez de divisões arbitrárias baseadas em tempo, fundamenta-se tanto em princípios cognitivos quanto em limitações técnicas dos LLMs. Esta abordagem facilita a compreensão humana e otimiza o processamento computacional, permitindo interações mais granulares e contextualizadas.
- **Transparência e controle do usuário:** A decisão de expor prompts para personalização e oferecer controle explícito sobre o contexto considerado reflete um compromisso com a transparência e adaptabilidade. Esta abordagem reconhece a diversidade de estilos de aprendizagem e objetivos educacionais, permitindo que o sistema se adapte às necessidades específicas de cada usuário.
- **Multimodalidade interativa:** A implementação de diferentes modalidades de

interação (quizzes, perguntas abertas, navegação estruturada) baseia-se na compreensão de que a aprendizagem efetiva envolve diversos processos cognitivos. Esta diversidade de interações promove um engajamento mais completo com o material, ativando diferentes mecanismos de processamento e retenção de conhecimento.

- **Function calling para estruturação de saídas:** Implementou-se function calling para garantir que as saídas dos LLMs seguissem formatos estruturados e previsíveis, especialmente na geração de quizzes e capítulos. Esta técnica permitiu maior confiabilidade na integração entre o LLM e os componentes da aplicação, reduzindo erros de parsing e garantindo consistência na experiência do usuário.
- **Abordagem simplificada para gerenciamento de contexto:** Optou-se por uma solução de seleção manual de contexto para o bate-papo, em vez de implementar sistemas mais complexos como RAG ou agentes autônomos. Apesar dos benefícios teóricos dessas tecnologias avançadas para lidar com conteúdos extensos, sua implementação exigiria recursos adicionais significativos. Esta decisão pragmática permitiu focar no desenvolvimento das funcionalidades essenciais, oferecendo ao usuário controle explícito sobre o contexto considerado e simplificando a implementação técnica.

Este capítulo apresentou a fundamentação teórica que sustenta o desenvolvimento deste trabalho, abordando os princípios de aprendizagem ativa, as características e limitações dos LLMs, e as arquiteturas de aplicações baseadas em IA. A revisão destes conceitos foi fundamental para embasar as decisões técnicas e de design tomadas durante o desenvolvimento da solução proposta. Os próximos capítulos detalharão como estes fundamentos teóricos foram aplicados na prática, descrevendo as tecnologias utilizadas e o processo de implementação da plataforma VideoLearnAI.

4 TECNOLOGIAS UTILIZADAS

4.1 Frameworks e Bibliotecas

4.1.1 FastAPI

FastAPI foi utilizado para criar um serviço backend complementar em Python, expondo endpoints para funcionalidades específicas de extração de dados do YouTube e operações com o modelo SaT(Segment Any Text). Um diferencial importante foi a geração automática de clients TypeScript através do OpenAPI, permitindo uma integração tipo-segura e seamless com o frontend Next.js. Esta capacidade de geração automática de código TypeScript eliminou a necessidade de definir tipos manualmente e garantiu consistência na comunicação entre os serviços.

O FastAPI utiliza o Pydantic como base para validação de dados, serialização e documentação. Esta integração oferece várias vantagens significativas:

- **Validação automática:** O Pydantic valida automaticamente os dados de entrada e saída, garantindo que correspondam aos tipos esperados e reduzindo a necessidade de código de validação manual.
- **Documentação automática:** Os modelos Pydantic são utilizados para gerar documentação OpenAPI detalhada, incluindo exemplos de requisição e resposta.
- **Geração de código TypeScript:** A documentação OpenAPI gerada pelo FastAPI pode ser consumida por ferramentas como o OpenAPI TypeScript Generator para criar automaticamente interfaces TypeScript, tipos e clients para o frontend.

Esta integração entre FastAPI, Pydantic e Next.js criou um fluxo de desenvolvimento produtivo, onde alterações na API Python são automaticamente refletidas no código TypeScript do frontend, reduzindo erros de integração e melhorando a experiência de desenvolvimento full-stack.

4.1.2 Next.js

Next.js atuou como o framework full-stack principal, gerenciando tanto o frontend quanto a maior parte do backend da aplicação. Através de seus recursos de API Routes

e Server Components, foi possível construir uma aplicação completa com renderização híbrida, manipulação de estado servidor/cliente e APIs RESTful. Sua arquitetura permitiu manter a maioria das operações de backend centralizadas, recorrendo ao serviço Python apenas para funcionalidades específicas.

4.1.3 Vercel AI SDK

O **Vercel AI SDK** desempenhou um papel fundamental na implementação de funcionalidades de inteligência artificial diretamente no **Next.js**, proporcionando uma integração eficiente com modelos de linguagem natural (LLMs).

Uma das principais vantagens dessa biblioteca é a facilidade na construção de interfaces de usuário interativas e dinâmicas para aplicações baseadas em IA. O SDK permite o **streaming de respostas**, garantindo uma experiência mais fluida para o usuário, sem a necessidade de aguardar a conclusão total do processamento. Essa abordagem melhora significativamente a experiência do usuário (**UX**), tornando as interações mais ágeis e responsivas.

Além do texto, outros elementos da interface, como **quizzes gerados dinamicamente**, também são entregues por meio de streaming. Isso significa que as perguntas e respostas dos quizzes não precisam ser completamente processadas antes de serem exibidas, pois são disponibilizadas conforme são geradas pelo modelo de IA. Todo esse processo é abstraído pelo **Vercel AI SDK**, simplificando a implementação e reduzindo a complexidade no desenvolvimento da interface.

Dessa forma, o uso do **Vercel AI SDK** permitiu a criação de uma **UI responsiva e eficiente**, eliminando a necessidade de comunicação constante com o backend Python para determinadas operações de IA, otimizando o desempenho e a escalabilidade da aplicação.

4.1.4 Drizzle ORM

O **Drizzle ORM** foi utilizado como ORM principal no **Next.js**, oferecendo uma interface *type-safe* para interações com o banco de dados **PostgreSQL**. Sua integração direta com **TypeScript** permitiu manter a consistência de tipos entre o banco de dados e a aplicação.

Além disso, o uso do **Drizzle ORM** faz sentido dentro da arquitetura **serverless** do projeto. Diferente de ORMs mais tradicionais, que podem apresentar desafios com conexões persistentes em ambientes serverless, o **Drizzle ORM** foi projetado para ser leve e eficiente, garantindo melhor compatibilidade com essa abordagem. Dessa forma, a aplicação pode escalar dinamicamente sem sobrecarga desnecessária na comunicação com o banco de dados.

4.1.5 ShadCN

ShadCN forneceu componentes de UI reutilizáveis e personalizáveis, sendo utilizado exclusivamente na camada de frontend do Next.js para construir interfaces consistentes e acessíveis. Diferentemente de bibliotecas de componentes tradicionais, o ShadCN não é instalado como um pacote npm, mas sim copiado diretamente para o código-fonte do projeto, permitindo total personalização e controle sobre os componentes. Esta abordagem "copy-paste" elimina dependências externas e facilita a adaptação dos componentes às necessidades específicas do projeto.

4.1.6 Zustand

Zustand foi utilizado para gerenciar o estado global do frontend, oferecendo uma solução leve e eficiente que complementa o gerenciamento de estado do Next.js. Um dos principais desafios no desenvolvimento React é a organização do estado compartilhado entre múltiplos componentes. Abordagens tradicionais como prop drilling tornam-se inviáveis em aplicações complexas, enquanto o Context API frequentemente causa renderizações desnecessárias em componentes que não precisam reagir a todas as mudanças.

A principal vantagem do Zustand é seu sistema de assinaturas seletivas, permitindo que componentes se inscrevam apenas nas partes específicas do estado que realmente utilizam. Isso significa que um componente é renderizado novamente somente quando os dados que ele observa são modificados, evitando ciclos de renderização desnecessários e melhorando a performance. Sua API minimalista baseada em hooks, sem boilerplate extenso, também contribuiu para manter o código limpo e legível, facilitando a implementação de um gerenciamento de estado sofisticado com menos complexidade.

4.1.7 Stripe

O Stripe foi implementado como solução de pagamento para o modelo de assinatura da plataforma. Webhooks foram configurados para sincronizar eventos de pagamento com o banco de dados da aplicação, atualizando automaticamente o status das assinaturas. A implementação seguiu o padrão de integração serverless, com endpoints específicos no Next.js API Routes processando eventos do Stripe e atualizando o estado da aplicação conforme necessário.

4.2 Modelos de Machine Learning

4.2.1 API Deepgram

A API Deepgram foi integrada como solução de reconhecimento automático de fala (ASR) para gerar transcrições de alta qualidade dos vídeos. O modelo Nova-2 da Deepgram foi selecionado por seu equilíbrio entre precisão e velocidade, oferecendo transcrições com timestamps no nível da palavra e suporte a múltiplos idiomas. A implementação utilizou o SDK oficial do Deepgram para Python. Uma característica importante foi a capacidade de processar arquivos longos sem necessidade de segmentação, simplificando significativamente o fluxo de trabalho em comparação com a implementação anterior baseada em Whisper. O Deepgram também oferece recursos avançados como diarização (identificação de diferentes falantes), que embora não implementados na versão atual, estão planejados para futuras iterações da plataforma.

4.2.2 API GPT

A API GPT foi integrada através do Next.js, utilizando o Vercel AI SDK para streaming de respostas e gerenciamento eficiente de prompts. Para funcionalidades específicas como geração de capítulos e quizzes, foi utilizado o recurso de function calling da API, permitindo obter respostas estruturadas em formato JSON que puderam ser diretamente integradas ao fluxo da aplicação. O modelo GPT-4o foi selecionado como padrão para a maioria das operações devido à sua capacidade superior de compreensão contextual e geração de conteúdo educacional de alta qualidade, enquanto o GPT-4o-mini foi utilizado em operações menos complexas para otimização de custos.

4.2.3 Segment Anything (SaT)

O modelo **Segment Anything (SaT)** foi utilizado para segmentar as legendas extraídas de vídeos do **YouTube** em parágrafos, com o objetivo de melhorar a legibilidade e organização do texto transcreto. Essa segmentação estruturada facilitou a compreensão e a análise do conteúdo, tornando a experiência do usuário mais intuitiva e agradável.

Durante a fase de desenvolvimento, o modelo foi executado no backend utilizando **FastAPI**. No entanto, devido ao alto custo associado à manutenção de servidores com **GPU**, optou-se por migrar a execução para a infraestrutura **Beam Serverless GPU** em produção. Esse serviço permitiu a execução do modelo de forma escalável e sob demanda, reduzindo os custos operacionais sem comprometer significativamente o desempenho. A principal desvantagem foi uma pequena latência ocasionada pelos *cold starts*, mas isso foi minimizado devido a várias técnicas disponibilizadas pelo serviço.

4.3 Infraestrutura e Serviços em Nuvem

4.3.1 Open Next.js

Open Next.js otimizou o deploy da aplicação Next.js full-stack, garantindo performance adequada tanto para o frontend quanto para as API Routes do backend. Esta ferramenta de código aberto resolve desafios específicos de implantação do Next.js em ambientes serverless, como o gerenciamento de rotas dinâmicas, middleware e Server Components. Ao gerar uma estrutura de implantação otimizada para provedores de nuvem, o Open Next.js permitiu aproveitar os benefícios da arquitetura serverless (escalabilidade automática, baixo custo de manutenção e alta disponibilidade) sem sacrificar as funcionalidades avançadas do Next.js.

Um benefício significativo da utilização do Open Next.js foi a redução de custos operacionais. Como alternativa à hospedagem direta na Vercel, que exigiria um plano Pro de US\$ 20 mensais para aplicações SaaS com autenticação e funcionalidades comerciais, o Open Next.js permitiu implantar a aplicação em provedores de infraestrutura mais econômicos, mantendo a mesma qualidade e funcionalidades. Esta economia foi particularmente importante na fase inicial do projeto, quando a otimização de custos era crucial para viabilizar o desenvolvimento e testes da plataforma sem comprometer recursos financeiros limitados.

