Universidade Federal de Ouro Preto
Escola de Minas
Engenharia de Controle e Automação

Ângelo César Bosada Júnior

Modelos de Linguagem de Grande Escala: Estudo de Caso e Perspectivas

Ouro Preto, 2026

Ângelo César Bosada Júnior

Modelos de Linguagem de Grande Escala: Estudo de Caso e Perspectivas

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheiro de Controle e Automação

Orientador: Prof. Agnaldo José da Rocha Reis

Ouro Preto
2026

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CONTROLE E AUTOMACAO

FOLHA DE APROVAÇÃO

Ângelo César Bosada Júnior

Modelos de Linguagem de Grande Escala: Estudo de Caso e Perspectivas

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheiro de Controle e Automação

Aprovada em 03 de março de 2026

Membros da banca

Doutor - Agnaldo José da Rocha Reis - Orientador (Universidade Federal de Ouro Preto)
Doutor - Eduardo José da Silva Luz - (Universidade Federal de Ouro Preto)
Mestre - Felipe Augusto Tavares - (Instituto Tecnológico Vale)

Agnaldo José da Rocha Reis, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 07/03/2026

Documento assinado eletronicamente por **Agnaldo Jose da Rocha Reis**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 07/03/2026, às 13:02, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **1070939** e o código CRC **C4F082AE**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591533 - www.ufop.br

Resumo

Este trabalho investiga os modelos de linguagem de larga escala (LLMs), explorando seus fundamentos, evolução, aplicações e limitações. Partindo de uma revisão histórica da inteligência artificial e do processamento de linguagem natural, o estudo destaca o papel central da arquitetura Transformer (VASWANI *et al.*, 2017), que, com seus mecanismos de autoatenção, superou as limitações de modelos anteriores baseados em RNNs e CNNs. Examina-se ainda a contribuição seminal de modelos pré-treinados como GPT-1 (RADFORD *et al.*, 2018), BERT (DEVLIN *et al.*, 2019) e GPT-3 (BROWN *et al.*, 2020), que estabeleceram o paradigma do pré-treinamento seguido de ajuste fino, potencializando a generalização. A discussão também abrange o surgimento dos modelos de fundação, notáveis por sua escalabilidade e adaptabilidade, e os fenômenos de aprendizado em contexto e de capacidades emergentes (BOMMASANI *et al.*, 2021). Metodologicamente, o estudo instanciou o modelo Mistral-7B-v0.3, treinando-o com a técnica de otimização QLoRA. A base de dados, composta por 26 artigos científicos, foi convertida em 873 pares de instrução–resposta. O treinamento, realizado em uma GPU NVIDIA RTX A4000 (16 GB de VRAM), ajustou aproximadamente 0,09% dos parâmetros do modelo, culminando em uma perda final média de 1,72 após 7 horas. A avaliação combinou análise quantitativa (função de perda) e qualitativa (comparação de respostas a seis questões de referência). Os resultados indicaram uma certa melhoria em cinco questões, com ganhos em precisão, coerência e citação de referências acadêmicas, enquanto uma questão apresentou piora, evidenciando a dependência crítica da qualidade dos dados de treinamento. Conclui-se que os LLMs constituem uma infraestrutura tecnológica central com potencial para transformar setores como saúde, finanças, direito e educação, democratizando o acesso a ferramentas avançadas. No entanto, persistem desafios urgentes relacionados a vieses, custo computacional, governança e alinhamento ético (OUYANG *et al.*, 2022). Este trabalho demonstra a viabilidade de pesquisas experimentais com recursos limitados, evidenciando que é possível realizar treinamento especializado de LLMs em ambientes acadêmicos e obter avanços qualitativos relevantes.

Palavras-chave: Inteligência Artificial Generativa; Modelos de Linguagem de Larga Escala; Transformers; Modelos de Fundação; Aprendizado em Contexto.