4.3.2 Google Cloud

O backend complementar em Python foi hospedado na Google Cloud Platform, fornecendo infraestrutura essencial para a aplicação. A implementação utilizou Cloud Run para containerização e implantação do serviço FastAPI, oferecendo escalabilidade automática e pagamento apenas pelo tempo de execução. Este modelo de escalabilidade automática foi particularmente valioso para um desenvolvedor solo, eliminando a necessidade de gerenciar manualmente a infraestrutura e permitindo foco exclusivo no desenvolvimento de funcionalidades. O Secret Manager do GCP foi utilizado para gerenciamento seguro de credenciais e chaves de API, garantindo que informações sensíveis não fossem expostas no código-fonte ou variáveis de ambiente não criptografadas. Outro benefício significativo foi o generoso programa de free trial do Google Cloud, que oferece US\$ 300 em créditos por 90 dias, permitindo testar e implementar a solução com custo inicial zero, aspecto crucial para viabilizar o desenvolvimento inicial do projeto.

4.3.3 Beam Serverless GPU

Beam Serverless GPU foi utilizado especificamente para o backend Python, fornecendo recursos de GPU para processamento de modelos de machine learning mais pesados. Esta plataforma especializada em computação serverless com

aceleração por hardware permitiu executar o modelo SaT de forma eficiente sem a necessidade de manter servidores GPU dedicados, resultando em economia significativa de custos. A implementação incluiu otimizações específicas para minimizar o impacto dos cold starts, como persistência do modelo em disco e pré-carregamento em memória compartilhada. O serviço também ofereceu monitoramento detalhado de utilização de recursos e latência, facilitando a identificação de gargalos e oportunidades de otimização. Um benefício significativo para a fase de validação do projeto foi o plano gratuito do Beam, que oferece 15 horas de uso de GPU sem custo, permitindo testar e refinar a implementação do modelo SaT em ambiente de produção sem investimento inicial.

4.3.4 Supabase

Supabase forneceu o banco de dados PostgreSQL para a aplicação, sendo acessado através do backend Next.js via Drizzle ORM. A plataforma oferece uma solução de banco de dados otimizada para ambientes serverless, com fácil configuração e manutenção. A combinação de Supabase, PostgreSQL e Drizzle ORM proporcionou uma base de dados robusta e tipo-segura, ideal para o desenvolvimento rápido da aplicação sem a necessidade de gerenciar infraestrutura de banco de dados complexa.

4.3.5 Sentry

Sentry foi implementado para monitoramento de erros e performance tanto no frontend Next.js quanto no backend Python. A ferramenta fornece alertas em tempo real e contexto detalhado sobre exceções, permitindo identificar e corrigir problemas rapidamente. Este sistema de monitoramento é importante para analisar o desempenho da aplicação, garantir sua estabilidade e melhorar continuamente a experiência do usuário.

4.4 Ferramentas de Desenvolvimento

4.4.1 Cursor

Cursor foi o IDE principal do projeto, proporcionando ganhos extraordinários de produtividade através de sua integração com modelos de linguagem avançados. Este editor potencializado por IA foi fundamental para o desenvolvimento full-stack, permitindo implementar rapidamente funcionalidades. O recurso de Composer Agêntico do Cursor possibilitou a escrita e modificação simultânea de múltiplos arquivos a partir de instruções em linguagem natural, facilitando a implementação de features que atravessam várias camadas da aplicação.

Particularmente valiosa foi a capacidade do Cursor de entender o contexto global do projeto, oferecendo refatorações inteligentes que consideravam as interdependências

entre diferentes componentes da aplicação. A ferramenta também se destacou na geração automática de código boilerplate, interfaces TypeScript, e na implementação de padrões complexos como autenticação, processamento de pagamentos e integração com APIs externas. Esta assistência baseada em IA reduziu drasticamente o tempo de desenvolvimento, permitindo que um desenvolvedor solo implementasse uma aplicação full-stack completa em uma fração do tempo que seria necessário com ferramentas tradicionais.

4.4.2 v0.dev

v0.dev simplificou significativamente o processo de desenvolvimento frontend, acelerando a criação de interfaces de usuário. Esta ferramenta de geração de UI baseada em IA converte descrições em linguagem natural em componentes React funcionais, compatíveis com o ecossistema ShadCN/Tailwind utilizado no projeto. O fluxo de trabalho consistiu em descrever a interface desejada, gerar o componente com v0.dev, e adaptá-lo às necessidades específicas da aplicação.

A combinação de Cursor e v0.dev proporcionou um ambiente de desenvolvimento eficiente, onde interfaces podiam ser prototipadas com v0.dev e posteriormente refinadas e integradas à lógica da aplicação com o Cursor. O resultado foi uma melhoria significativa na velocidade de desenvolvimento, especialmente valioso no contexto de um projeto desenvolvido por uma única pessoa.

4.5 Conclusão do capítulo

Este capítulo apresentou as tecnologias e ferramentas utilizadas no desenvolvimento do VideoLearnAI, destacando as escolhas técnicas que viabilizaram a implementação das funcionalidades propostas com performance e escalabilidade adequadas.

- **Stack de frontend moderno:** Next.js, Tailwind CSS e Shadcn UI proporcionaram uma base sólida para interfaces interativas e visualmente consistentes, sem comprometer a manutenibilidade do código.
- **Infraestrutura serverless:** Uma abordagem serverless foi adotada em todas as camadas da aplicação para maximizar a eficiência de custos. OpenNext.js permitiu implantar o código Next.js sem depender de plataformas comerciais como Vercel. Complementarmente, o Google Cloud Run foi utilizado para hospedar o backend especializado em Python/FastAPI, aproveitando créditos gratuitos (\$300 iniciais válidos por 3 meses). Para o processamento intensivo do modelo SAT, a infraestrutura GPU serverless permitiu acessar poder computacional avançado pagando apenas pelos segundos de processamento efetivamente utilizados. Esta estratégia completa de computação sob demanda viabilizou economicamente o

desenvolvimento do projeto por um desenvolvedor independente, eliminando a necessidade de investimentos significativos em infraestrutura antes da validação do conceito.

- **Integração estratégica de APIs:** A integração de serviços como Deepgram para transcrição e OpenAI para processamento de linguagem natural permitiu implementar as funcionalidades essenciais da plataforma de forma eficiente. Estas APIs forneceram as capacidades técnicas necessárias para transformar vídeos em experiências de aprendizagem interativas, sem adicionar complexidade desnecessária ao desenvolvimento.
- **Desenvolvimento potencializado por IA:** Ferramentas como Cursor e v0.dev transformaram fundamentalmente o processo de desenvolvimento, permitindo que um desenvolvedor solo implementasse funcionalidades complexas em tempo reduzido e acelerando significativamente a criação de interfaces.
- **Persistência de dados serverless:** O Supabase, uma plataforma PostgreSQL serverless, combinado com Drizzle ORM (otimizado para ambientes serverless com baixo overhead de inicialização), proporcionou uma solução de banco de dados alinhada à filosofia do projeto, eliminando a necessidade de gerenciar infraestrutura enquanto manteve os benefícios de um sistema relacional tipo-seguro.

O próximo capítulo detalhará como estas tecnologias foram aplicadas na implementação concreta das funcionalidades do VideoLearnAI, descrevendo os desafios encontrados e as soluções desenvolvidas.

5 DESENVOLVIMENTO DA APLICAÇÃO

5.1 Visão Geral da Arquitetura

A arquitetura do sistema VideoLearnAI foi projetada para ser escalável, modular e eficiente, utilizando tecnologias modernas para fornecer uma experiência de aprendizado aprimorada. A Figura 1 apresenta uma visão geral dos componentes e suas interações.

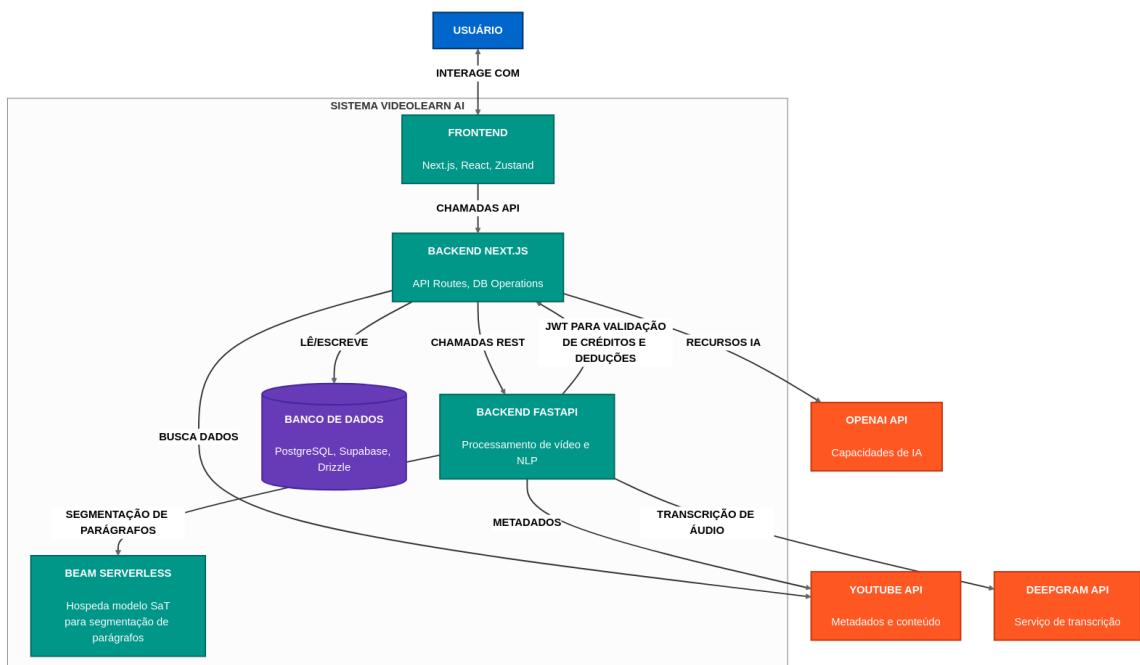


Figura 1 – Diagrama de arquitetura do sistema VideoLearnAI

5.1.1 Componentes Principais

5.1.1.1 Frontend

O frontend da aplicação é construído com Next.js, React e utiliza Zustand para gerenciamento de estado. Esta camada é responsável pela interface do usuário, permitindo interações intuitivas com o conteúdo de vídeo e os recursos de aprendizado.

5.1.1.2 Backend Next.js

O backend implementado com Next.js API Routes gerencia as operações de banco de dados e integrações com serviços externos. Este componente é o único com acesso direto ao banco de dados, centralizando a lógica de persistência e garantindo a integridade dos dados. Além disso, ele se comunica com APIs externas como YouTube e OpenAI para obter metadados de vídeos e recursos de IA, respectivamente.

5.1.1.3 Backend FastAPI

Desenvolvido em Python, essa camada é especializada em processamento e análise de conteúdo de vídeos do YouTube. Ele extrai metadados, baixa áudio, realiza transcrição automática via Deepgram API e segmenta o texto em parágrafos utilizando o serviço Beam. O sistema implementa autenticação JWT para validar usuários e gerencia créditos através de comunicação com o frontend, deduzindo-os conforme o tempo de vídeo processado. A arquitetura é modular, com rotas específicas para cada funcionalidade: obtenção de dados do vídeo, download de áudio, transcrição, segmentação de texto e pesquisa. O backend não acessa diretamente o banco de dados, mas se comunica com o frontend para validação de usuários e gerenciamento de créditos, garantindo controle de acesso e uso dos recursos.

5.1.1.4 Beam Serverless

Esta camada hospeda o modelo SaT (Segment Any Text), especializado na segmentação de parágrafos. O Beam Serverless permite a execução eficiente de modelos de IA sem a necessidade de gerenciar infraestrutura, oferecendo escalabilidade automática conforme a demanda e pagando apenas pelo tempo de execução.

5.1.1.5 Banco de Dados

O sistema utiliza PostgreSQL como banco de dados relacional, gerenciado através do Supabase e com esquema definido pelo Drizzle ORM. Esta combinação oferece robustez para armazenamento de dados estruturados, incluindo informações de usuários, conteúdo de vídeos, transcrições e outros dados essenciais para o funcionamento da plataforma.

5.1.2 Integrações Externas

- **YouTube API:** Fornece metadados e conteúdo de vídeos, permitindo que a plataforma acesse informações como título, descrição, duração e outros detalhes relevantes dos vídeos.
- **OpenAI API:** Oferece capacidades avançadas de IA para processamento de texto, sendo utilizada pelo backend Next.js para gerar conteúdo enriquecido, como

resumos, perguntas e respostas, e outros recursos educacionais baseados no conteúdo dos vídeos.

- **Deepgram API:** Serviço especializado em transcrição de áudio, utilizado pelo backend FastAPI para converter o áudio dos vídeos em texto, possibilitando a análise e processamento posterior do conteúdo.

5.1.3 Fluxos de Comunicação

Os principais fluxos de comunicação entre os componentes incluem:

1. **Interação do Usuário:** Os usuários interagem com o frontend da aplicação, que apresenta a interface e gerencia as interações.
2. **Processamento de Dados e Validação de Créditos:** O frontend faz chamadas API para o backend Next.js, que por sua vez pode fazer chamadas REST para o backend FastAPI quando necessário processamento especializado. A comunicação bidirecional entre os backends utiliza tokens JWT para autenticação e autorização, garantindo a segurança e integridade das requisições. Este mecanismo também permite que o backend FastAPI valide e deduza créditos dos usuários através do backend Next.js, assegurando que apenas usuários autorizados accessem recursos premium.
3. **Segmentação de Parágrafos:** O backend FastAPI se comunica com o Beam Serverless para realizar a segmentação de parágrafos utilizando o modelo SaT.
4. **Transcrição de Áudio:** O backend FastAPI utiliza o Deepgram API para transcrever o áudio dos vídeos em texto.
5. **Acesso a Dados:** O backend Next.js é responsável por ler e escrever no banco de dados PostgreSQL através do Supabase e Drizzle ORM.

Esta arquitetura modular permite que o sistema seja escalável, manutenível e extensível, facilitando a adição de novos recursos e a otimização de componentes existentes. A separação clara de responsabilidades entre os diferentes backends e a utilização de serviços especializados para tarefas específicas contribuem para a eficiência e robustez do sistema VideoLearnAI.

5.2 Melhoria da Legibilidade das Legendas

Para resolver o desafio de aprimorar a legibilidade das legendas, foram exploradas duas abordagens distintas. Inicialmente, testou-se uma solução baseada em LLMs (Large Language Models) combinada com um sistema de validação do resultado. Posteriormente, realizou-se uma segunda implementação, dessa vez utilizando o modelo SaT (Segment Any Text). Essa última abordagem se mostrou mais promissora.

5.2.1 O Problema da Legibilidade

As legendas de vídeos frequentemente apresentam problemas de formatação e segmentação, dificultando a compreensão do conteúdo. Esse problema ocorre porque as transcrições brutas costumam ser geradas como um fluxo contínuo de palavras, sem uma estrutura clara de frases e parágrafos. Não foram construídas tendo em mente a leitura da transcrição como um todo e sim apenas aparecer na tela no momento certo.

A disponibilização da transcrição com boa legibilidade, lado a lado com o conteúdo audiovisual traz diversos benefícios para o processo de aprendizagem. Primeiro, permite que o usuário rapidamente navegue pelo conteúdo, identificando e pulando seções menos relevantes para seu objetivo de estudo. Segundo, quando algum trecho do vídeo não foi bem compreendido, a possibilidade de reler a transcrição daquela parte específica oferece uma abordagem alternativa para entender o conteúdo. Por fim, a integração entre vídeo e transcrição oferece navegação bidirecional: além de realçar automaticamente o texto conforme o vídeo progride, permite saltar para qualquer momento do vídeo com um clique na transcrição, tornando a experiência do usuário mais fluida e interativa.

Na aplicação desenvolvida, o foco foi especificamente na melhoria da segmentação do texto em frases e parágrafos. Embora existam modelos promissores para adicionar pontuação adequada ao texto, como em (Guhr; Schumann; Bahrmann; Böhme, 2021) que utiliza transformers multilíngues para essa tarefa obtendo F1-score médio de 0,94 para detecção de final de sentença em textos em inglês, alemão, francês e italiano, essa funcionalidade será explorada em uma iteração futura do projeto.

5.2.2 Primeira Abordagem com LLMs

Inicialmente, foi utilizado um modelo de linguagem de grande porte (LLM) para segmentar e melhorar a legibilidade do texto das legendas. A implementação foi feita utilizando a biblioteca `instructor`, que permite a validação do texto gerado através dos modelos de tipagem do `pydantic`, garantindo conformidade com um formato estruturado.

Para evitar alterações indesejadas no conteúdo original, no modelo Pydantic utilizado, adicionou-se o decorador `@field_validator` para executar uma função de validação customizada. Esta função utilizava a normalização de texto para checar se apenas espaços e pontuações foram modificados entre o texto processado e o original.

Quando a validação falhava, a biblioteca `instructor` automaticamente incluía no próximo prompt o texto original, a saída que falhou e o erro específico da validação, facilitando que a LLM corrigisse seus erros. Este processo se repetia automaticamente por até 5 tentativas (definido por `max_retries=5`) para cada texto processado.

Entretanto, essa abordagem apresentou desafios significativos:

- **Latência elevada:** O processo todo acaba sendo muito demorado pois necessita de várias interações devido a limitação da janela de contexto. Tornando a solução pouco eficiente para processar legendas de vídeos de longa duração
- **Custo financeiro elevado:** O uso de LLMs para processamento de texto em larga escala mostrou-se financeiramente custoso, considerando o volume de requisições necessárias.
- **Alterações não desejadas no texto:** Apesar das instruções explícitas no prompt para evitar adições ou remoções de palavras, o modelo ocasionalmente incluía frases como "Aqui está o seu texto com legibilidade aprimorada" ou alterava partes do conteúdo original, omissão de algumas palavras ou frases por exemplo.
- **Falhas no processo:** Mesmo com o mecanismo de validação e o sistema de retentativas implementados, havia casos em que o processo esgotava o número máximo de iterações sem terminar sem erros.

Esses fatores tornaram a abordagem com LLMs inviável para o problema proposto.

5.2.3 Implementação com SaT (Segment Anything Text)

Diante das limitações arquiteturais dos LLMs, discutidas na seção 3.1.5 sobre limitações fundamentais dos modelos de linguagem, foi adotado o SAT (*Segment Any Text*) (Frohmann; Sterner; Vulic; Minixhofer; Schedl, 2024), um transformer especializado para segmentação de texto que oferece uma solução mais robusta e eficiente para a melhoria da legibilidade das legendas. Diferente dos LLMs que focam em geração de texto, o SAT utiliza uma arquitetura de classificação binária que apenas identifica os limites entre sentenças, apresentando as seguintes vantagens:

- **Baixa latência:** A segmentação ocorre de maneira rápida e eficiente graças à sua arquitetura otimizada com tokenização por subpalavras com apenas 3 camadas transformer, resultando em um modelo muito menor e especializado para a tarefa.
- **Preservação do conteúdo original:** O modelo não gera ou modifica texto, apenas classifica as posições como limites de sentença ou não. Esta abordagem focada em classificação binária garante tanto a fidelidade ao texto original quanto maior confiabilidade e previsibilidade no processamento das legendas.
- **Escalabilidade:** Por ser um modelo mais leve, o SAT pode processar textos longos com menor consumo de recursos computacionais, tornando-o ideal para aplicações em produção com restrições de custo e latência.
- **Precisão:** Apesar de sua simplicidade relativa, o modelo demonstrou alta precisão na identificação de limites de parágrafos, resultando em textos bem estruturados e de fácil leitura.

A implementação do SAT no VideoLearnAI foi realizada através de um serviço dedicado no backend Python (FastAPI), que se comunica com a infraestrutura Beam Serverless GPU para processamento eficiente. O fluxo operacional envolve o recebimento da legenda bruta pelo FastAPI, remoção das informações de tempo (timestamps) para preparar o texto puro, envio ao Beam Serverless para execução do modelo SAT, identificação dos limites de parágrafos, retorno ao FastAPI para reintegração das informações temporais originais ao texto já segmentado, e finalmente formatação do texto estruturado para apresentação ao usuário. Esta arquitetura preserva a sincronização precisa entre o texto segmentado e o vídeo, permitindo que o usuário navegue pelo conteúdo audiovisual através da transcrição, resultando em uma melhoria significativa na legibilidade sem perder a funcionalidade de navegação temporal.

5.2.4 Interface e Experiência do Usuário

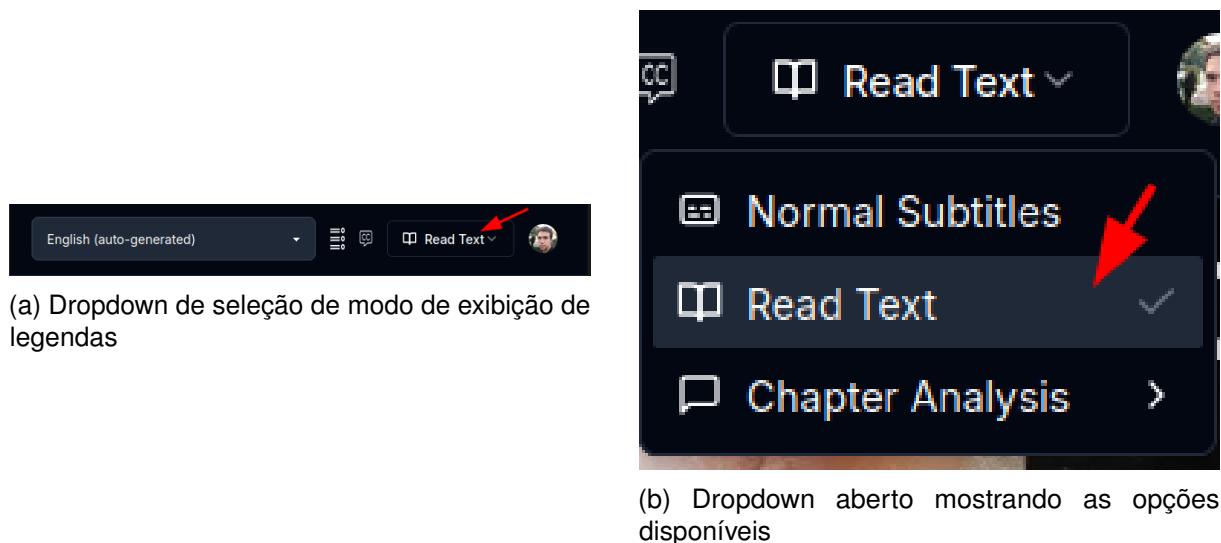
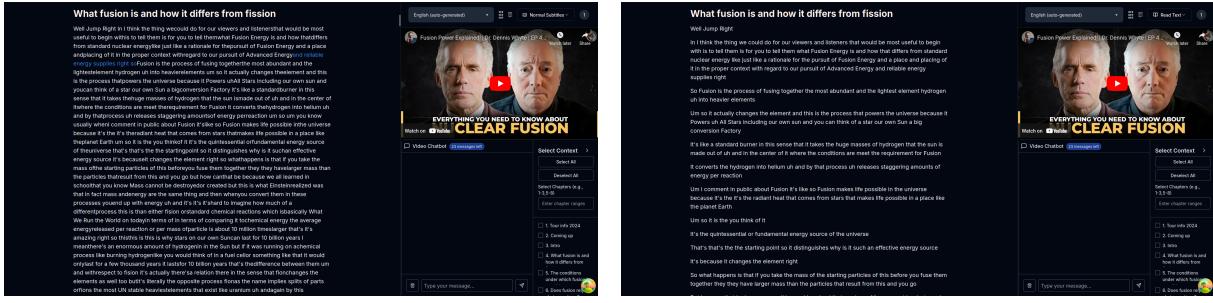


Figura 2 – Interface para seleção do modo de leitura da transcrição

Para melhorar a legibilidade do texto das legendas, o usuário pode utilizar o dropdown de seleção mostrado na Figura 2. Ao clicar neste elemento de interface (Figura 2a), são apresentadas duas opções: "Normal Subtitle" e "Read Text" (Figura 2b). Ao selecionar a opção "Read Text", o sistema inicia o processamento para melhorar a legibilidade das legendas, exibindo um toast informativo que indica que o processo está em andamento.