Abstract

This work investigates Large Language Models (LLMs), exploring their foundations, evolution, applications, and limitations. Starting from a historical review of artificial intelligence and natural language processing, the study highlights the central role of the Transformer architecture (VASWANI *et al.*, 2017), which, through its self-attention mechanisms, overcame the limitations of previous models based on RNNs and CNNs. The study also examines the seminal contributions of pretrained models such as GPT-1 (RADFORD *et al.*, 2018), BERT (DEVLIN *et al.*, 2019), and GPT-3 (BROWN *et al.*, 2020), which established the paradigm of pre-training followed by fine-tuning, enhancing generalization capabilities. The discussion also addresses the emergence of foundation models, notable for their scalability and adaptability, as well as the phenomena of in-context learning and emergent capabilities (BOMMASANI *et al.*, 2021). Methodologically, the study instantiated the Mistral-7B-v0.3 model, training it using the QLoRA optimization technique. The dataset, composed of 26 scientific articles, was converted into 873 instruction–response pairs. The training, performed on an NVIDIA RTX A4000 GPU (16 GB of VRAM), adjusted approximately 0.09% of the model's parameters, resulting in a final average loss of 1.72 after 7 hours. The evaluation combined quantitative analysis (loss function) and qualitative analysis (comparison of responses to six reference questions). The results indicated a certain improvement in five questions, with gains in accuracy, coherence, and citation of academic references, while one question showed a decline, highlighting the critical dependence on the quality of the training data. It is concluded that LLMs constitute a central technological infrastructure with the potential to transform sectors such as healthcare, finance, law, and education, democratizing access to advanced tools. However, urgent challenges remain related to bias, computational cost, governance, and ethical alignment (OUYANG *et al.*, 2022). This work demonstrates the feasibility of experimental research with limited resources, showing that it is possible to perform specialized LLM training in academic environments and obtain relevant qualitative advances.

Keywords: Generative Artificial Intelligence; Large Language Models; Transformers; Foundation Models; In-Context Learning.

Lista de figuras

Sumário

1 Introdução

Nos últimos anos, os modelos de linguagem de larga escala (Large Language Models – LLMs) ganharam destaque devido à sua capacidade de gerar texto semelhante ao produzido por seres humanos e de executar uma ampla gama de tarefas de processamento de linguagem natural (PLN). Esses sistemas, conhecidos como *inteligências artificiais generativas*, são treinados em extensos conjuntos de dados textuais e podem ser ajustados para diferentes aplicações por meio de técnicas de *fine-tuning*. O avanço desses modelos foi possibilitado, em grande medida, pela introdução da arquitetura Transformer Vaswani *et al.* (2017), que substituiu abordagens recorrentes por mecanismos de autoatenção.

A evolução desse campo pode ser observada em modelos como o GPT Radford *et al.* (2018), o BERT Devlin *et al.* (2019) e o GPT-3 Brown *et al.* (2020), os quais ampliaram significativamente a capacidade de aprendizado e generalização. Paralelamente, surgiram modelos de fundação, como o LLaMA Touvron *et al.* (2023), que demonstram ser possível alcançar desempenho competitivo utilizando dados públicos e acessíveis à comunidade científica.

1.1 Justificativas e Relevância

O crescimento recente das IAs generativas gerou um verdadeiro "boom" no debate público e científico. Muitas vezes, essas tecnologias são vistas como soluções ilimitadas, o que gera expectativas irreais. Contudo, apesar das limitações e riscos apontados pela literatura Ouyang *et al.* (2022) e Christiano *et al.* (2017), os modelos de linguagem constituem ferramentas de grande relevância quando aplicados de forma adequada.

Dessa forma, este trabalho se justifica pela necessidade de compreender, de forma técnica, o funcionamento dos LLMs e suas reais possibilidades de aplicação em diferentes setores. Sua relevância está em mostrar que, embora não sejam soluções mágicas, os LLMs representam instrumentos eficazes e adaptáveis para tarefas que demandam automação, análise de grandes volumes de dados e suporte à tomada de decisão.

1.2 Objetivos

1.2.1 Objetivo Geral

Compreender e avaliar o funcionamento dos modelos de linguagem de larga escala (*Large Language Models* – LLMs), por meio de uma análise teórica e da instanciação prática

de um modelo especializado, evidenciando suas potencialidades, limitações e aplicações em diferentes contextos.

1.2.2 Objetivos Específicos

Para alcançar o objetivo geral, este trabalho tem como objetivos específicos:

1. Realizar uma revisão sistemática da literatura sobre inteligências artificiais generativas e arquiteturas baseadas em *Transformers*, destacando sua evolução histórica e fundamentos técnicos;
2. Implementar e treinar o modelo **Mistral-7B** utilizando uma base composta por artigos científicos selecionados neste TCC, com o propósito de construir uma versão especializada no tema;
3. Avaliar o desempenho do modelo instanciado em tarefas de geração e compreensão de texto comparando uma versão pré-treinada antes e após um fine-tuning.