O sistema foi projetado para otimizar a experiência do usuário através do armazenamento de resultados prévios. Quando um vídeo já teve suas legendas processadas anteriormente, o resultado é recuperado automaticamente do banco de dados e o dropdown é inicializado com a opção "Read Text" selecionada. Caso contrário, a opção padrão "Normal Subtitle" é apresentada, permitindo que o usuário solicite o processamento quando desejar.



A Figura 3 demonstra o impacto visual da melhoria de legibilidade proporcionada pelo modelo SAT. À esquerda (Figura 3a), observa-se a transcrição original com texto contínuo e sem estruturação clara, dificultando a leitura e compreensão. À direita (Figura 3b), o mesmo conteúdo após o processamento apresenta uma organização em parágrafos lógicos, com espaçamento adequado e estrutura visual que facilita o acompanhamento do conteúdo.

O processamento é realizado de forma eficiente pela infraestrutura Beam Serverless GPU, como evidenciado pelos logs de execução na Figura 4. Nota-se que, embora exista um tempo inicial de carregamento (cold start) de aproximadamente 18 segundos na primeira requisição, as requisições subsequentes são processadas em apenas 50-100ms. Esta otimização de performance é crucial para a experiência do usuário, garantindo que, após o carregamento inicial, a interação com o sistema seja fluida e responsiva, mesmo ao processar vídeos de longa duração.

Request ID	Path	Status	Start Time	End Time	Latency	Throughput
5b764967-b9de-4cb1-9d4a-14b8eae62aa4	endpoint/deployment/script:segment_text	Complete	2/5/25 19:07	2/5/25 19:07	103ms	42ms
d4e3b209-398c-4ee6-97d1-e294775fdd4	endpoint/deployment/script:segment_text	Complete	2/5/25 19:07	2/5/25 19:07	51ms	50ms
1f470137-e99-4567-ao1-8aa468a3c0b6	endpoint/deployment/script:segment_text	Complete	2/5/25 19:07	2/5/25 19:07	104ms	45ms
dd4d16ea-c5d4-42b3-9e11-a91096cf568	endpoint/deployment/script:segment_text	Complete	2/5/25 19:07	2/5/25 19:07	61ms	48ms
b562e123-d34-4c20-a1f4-1af5ef58d22	endpoint/deployment/script:segment_text	Complete	2/5/25 19:07	2/5/25 19:07	18s 573ms	350ms

Figura 4 – Logs de execução no Beam Serverless GPU demonstrando o tempo de cold start (18s 573ms) na primeira requisição e tempos de resposta reduzidos (aproximadamente 50-100ms) nas requisições subsequentes

5.3 Geração de Capítulos

A geração automática de capítulos para vídeos do YouTube facilita a navegação e compreensão do conteúdo, organizando transcrições longas em seções temáticas. Esta funcionalidade permite que os usuários localizem rapidamente informações específicas e compreendam melhor a estrutura do material apresentado.

5.3.1 Arquitetura do Sistema de Geração de Capítulos

O sistema implementado no VideoLearnAI utiliza uma abordagem em duas etapas que combina segmentação de texto e processamento de linguagem natural para identificar transições temáticas em transcrições de vídeo:

1. **Segmentação de Parágrafos com o Modelo SAT:** Inicialmente, a transcrição completa do vídeo é processada pelo modelo SAT (Segment Any Text), que identifica fronteiras naturais de parágrafos baseando-se no contexto semântico.
2. **Identificação de Transições Temáticas via LLM:** Os parágrafos segmentados são então analisados por um modelo de linguagem grande (LLM) que identifica mudanças significativas de tópico e gera títulos descritivos para cada seção.

Esta abordagem híbrida aproveita as forças complementares de cada modelo: a precisão do SAT na segmentação estrutural do texto e a capacidade do LLM de compreender contexto semântico e identificar transições temáticas.

5.3.2 Fluxo de Processamento

O processo de geração de capítulos segue um fluxo bem definido:

1. **Estruturação do texto:** A transcrição do vídeo é segmentada em parágrafos coerentes utilizando o modelo SAT, que comprehende o contexto semântico e identifica fronteiras naturais no texto, mesmo em transcrições que carecem de pontuação adequada.
2. **Numeração dos parágrafos:** Cada parágrafo recebe um identificador numérico sequencial e é associado ao seu timestamp de início correspondente. Esta associação é fundamental para o mapeamento posterior entre os capítulos identificados e os momentos específicos do vídeo.
3. **Identificação de transições temáticas:** O texto segmentado e numerado é enviado para um LLM (especificamente o GPT-4o) através de uma prompt cuidadosamente elaborada. A prompt instrui o modelo a identificar onde novos capítulos devem começar e a gerar títulos descritivos para cada seção.
4. **Transformação em capítulos sincronizados:** Os tópicos identificados pelo LLM são transformados em capítulos sincronizados com o vídeo, mapeando os números de parágrafo para os timestamps correspondentes. Cada capítulo contém um título, tempos de início e término, e um estado de ativação para indicar se o capítulo está ativo ou não na interface.

5.3.3 Integração com LLMs via Function Calling

Para a identificação das transições de tópico, o sistema utiliza function calling para interagir com o LLM. Esta abordagem estruturada permite enviar a transcrição segmentada e numerada, solicitando ao modelo que retorne uma lista de objetos representando os pontos de mudança mais relevantes.

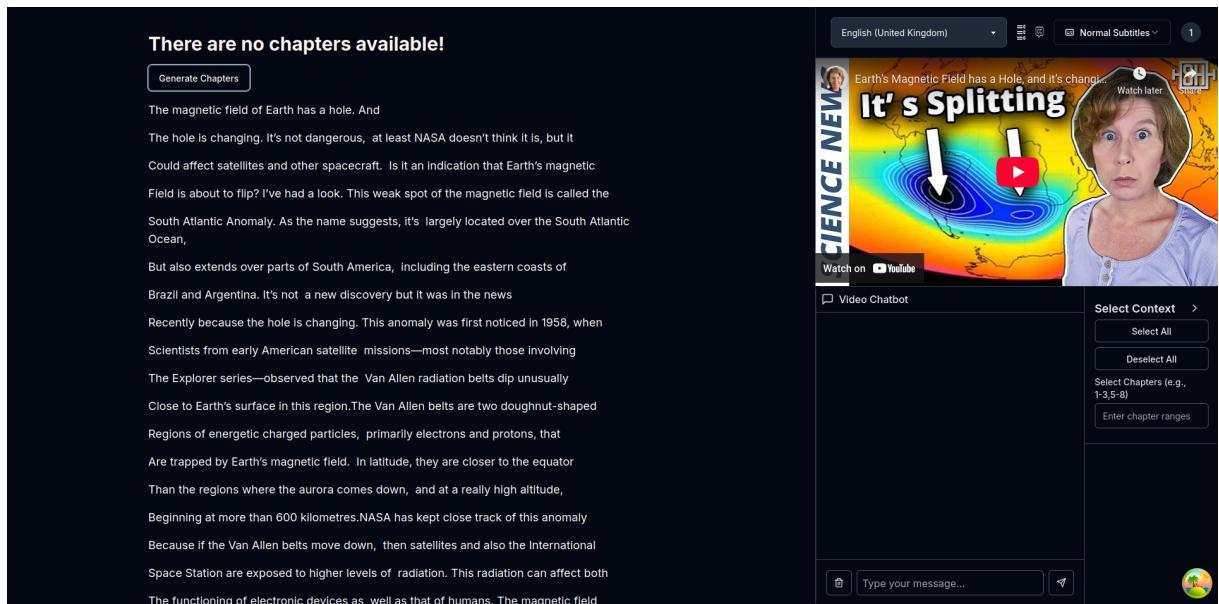


Figura 5 – Interface para iniciar o processo de geração automática de capítulos, mostrando o botão "Generate Chapters" que aciona o processamento

Como mostrado na Figura 5, o usuário pode iniciar o processo de geração de capítulos através de um botão dedicado na interface. Ao clicar neste botão, o sistema inicia o processamento da transcrição, aplicando primeiro a segmentação de parágrafos e em seguida a análise semântica para identificação de tópicos.

A função chamada retorna uma lista estruturada de tópicos, cada um contendo um título descritivo e o número do parágrafo onde o capítulo começa. Esta abordagem estruturada facilita o processamento posterior e a sincronização com o vídeo.

Inicialmente, foi testada uma abordagem em que o LLM era solicitado a identificar diretamente os tópicos e seus timestamps correspondentes. No entanto, esta tarefa mostrou-se excessivamente complexa para o modelo, resultando em frequentes imprecisões temporais. A solução implementada simplifica significativamente o problema: em vez de exigir que o LLM compreenda e gere informações temporais precisas, o sistema solicita apenas a identificação dos números dos parágrafos onde ocorrem transições temáticas.

Esta estratégia reduz a complexidade cognitiva para o LLM, permitindo que ele se concentre exclusivamente na análise semântica do conteúdo - tarefa para a qual é naturalmente otimizado. O mapeamento entre parágrafos e timestamps é realizado posteriormente pelo sistema, utilizando as associações temporais já estabelecidas

durante a segmentação inicial.

The South Atlantic Anomaly

The magnetic field of Earth has a hole.

And the hole is changing, at least NASA doesn't think it is, but it could affect satellites and other spacecraft.

Is it an indication that Earth's magnetic field is about to flip?

I've had a look. This weak spot of the magnetic field is called the South Atlantic Anomaly.

As the name suggests, it's largely located over the South Atlantic Ocean, but also extends over parts of South America, including the eastern coasts of Brazil and Argentina.

It's not a new discovery but it was in the news recently because the hole is changing.

This anomaly was first noticed in 1958, when scientists from early American satellite missions—most notably those involving the Explorer series—observed that the Van Allen radiation belts dip unusually close to Earth's surface in this region.

The Van Allen belts are two doughnut-shaped regions of energetic charged particles, primarily electrons and protons, that are trapped by Earth's magnetic field.

In latitude, they are closer to the equator than the regions where the aurora comes down, and at a really high altitude, beginning at more than 600 kilometres.

NASA has kept close track of this anomaly because if the Van Allen belts move down, then satellites and also the International Space Station are exposed to higher levels of radiation.

This radiation can affect both the functioning of electronic devices as well as that of humans.

The magnetic field anomaly however doesn't affect aircraft, because these fly at much lower altitudes.

This hole in the magnetic field seems to be splitting apart into two distinct patches.

A few years ago, Scientists from NASA's Goddard Space Flight Centre used data from the European Space Agency's Swarm satellite missions and the CubeSat ELFIN mission to track how the hole has been developing.

SCIENCE NEWS

It's Splitting

Earth's Magnetic Field has a Hole, and it's changing... Watch later Share

Watch on YouTube

Video Chatbot

Select Context >

- Select All
- Deselect All
- Select Chapters (e.g., 1-3, 5-8)
- Enter chapter ranges

- 1. The South Atlantic Anomaly
- 2. Magnetic Field Reversals
- 3. The Importance of Earth's Magnetic Field
- 4. Brilliant's Science Courses

Type your message...

Figura 6 – Interface após a geração de capítulos, os capítulos são listados no contexto do Chat e também na tabela de conteúdos que será apresentada na próxima seção

O resultado do processo de geração de capítulos é ilustrado na Figura 6, onde se pode observar a interface atualizada com a lista de capítulos gerados.

5.3.4 Tabela de Conteúdos e Histórico de Visualização

Os capítulos gerados automaticamente servem como fundamento para recursos adicionais que enriquecem a experiência de aprendizagem. Dois desses recursos são particularmente importantes: o Sumário de Conteúdo e o Histórico de Visualização com Progresso.

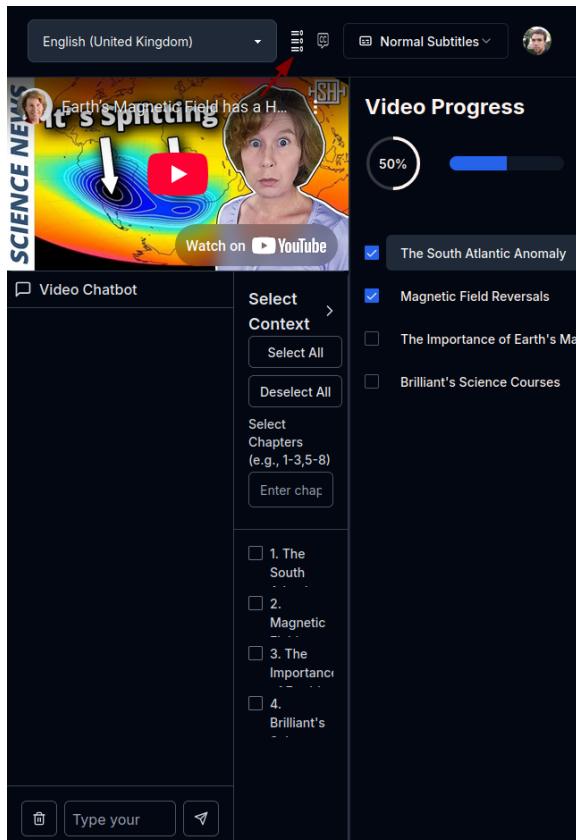


Figura 7 – Interface do Sumário de Conteúdo, mostrando a lista de capítulos com indicadores de progresso e opções para marcar capítulos como concluídos

Como ilustrado na Figura 7, o sistema oferece um Sumário de Conteúdo acessível através de um botão localizado ao lado do seletor de transcrição. Este recurso apresenta uma visão estruturada de todos os capítulos do vídeo, permitindo que o usuário:

- Navegue diretamente para qualquer capítulo com um clique, alterando automaticamente a posição do vídeo e o scroll da transcrição
- Marque capítulos individuais como concluídos, facilitando o acompanhamento do progresso de estudo
- Visualize o percentual total de conclusão do vídeo, baseado nos capítulos marcados como concluídos

Esta funcionalidade é particularmente útil para vídeos longos, permitindo que o usuário acompanhe seu progresso e retome o estudo de forma organizada. Vale ressaltar que o sistema funciona tanto com capítulos nativos do YouTube quanto com capítulos gerados automaticamente pela plataforma, garantindo sua utilidade para qualquer tipo de conteúdo.

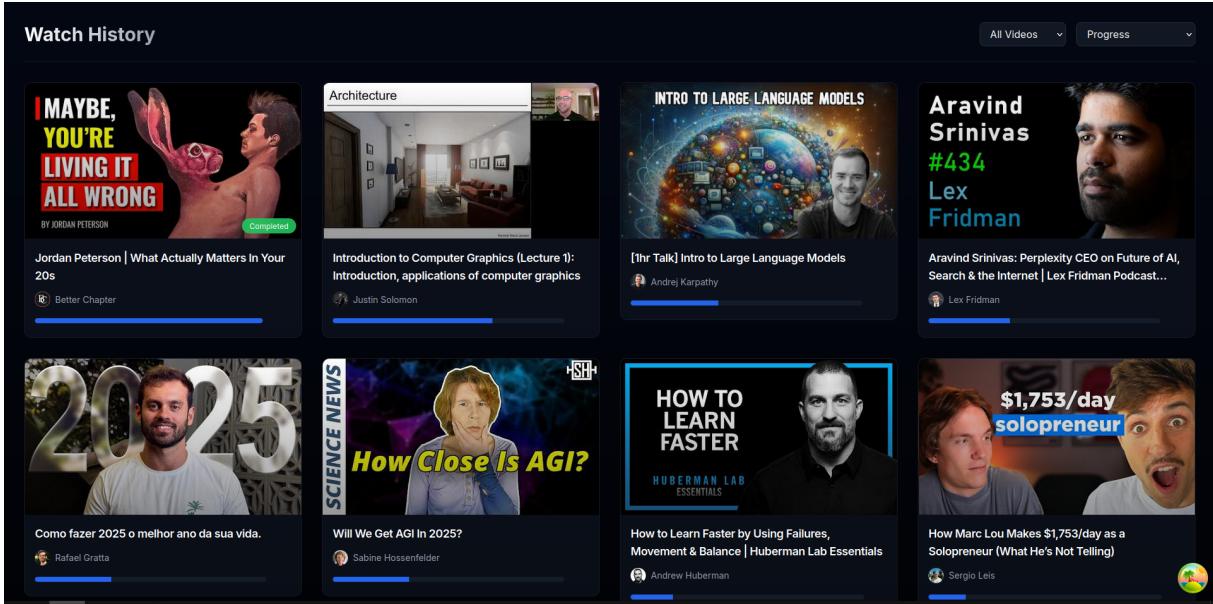


Figura 8 – Tela de Histórico de Visualização, exibindo os vídeos assistidos pelo usuário com seus respectivos percentuais de progresso

Complementando o Sumário de Conteúdo, o sistema também oferece uma visão consolidada do progresso através da tela de Histórico de Visualização, mostrada na Figura 8. Esta interface apresenta todos os vídeos que o usuário começou a assistir, junto com:

- O percentual de progresso para cada vídeo, calculado com base nos capítulos marcados como concluídos
- Informações básicas sobre cada vídeo, como título e miniatura
- Acesso direto para retomar a visualização de qualquer vídeo

Este recurso facilita o gerenciamento de múltiplos vídeos de estudo, eliminando a necessidade de lembrar manualmente onde o usuário parou em cada conteúdo e proporcionando uma visão clara de seu progresso geral.

A implementação destas funcionalidades demonstra como a estruturação do conteúdo em capítulos, sejam eles nativos ou gerados automaticamente, pode melhorar significativamente a experiência de aprendizagem com vídeos.

5.4 Geração de Transcrição

A transcrição precisa de vídeos do YouTube é essencial para gerar textos mais confiáveis do que as legendas automáticas, que apresentam qualidade inferior especialmente para línguas que não são o inglês.

5.4.1 Primeira implementação com Whisper

No fluxo original utilizando o Whisper, o processo consistia em:

- 1. Download do áudio** (comum a ambos os métodos):

- Uso da biblioteca `pytubefix` para download
- Progresso reportado ao usuário via SSE (Server-Sent Events)
- Arquivo de áudio com qualidade balanceada entre precisão e velocidade

- 2. Pré-processamento específico para Whisper:**

- Divisão do áudio em segmentos de 10 minutos
- Cortes preferencialmente em momentos de silêncio
- Envio paralelo dos segmentos para transcrição
- Necessidade decorrente das limitações do Whisper com arquivos longos

5.4.2 Migração para o Deepgram

Com a adoção do Deepgram, o fluxo foi otimizado:

- 1. Download do áudio mantido:**

- Mesmo processo via `pytubefix` com feedback via SSE
- Eliminação da etapa de divisão do áudio

- 2. Transcrição direta:**

- Envio do arquivo de áudio completo
- Processamento único sem necessidade de paralelização
- Interface exibe estimativa de tempo restante
- Capacidade nativa de lidar com longas durações

5.4.3 Interface e Experiência do Usuário

O processo de transcrição começa quando o usuário clica no botão "CC" localizado na interface principal, como mostrado na Figura 9.



Figura 9 – Botão "CC" para iniciar o processo de transcrição

Ao clicar neste botão, é exibido um modal de confirmação que apresenta informações importantes para o usuário: a quantidade de créditos disponíveis em sua conta e a duração do vídeo a ser processado, como ilustrado na Figura 10.

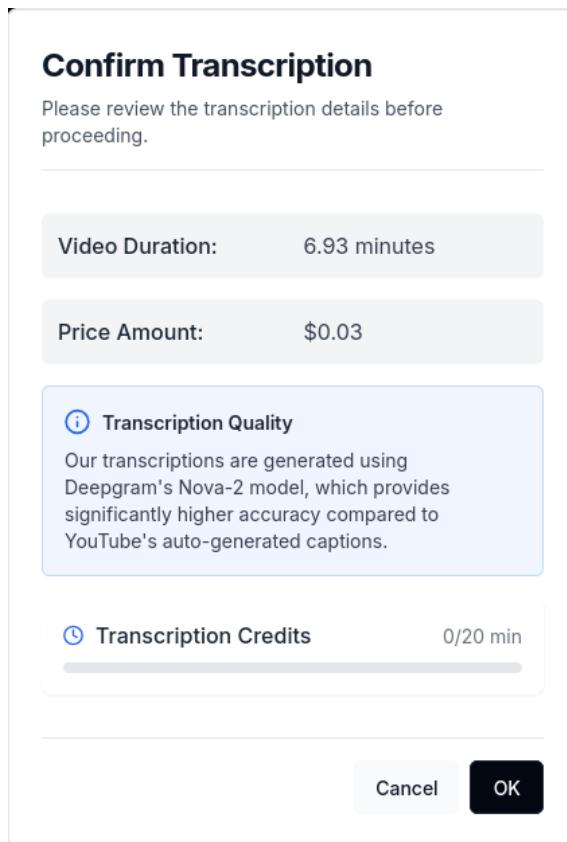


Figura 10 – Modal de confirmação mostrando créditos disponíveis e duração do vídeo

Após o usuário confirmar a operação clicando em "OK", o sistema inicia uma série de etapas de processamento. Primeiramente, ocorre a validação da duração do vídeo, onde o backend Next.js verifica a duração real através da API do YouTube. Esta verificação de segurança garante que a informação de duração não tenha sido adulterada no frontend, como mostrado na Figura 11.

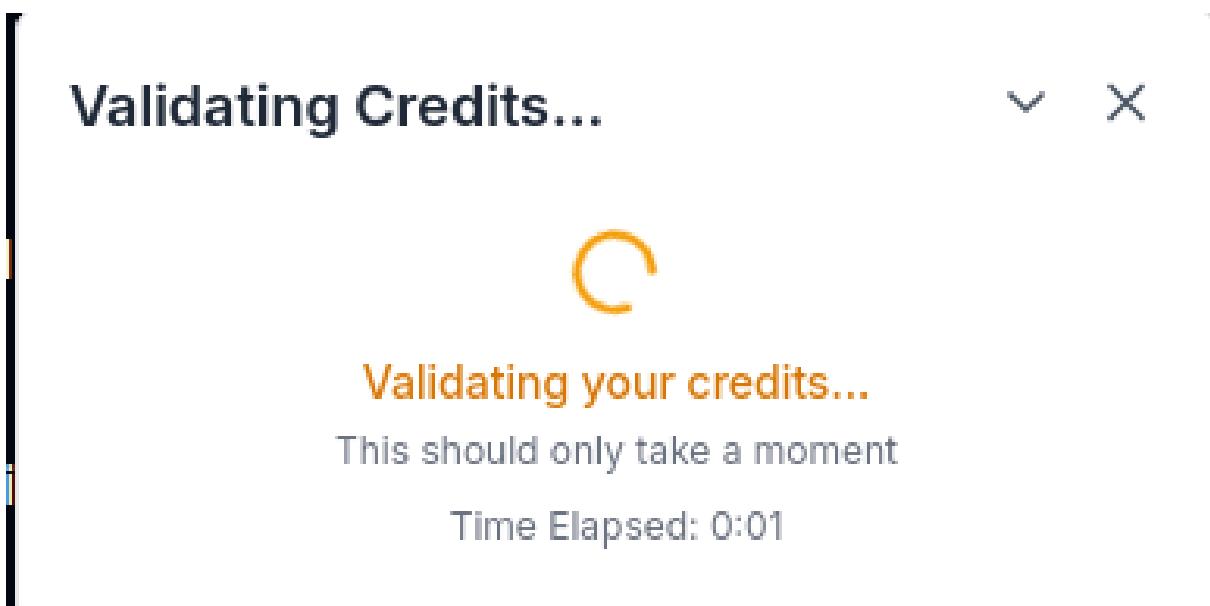


Figura 11 – Etapa de validação da duração do vídeo via API do YouTube

Em seguida, o sistema inicia o download do áudio do vídeo. Durante esta etapa, o progresso é exibido em tempo real com atualizações percentuais, permitindo que o usuário acompanhe o andamento do processo, como ilustrado na Figura 12.



Figura 12 – Progresso do download do áudio do vídeo com atualização percentual

Após o download completo do áudio, inicia-se o processo de transcrição propriamente dito. Diferentemente da etapa de download, a API do Deepgram não fornece atualizações percentuais do progresso. No entanto, o sistema apresenta uma estimativa de tempo restante baseada na velocidade de processamento informada pelo modelo Nova-2 do Deepgram, como mostrado na Figura 13.

Generating Transcription...

Time Elapsed: 0:01

Estimated time: 1.04 seconds for this video duration.