1.3 Organização e Estrutura

Este trabalho está organizado da seguinte forma: na **Capítulo 1**, apresenta-se a introdução do tema, sua justificativa, objetivos e organização. O **Capítulo 2** traz a fundamentação teórica, discutindo os conceitos centrais das IAs generativas, suas arquiteturas e evolução histórica. O **Capítulo 3** descreve a metodologia utilizada, com destaque para os procedimentos de treinamento experimental. O **Capítulo 4** apresenta os resultados obtidos e a análise das aplicações estudadas. Por fim, o **Capítulo 5** expõe as conclusões, ressaltando as contribuições, limitações e sugestões para trabalhos futuros.

2 Revisão de literatura

O Processamento de Linguagem Natural (PLN) evoluiu ao longo de diferentes paradigmas dentro da inteligência artificial. Em sua fase inicial, predominavam abordagens baseadas em regras linguísticas, nas quais especialistas codificavam manualmente estruturas sintáticas e semânticas para permitir a interpretação automática da linguagem (SPARCK JONES, 1994). Embora eficazes em domínios restritos, esses sistemas apresentavam limitações de escalabilidade e dificuldades em lidar com a ambiguidade inerente à linguagem humana.

A partir da década de 1990, o campo passou a adotar abordagens estatísticas baseadas em grandes corpora textuais. Modelos probabilísticos, como n-grams e Hidden Markov Models (HMMs), tornaram-se amplamente utilizados em tarefas como reconhecimento de fala e modelagem de linguagem (JELINEK, 1997; RABINER, 1989). Nesse período também surgiram os primeiros modelos neurais de linguagem, capazes de aprender representações distribuídas de palavras a partir de grandes volumes de dados textuais (**bengio2003**).

Com o avanço do aprendizado profundo na década de 2010, arquiteturas como as Redes Neurais Recorrentes (RNNs) e suas variantes com memória de longo prazo, como as LSTMs, passaram a dominar o estado da arte em tarefas sequenciais de linguagem (HOCHREITER; SCHMIDHUBER, 1997). Esses modelos permitiram avanços importantes em aplicações como tradução automática, classificação de textos e sistemas de resposta a perguntas.

Nesse contexto consolidou-se também o paradigma dos modelos pré-treinados, nos quais um modelo é inicialmente treinado em grandes volumes de texto e posteriormente adaptado para tarefas específicas. Exemplos marcantes incluem o GPT, baseado em modelagem autoregressiva de linguagem (RADFORD *et al.*, 2018), e o BERT, que introduziu representações bidirecionais capazes de capturar o contexto completo de uma sequência textual (DEVLIN *et al.*, 2019). Esses avanços estabeleceram as bases para o desenvolvimento dos modelos baseados em arquitetura Transformer (VASWANI *et al.*, 2017), que posteriormente redefiniriam o estado da arte no processamento de linguagem natural.

2.1 Arquitetura Transformer

A arquitetura *Transformer*, apresentada no artigo *Attention is All You Need* Vaswani *et al.* (2017), representou uma mudança no campo do Processamento de Linguagem Natural (PLN). Para compreender sua importância, é necessário lembrar como eram os modelos dominantes antes de sua introdução.