Figura 13 – Processo de transcrição em andamento com estimativa de tempo restante

Após a conclusão do processo, o sistema exibe uma tela de confirmação indicando que a transcrição foi finalizada com sucesso, como ilustrado na Figura 14.

Transcription Complete!

Window will close in 3 seconds...

Figura 14 – Tela de conclusão do processo de transcrição

Após a conclusão bem-sucedida do processo de transcrição, a interface é atualizada automaticamente para exibir o texto transcritado, e o seletor de transcrição adiciona no dropdown o grupo AI transcription e adiciona um item chamado de Transcription. Esse elemento, a transcrição gerada é automaticamente selecionada.

5.5 Geração de Quizzes

A geração de quizzes para cada capítulo representa uma aplicação direta dos princípios de aprendizagem ativa discutidos na fundamentação teórica. Como visto anteriormente, o ato de testar-se durante o processo de estudo ativa o mecanismo de prática de recuperação, que fortalece significativamente a retenção de longo prazo. Esta funcionalidade transforma momentos de consumo passivo em oportunidades de engajamento cognitivo, permitindo que o usuário verifique sua compreensão do conteúdo antes de avançar para a próxima seção.

Foram desenvolvidas duas interfaces gráficas distintas para os quizzes, adaptadas a diferentes tipos de perguntas:

- Interface para questões de *Verdadeiro ou Falso*.
- Interface para questões do tipo *Pergunta e Resposta (Q&A)*.

Em ambos os casos, a interação com o quiz é implementada de forma dinâmica através do Vercel AI SDK. Esta biblioteca permite que a interface do usuário seja atualizada progressivamente, exibindo as perguntas e respostas à medida que são geradas pela Large Language Model (LLM), sem necessidade de aguardar o processamento completo. Esta abordagem de streaming melhora significativamente a experiência do usuário (UX), reduzindo a percepção de tempo de espera e proporcionando feedback visual imediato durante o processo de geração.

Para garantir a compatibilidade entre as respostas geradas pela LLM e as interfaces gráficas, o sistema utiliza o mecanismo de function calling. Esta técnica, detalhada na fundamentação teórica, permite definir um schema estruturado que a LLM deve seguir ao gerar seu output. Ao especificar o formato exato esperado para as questões e respostas, o sistema consegue processar e renderizar corretamente os elementos de interface em tempo real, eliminando problemas de parsing e garantindo consistência na apresentação dos quizzes.

Além disso, para ambos os tipos de quiz, o usuário tem a opção de personalizar o comportamento da LLM por meio da edição do prompt específico. Esta personalização é acessível através do botão de três pontos na interface do quiz, permitindo ajustes nas diretrizes de geração das questões. O usuário pode, por exemplo, solicitar questões mais desafiadoras, focar em conceitos específicos ou alterar o estilo das perguntas. Esta capacidade de customização torna o sistema adaptável a diferentes objetivos pedagógicos e estilos de aprendizagem, aumentando sua utilidade como ferramenta educacional.

5.5.1 Questões discursivas

Question	Answer	AI Answer
What is Fusion Energy?		👁️ 刪 Fusion Energy is the process of fusing together hydrogen atoms to form heavier elements, releasing a significant amount of energy in the process. It powers stars, including our sun, and is considered the fundamental energy source of the universe.
How does Fusion Energy differ from standard nuclear energy?		👁️ 刪 Fusion Energy differs from
Why is the pursuit of Fusion Energy important?		👁️ 刪 The pursuit of Fusion Energy
What are the advantages of Fusion Energy over chemical energy?		👁️ 刪 Fusion Energy releases about
What is the significance of Einstein's theory in understanding Fusion Energy?		👁️ 刪 Einstein's theory highlights
+		

Figura 15 – Exemplo de interface para perguntas e respostas.

A interface de perguntas e respostas (Q&A) apresenta uma tabela com três colunas: a primeira exibe a pergunta gerada, a segunda, chamada “Resposta”, permite ao usuário inserir sua resposta manualmente, e a terceira, “AI Answer”, que é inicialmente oculta por um efeito de *blur*. A visibilidade dessa coluna pode ser alternada através de um botão, permitindo que o usuário visualize ou oculte a resposta gerada pela IA. O objetivo desse design é incentivar o usuário a tentar responder antes de consultar a resposta fornecida pela IA, promovendo a aprendizagem ativa.

5.5.2 Questões de Verdadeiro ou Falso

The screenshot shows a dark-themed quiz interface. At the top left, there is a battery icon with the text "5 credits remaining". Below it, a section titled "True or False" is shown with a downward arrow icon. The first question is: "Fusion energy is the process of fusing hydrogen into heavier elements, which powers the universe." Below the question are two buttons: "True" and "False". To the right of the buttons, the word "Correct" is displayed in green. The second question is: "Fusion energy is the same as fission energy." Below the question are two buttons: "True" and "False". To the right of the buttons, the word "Correct" is displayed in green. The third question is: "The energy released from fusion reactions is significantly greater than that from chemical reactions." Below the question are two buttons: "True" and "False". To the right of the buttons, the word "Incorrect" is displayed in red, followed by the explanatory text: "Incorrect. The energy released per reaction in fusion is about 10 million times larger than that from standard chemical reactions.". The fourth question is: "The sun can last for billions of years because it runs on chemical processes." Below the question are two buttons: "True" and "False". To the right of the buttons, the word "Incorrect" is displayed in red, followed by the explanatory text: "Incorrect. The sun lasts for billions of years because it runs on fusion processes, not chemical processes.". The fifth question is: "Harnessing fusion energy on Earth would have consequences similar to those of fossil fuel energy." Below the question are two buttons: "True" and "False". To the right of the buttons, the word "Incorrect" is displayed in red, followed by the explanatory text: "Incorrect. Harnessing fusion energy would have very different consequences compared to fossil fuel energy, as it is a fundamentally different energy source.".

Figura 16 – Exemplo de interface para questões de verdadeiro ou falso.

Na interface de Verdadeiro ou Falso, as perguntas são apresentadas como afirmações textuais, e abaixo de cada uma há dois botões que permitem ao usuário marcar a resposta como verdadeira ou falsa. Após a seleção um texto adicional surge abaixo, indicando o resultado da resposta: caso esteja correta, a confirmação aparece em verde; caso esteja errada, um texto em vermelho explica o motivo do erro. Esse feedback imediato auxilia na compreensão dos conceitos e melhora a fixação do aprendizado.

5.5.3 Geração e Personalização de Quizzes

Uma característica distintiva do sistema de quizzes implementado é a possibilidade de personalização dos prompts utilizados para gerar as questões. Esta funcionalidade permite que o usuário adapte o comportamento do modelo de linguagem às suas necessidades específicas de aprendizado, seja alterando o estilo, a dificuldade ou mesmo o idioma das questões geradas.

O fluxo de interação para geração e customização de quizzes inicia-se ao final de cada capítulo, onde um botão é apresentado, como ilustrado na Figura 17.

Yeah. - And so you have to figure out how much more instances to buy, those kinds of things, you have to- -

Yeah, that's the kind of problems you need to solve.

Like whether you wanna, like, keep...

You know, it's a whole reason it's called Elastic.

Some these things can be scaled very gracefully, but other things so much not, like GPUs or models, like you need to still, like, make decisions on a discreet basis.



1 million H100 GPUs

- You tweeted a poll asking "Who's likely to build the first one million H100 GPU-equivalent data center?

Figura 17 – Botão ao final de cada capítulo, oferecendo diferentes opções de interação com o conteúdo

Ao clicar neste botão, o usuário é apresentado a um menu com diferentes opções para interagir com o conteúdo do capítulo. As opções padrão incluem a geração de questões de Verdadeiro ou Falso e perguntas discursivas (Q&A). Além destas, há uma terceira opção chamada "Custom Text Prompt", que diferentemente das anteriores, não renderiza uma interface específica de quiz, mas processa o texto do capítulo e retorna o resultado como texto simples. Por padrão, esta opção realiza uma sumarização do conteúdo, mas pode ser personalizada para executar qualquer tipo de transformação textual.

Cada uma dessas opções possui dois elementos: um botão principal com o nome da funcionalidade, que ao ser clicado gera imediatamente o resultado usando o prompt padrão, e um botão de três pontos adjacente que oferece possibilidade de customização do prompt, como mostrado na Figura 18.

Some these things can be scaled very gracefully, but other things so much not, like GPUs or models, like you need to still, like, make decisions on a discreet basis.



True or False



Q&A Table



Custom text prompt



1 million H100 GPUs

- You tweeted a poll asking "Who's likely to build the first one million H100 GPU-equivalent data center?

Figura 18 – Botão de três pontos para acesso a customização do prompt, nesse caso específico para a opção de geração de questões de Verdadeiro ou Falso

Ao clicar no botão de três pontos ao lado da opção desejada (Verdadeiro ou Falso, Q&A ou Custom Text Prompt), um popover é exibido contendo um campo de texto onde

o usuário pode inserir instruções personalizadas para o modelo de linguagem, como ilustrado na Figura 19.

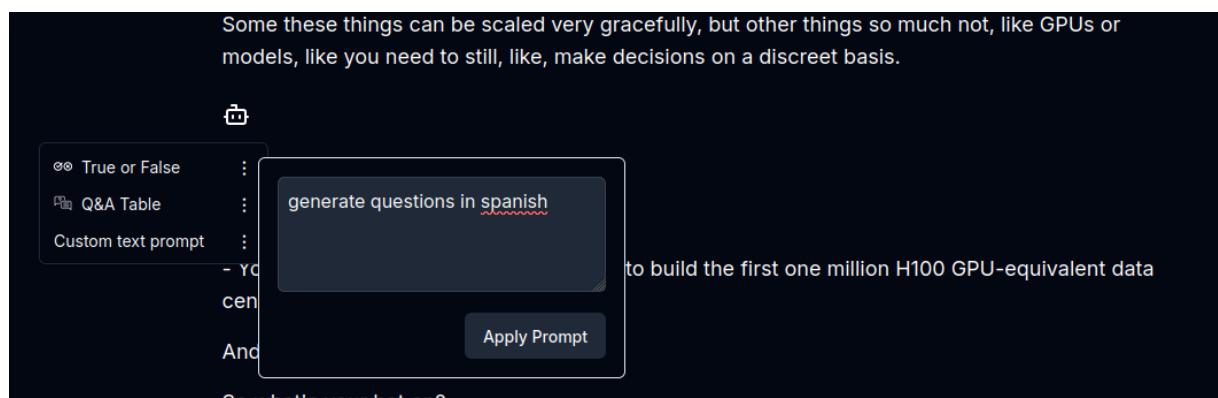


Figura 19 – Popover para customização do prompt utilizado na geração de questões

Este campo permite personalizar o comportamento do modelo. Após inserir o prompt personalizado e confirmar, o sistema gera as questões seguindo as diretrizes especificadas.

A Figura 20 demonstra um exemplo prático desta funcionalidade, onde o usuário solicitou que as questões fossem geradas em espanhol.

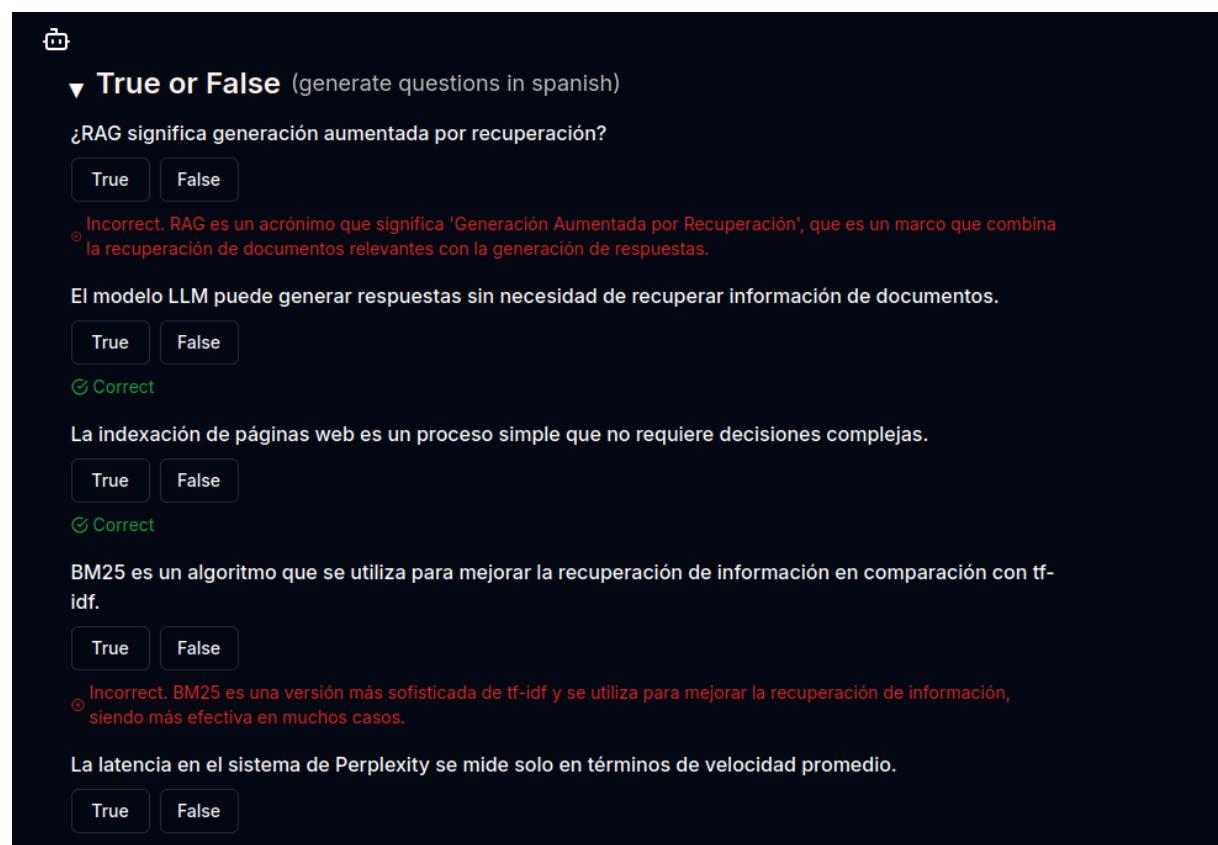


Figura 20 – Questões de Verdadeiro ou Falso geradas em espanhol após customização do prompt

A exposição do prompt ao usuário traz benefícios significativos ao processo de

aprendizagem. Esta transparência permite adaptações para diferentes estilos de aprendizagem e níveis de conhecimento. Os usuários podem experimentar diferentes abordagens e refinar suas instruções para obter resultados mais alinhados com suas necessidades específicas.

5.6 Bate-Papo com Vídeo

O componente de bate-papo com vídeo permite aos usuários fazer perguntas e obter respostas contextualizadas sobre o conteúdo dos vídeos. Esta funcionalidade transforma a experiência de assistir vídeos ao possibilitar interações diretas com o material apresentado.

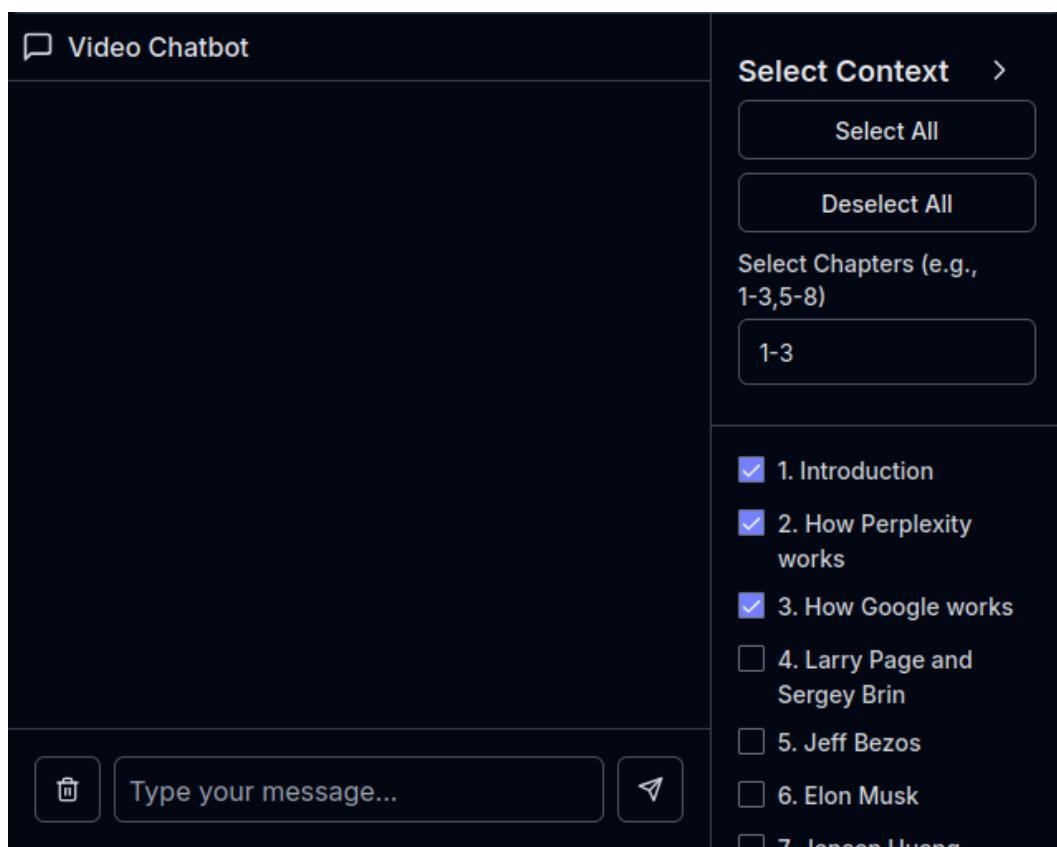


Figura 21 – Interface do componente de bate-papo com vídeo, mostrando painel de seleção de contexto e área de conversação

5.6.1 Interface do Usuário

A interface do bate-papo é composta por dois elementos principais:

- **Painel de Seleção de Contexto:** Um painel expansível à direita permite aos usuários escolher quais partes do vídeo são relevantes para suas consultas. Este componente oferece duas modalidades de seleção: checkboxes individuais para cada capítulo e entrada textual com sintaxe simplificada (por exemplo, "1-3,5" para selecionar os capítulos 1, 2, 3 e 5).

- **Área de Conversação:** Abaixo do player, a área de chat apresenta as mensagens trocadas entre o usuário e o assistente, com suporte a formatação e indicadores de processamento.

Um elemento importante da interface é o botão de exclusão de mensagens, que permite ao usuário limpar todo o histórico de conversa e o contexto selecionado, possibilitando um novo início de interação.

5.6.2 Gerenciamento de Contexto

O sistema implementa duas estratégias principais para seleção de contexto:

- **Contexto com Transcrição na Íntegra:** Quando o vídeo não possui capítulos definidos, o sistema utiliza a transcrição completa como contexto, aplicando truncamento quando necessário.
- **Contexto com Transcrição Dividida em Capítulos:** Quando disponíveis, os capítulos permitem uma seleção mais granular, possibilitando ao usuário focar suas consultas em seções específicas do vídeo.

Para vídeos extensos sem capítulos, o sistema notifica o usuário sobre a necessidade de truncar a transcrição e sugere a geração de capítulos como alternativa. Quando o contexto selecionado é muito grande, o sistema faz ajustes e informa o usuário através de notificações.

5.6.3 Benefícios e Características

O sistema de bate-papo oferece respostas contextualizadas sobre o conteúdo do vídeo e mantém a coerência conversacional durante a interação. A interface responsiva e o gerenciamento de contexto proporcionam uma experiência prática para análise e discussão do conteúdo em vídeo.

Entre os principais benefícios para o usuário estão:

- **Acesso não-linear ao conteúdo:** Possibilidade de explorar partes específicas do vídeo sem assistir à sequência completa
- **Extração de informações:** Obtenção de respostas específicas sem necessidade de assistir ou rever o vídeo inteiro
- **Personalização da experiência:** Foco em áreas de interesse particular
- **Aprofundamento do entendimento:** Capacidade de fazer perguntas para esclarecer conceitos

O sistema transforma a experiência de assistir vídeos ao permitir que os usuários interajam diretamente com o conteúdo através de perguntas e respostas contextualizadas.

5.7 Discussão

Este capítulo apresentou a implementação do VideoLearnAI e as decisões técnicas que moldaram seu desenvolvimento. O projeto enfrentou diversos desafios que exigiram adaptações e mudanças de abordagem ao longo do processo.

Na melhoria de legibilidade das legendas, a tentativa inicial de utilizar LLMs apresentou inconsistências nos resultados, alto custo computacional e tempos de resposta inaceitáveis. A migração para o modelo especializado SaT, executado em infraestrutura serverless GPU, tornou o processo muito mais rápido e confiável, com a vantagem adicional de não alterar o conteúdo original do texto, apenas melhorando sua estruturação.

Para a geração de capítulos, a abordagem inicial de solicitar ao LLM a identificação direta de timestamps mostrou-se imprecisa, pois os modelos de linguagem têm dificuldade em lidar com informações temporais específicas. A solução adotada dividiu o problema em duas etapas: segmentação prévia da transcrição em parágrafos numerados com timestamps associados, seguida pela identificação semântica das transições temáticas pelo LLM, que apenas indicava os números dos parágrafos onde ocorriam mudanças de tópico. Uma lógica posterior recolocava os timestamps corretos com base nos números dos parágrafos identificados. Esta decomposição do problema resultou em capítulos mais coerentes e precisos temporalmente.

Na transcrição de áudio, a migração do Whisper para o Deepgram eliminou a necessidade de segmentação de arquivos longos e reduziu drasticamente o tempo de processamento, melhorando significativamente a experiência do usuário. O Deepgram oferece vantagens adicionais como timestamps no nível da palavra, capacidade de diarização (identificação de diferentes falantes) e um modelo de negócios que disponibiliza créditos gratuitos para desenvolvimento, tornando-o uma escolha mais viável para a plataforma.

Para os quizzes interativos, a decisão de expor os prompts ao usuário final, permitindo personalização, equilibrou simplicidade e flexibilidade, beneficiando tanto usuários iniciantes quanto avançados. O uso de function calling garantiu saídas estruturadas e compatíveis com as interfaces gráficas.

No sistema de bate-papo, a seleção manual de contexto baseada em capítulos foi inicialmente preferida a abordagens mais complexas como RAG (Retrieval-Augmented Generation), priorizando controle do usuário e simplicidade de implementação na fase inicial do projeto.

6 CONCLUSÃO

O desenvolvimento do VideoLearnAI demonstrou o potencial significativo da integração entre modelos de linguagem natural e tecnologias web modernas para criar experiências educacionais mais efetivas. A plataforma conseguiu transformar o consumo passivo de vídeos em um processo de aprendizagem ativa, oferecendo ferramentas que promovem maior engajamento e compreensão do conteúdo.

6.1 Objetivos Alcançados

Os objetivos inicialmente propostos foram alcançados através da implementação bem-sucedida das cinco funcionalidades principais:

1. **Melhoria da Legibilidade das Legendas:** A implementação do modelo SAT (Segment Any Text) proporcionou uma solução eficiente e precisa para a segmentação de texto, superando as limitações encontradas na abordagem inicial com LLMs. Esta funcionalidade demonstrou ser fundamental para melhorar a experiência de leitura e compreensão do conteúdo.
2. **Geração de Capítulos:** O sistema de geração automática de capítulos, combinando segmentação de texto com análise por LLMs, mostrou-se eficaz na organização estruturada do conteúdo, facilitando a navegação e o acesso a informações específicas.
3. **Legendas de maior qualidade:** A geração de legendas com o serviço Deepgram ao invés das legendas geradas automaticamente pelo youtube, resultou em um processo mais eficiente e preciso de transcrição, com benefícios adicionais como marcação temporal no nível da palavra e capacidade de diarização dos locutores.
4. **Quizzes Interativos:** Superando as limitações do consumo passivo de vídeo, os questionários interativos com feedback em tempo real promovem engajamento ativo através de exercícios práticos e reflexivos, maximizando a absorção do conteúdo.

5. **Bate-Papo Contextual:** O sistema de chat contextualizado proporcionou uma forma natural e eficiente de interação com o conteúdo do vídeo, com gerenciamento adequado de contexto, transparente para o usuário.
6. **Sistema de Salvamento de Progresso:** Foi implementado um mecanismo que permite aos usuários salvar seu progresso de estudo. Esta funcionalidade possibilita que os usuários retomem seus estudos exatamente de onde pararam, promovendo continuidade no processo de aprendizagem.

6.2 Trabalhos Futuros

Após a implementação inicial do VideoLearnAI, ficou evidente que além das funcionalidades básicas já desenvolvidas, existem melhorias possíveis que podem enriquecer gradualmente a experiência dos usuários. A seguir, apresento algumas propostas viáveis para o aprimoramento contínuo da plataforma.

Uma evolução natural seria aproveitar plenamente os recursos já disponíveis no serviço Deepgram, especificamente a funcionalidade de diarização que já está incluída na API utilizada. A integração desta capacidade permitiria identificar e diferenciar múltiplos falantes nas transcrições, enriquecendo substancialmente a experiência do usuário através de elementos visuais distintivos como legendas coloridas ou avatares para cada interlocutor. Esta implementação não exigiria mudanças significativas na infraestrutura existente, apenas o processamento adicional dos metadados de diarização já fornecidos pelo Deepgram. Além de melhorar a legibilidade, esta funcionalidade forneceria contexto adicional valioso para o sistema de chat e geração de quizzes, permitindo análises mais granulares do conteúdo e facilitando a compreensão de debates e discussões em grupo.

Um aprimoramento significativo para a legibilidade das legendas seria a implementação de modelos especializados em pontuação automática, complementando o atual sistema de segmentação de parágrafos. A integração destes modelos permitiria a adição precisa de sinais de pontuação, aproximando a transcrição de um texto natural e fluido.

A evolução do sistema de quizzes para incorporar princípios de repetição espaçada representa outra frente promissora. Esta abordagem, fundamentada em pesquisas sobre memória e retenção de conhecimento, programaria revisões estratégicas do conteúdo em intervalos crescentes, maximizando a retenção de longo prazo. A implementação poderia incluir notificações personalizadas para revisão de conceitos específicos e adaptação dinâmica da dificuldade das questões com base no histórico de desempenho do usuário, criando um ciclo de aprendizado verdadeiramente adaptativo.

Para o componente de bate-papo, a integração de um sistema RAG (Retrieval-Augmented Generation) agêntico representaria um salto qualitativo significativo. Esta

evolução permitiria que o sistema não apenas respondesse a perguntas com base no contexto selecionado, mas também navegasse inteligentemente por todo o conteúdo do vídeo, recuperando informações relevantes de forma autônoma. Um sistema RAG agêntico poderia realizar análises comparativas entre diferentes seções do vídeo, identificar contradições ou complementaridades no discurso e até mesmo buscar informações em fontes externas quando apropriado, tudo mantendo uma interação natural e contextualizada com o usuário.

A dimensão social da aprendizagem poderia ser explorada através da implementação de elementos de gamificação centrados na competição saudável. Um sistema de ranking por vídeo permitiria aos usuários comparar seu desempenho nos quizzes com outros estudantes, estimulando maior engajamento com o conteúdo. Esta funcionalidade poderia ser expandida para incluir badges por conquistas específicas, como completar todos os quizzes de um vídeo ou manter uma sequência de respostas corretas. Tais elementos de gamificação não apenas tornariam a experiência mais envolvente, mas também forneceriam motivação adicional para os usuários revisitarem conteúdos e aprimorarem seu entendimento, transformando o processo de aprendizagem em uma atividade mais dinâmica e recompensadora.

Por fim, a internacionalização da plataforma representa uma oportunidade significativa de expansão. A implementação incluiria a tradução da interface do usuário para múltiplos idiomas e a integração com serviços de tradução como o *DeepL*¹ para converter automaticamente as legendas dos vídeos. Esta abordagem simplificada removeria barreiras linguísticas ao conhecimento, permitindo que usuários accessem conteúdo educacional independentemente do idioma original, sem necessidade de implementações complexas adicionais.

Estas direções de desenvolvimento, embora ambiciosas, são tecnicamente viáveis e alinhadas com a visão original do VideoLearnAI: transformar o consumo passivo de vídeos em experiências de aprendizagem ativas, personalizadas e eficazes.

¹<https://www.deepl.com/pt-BR/translator>

REFERÊNCIAS

- BARBERO, F.; BANINO, A.; KAPTUROWSKI, S.; KUMARAN, D.; ARAÚJO, J. G.; VITVITSKYI, A.; PASCANU, R.; VELIČKOVIĆ, P. Transformers need glasses! Information over-squashing in language tasks. **arXiv preprint arXiv:2406.04267**, [S.I.], 2024.
- BELTAGY, I.; PETERS, M. E.; COHAN, A. Longformer: The long-document transformer. In: XIV PREPRINT ARXIV:2004.05150, 2020. **Anais...** [S.I.: s.n.], 2020.
- BOMMASANI, R.; HUDSON, D. A.; ADELI, E.; ALTMAN, R.; ARORA, S.; ARX, S. von; BERNSTEIN, M. S.; BOHG, J.; BOSSELUT, A.; BRUNSKILL, E. et al. On the opportunities and risks of foundation models. **arXiv preprint arXiv:2108.07258**, [S.I.], 2021.
- BONWELL, C. C.; EISON, J. A. **Active Learning**: Creating Excitement in the Classroom. [S.I.]: ASHE-ERIC Higher Education Report, 1991.
- BROWN, T.; MANN, B.; RYDER, N.; SUBBIAH, M.; KAPLAN, J. D.; DHARIWAL, P.; NEELAKANTAN, A.; SHYAM, P.; SAstry, G.; ASKELL, A. et al. Language models are few-shot learners. **Advances in neural information processing systems**, [S.I.], v.33, p.1877–1901, 2020.
- DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: CONFERENCE OF THE NORTH AMERICAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS: HUMAN LANGUAGE TECHNOLOGIES, 2019., 2019. **Proceedings...** [S.I.: s.n.], 2019. p.4171–4186.
- DUNLOSKY, J.; RAWSON, K. A.; MARSH, E. J.; NATHAN, M. J.; WILLINGHAM, D. T. Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. **Psychological Science in the Public Interest**, [S.I.], v.14, n.1, p.4–58, 2013.

FREEMAN, S.; EDDY, S. L.; McDONOUGH, M.; SMITH, M. K.; OKOROAFOR, N.; JORDT, H.; WENDEROTH, M. P. Active learning increases student performance in science, engineering, and mathematics. **Proceedings of the National Academy of Sciences**, [S.I.], v.111, n.23, p.8410–8415, 2014.

FROHMANN, M.; STERNER, I.; VULIC, I.; MINIXHOFER, B.; SCHEDL, M. Segment Any Text: A Universal Approach for Robust, Efficient and Adaptable Sentence Segmentation. **arXiv preprint arXiv:2406.16678**, [S.I.], 2024.

GAO, Y.; XIONG, Y.; GAO, X.; JIANG, K.; SHEN, J.; REN, X.; HAN, J. Retrieval-augmented generation for large language models: A survey. **arXiv preprint arXiv:2312.10997**, [S.I.], 2023.

GUHR, O.; SCHUMANN, A.-K.; BAHRMANN, F.; BÖHME, H. J. FullStop: Multilingual Deep Models for Punctuation Prediction. In: SWISS TEXT ANALYTICS CONFERENCE 2021, 2021, Winterthur, Switzerland. **Proceedings... CEUR Workshop Proceedings**, 2021.

GUO, P. J.; KIM, J.; RUBIN, R. How video production affects student engagement: An empirical study of MOOC videos. **Proceedings of the first ACM conference on Learning@ scale conference**, [S.I.], p.41–50, 2014.

KAPLAN, J.; MCCANDLISH, S.; HENIGHAN, T.; BROWN, T. B.; CHESS, B.; CHILD, R.; GRAY, S.; RADFORD, A.; WU, J.; AMODEI, D. Scaling laws for neural language models. **arXiv preprint arXiv:2001.08361**, [S.I.], 2020.

KOJIMA, T.; GU, S. S.; REID, M.; MATSUO, Y.; IWASAWA, Y. Large language models are zero-shot reasoners. **Advances in Neural Information Processing Systems**, [S.I.], v.35, p.22199–22213, 2022.

LEWIS, P.; PEREZ, E.; PIKTUS, A.; PETRONI, F.; KARPUKHIN, V.; GOYAL, N.; KÜTLER, H.; LEWIS, M.; YIH, W.-t.; ROCKTÄSCHEL, T. et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. **Advances in Neural Information Processing Systems**, [S.I.], v.33, p.9459–9474, 2020.

LIU, N. F.; BOSSELUT, A.; SRINIVASAN, D.; CHOI, Y.; HAJISHIRZI, H.; KHASHABI, D. Lost in the middle: How language models use long contexts. **arXiv preprint arXiv:2307.03172**, [S.I.], 2023.

LIU, P.; YUAN, W.; FU, J.; JIANG, Z.; HAYASHI, H.; NEUBIG, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. **ACM Computing Surveys**, [S.I.], v.55, n.9, p.1–35, 2023.

- LIU, Y.; SUN, Y.; ZHONG, Y.; HAN, C.; SUN, R.; ZHANG, X.; PENG, Y.; TANG, Z.; HUANG, J.; LAI, Z. et al. Evaluating large language models: A comprehensive survey. **arXiv preprint arXiv:2310.19736**, [S.I.], 2023.
- OpenAI. GPT-4 Technical Report. **arXiv preprint arXiv:2303.08774**, [S.I.], 2023.
- OpenAI. **Pricing**. Accessed: 2023-12-01, <https://openai.com/pricing>.
- PRINCE, M. Does active learning work? A review of the research. **Journal of Engineering Education**, [S.I.], v.93, n.3, p.223–231, 2004.
- RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. Language models are unsupervised multitask learners. **OpenAI blog**, [S.I.], v.1, n.8, p.9, 2019.
- RAFFEL, C.; SHAZER, N.; ROBERTS, A.; LEE, K.; NARANG, S.; MATENA, M.; ZHOU, Y.; LI, W.; LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. **Journal of Machine Learning Research**, [S.I.], v.21, p.1–67, 2020.
- SCHWAN, S.; RIEMPP, R. Learning by viewing versus learning by doing: Evidence-based guidelines for principled learning environments. **Learning and Instruction**, [S.I.], v.14, n.6, p.587–596, 2004.
- SINGH, A.; EHTESHAM, A.; KUMAR, S.; KHOEI, T. T. Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG. **arXiv preprint arXiv:2501.09136**, [S.I.], 2025.
- VASWANI, A.; SHAZER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: **ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS**, 2017. **Anais...** [S.I.: s.n.], 2017. p.5998–6008.
- VEMPRALA, S.; BONATTI, R.; BUCKER, A.; KAPOOR, A. ChatGPT for robotics: Design principles and model abilities. **arXiv preprint arXiv:2306.17582**, [S.I.], 2023.
- WEI, J.; WANG, X.; SCHUURMANS, D.; BOSMA, M.; ICHTER, B.; XIA, F.; CHI, E.; LE, Q.; ZHOU, D. Chain of thought prompting elicits reasoning in large language models. **Advances in Neural Information Processing Systems**, [S.I.], v.35, p.24824–24837, 2022.
- WENG, L. LLM powered autonomous agents. **lilianweng.github.io**, [S.I.], 2023.
- WHITE, J.; FU, Q.; HAYS, S.; SANDBORN, M.; OLEA, C.; GILBERT, H.; ELNASHAR, A.; SPENCER-SMITH, J.; SCHMIDT, D. C. Prompt engineering for large language models: A survey. **arXiv preprint arXiv:2307.10169**, [S.I.], 2023.

XU, Y.; SARTHI, S.; AGARWAL, A.; GUPTA, A.; SAXENA, A.; ARALIKATTE, R.; BATRA, D.; PARIKH, D.; MISRA, I.; AWADALLAH, A. Retrieval-augmented generation for knowledge-intensive nlp tasks. **arXiv preprint arXiv:2305.14002**, [S.I.], 2023.

YAO, S.; ZHAO, J.; YU, D.; DU, N.; SHAFRAN, I.; NARASIMHAN, K.; CAO, Y. ReAct: Synergizing reasoning and acting in language models. **arXiv preprint arXiv:2210.03629**, [S.I.], 2023.