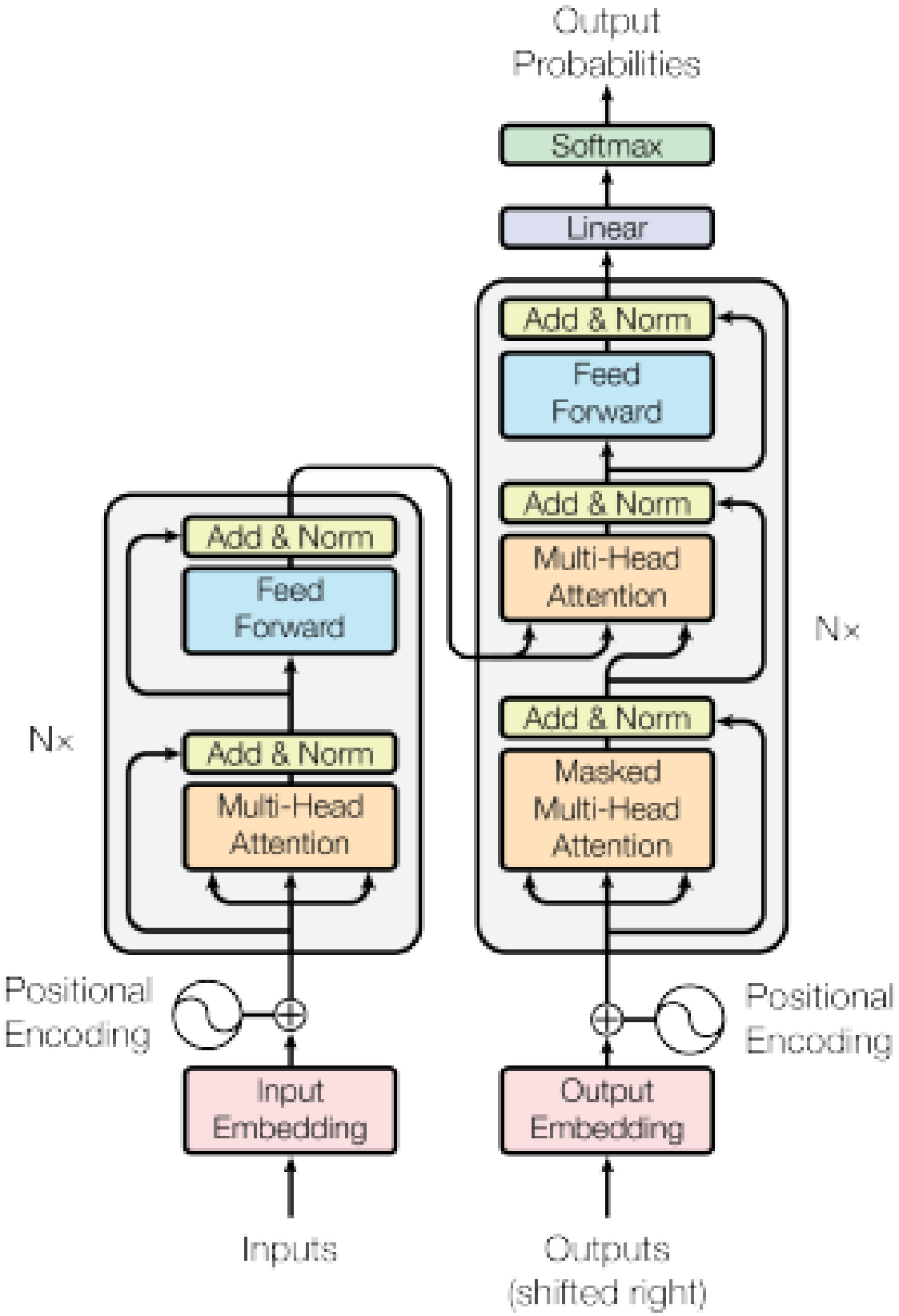

Figura 1 – Arquitetura Transformer apresentada em *Attention is All You Need*.

Adaptado de Vaswani *et al.* (2017).

Até então, prevaleciam as Redes Neurais Recorrentes (RNNs) e, em especial, suas versões mais avançadas como as LSTMs (*Long Short-Term Memory*), introduzidas por Hochreiter e Schmidhuber (1997). Essas redes processavam sequências palavra a palavra, mantendo um estado interno que funcionava como "memória" do contexto anterior. Embora eficazes em dependências locais, apresentavam duas limitações cruciais: (i) eram lentas, pois processavam tokens de forma sequencial, dificultando a paralelização em GPUs modernas, e (ii) falhavam na captura de dependências de longo alcance, pois a informação se degradava à medida que a sequência crescia (HOCHREITER; SCHMIDHUBER, 1997; VASWANI *et al.*, 2017).

O Transformer solucionou essas restrições ao abandonar a recorrência e propor o mecanismo de **autoatenção (self-attention)**. Essa inovação permite que cada palavra interaja diretamente com todas as outras em uma sentença, independentemente da distância entre elas. Por exemplo, ao interpretar o pronome "ele", o modelo pode imediatamente associá-lo ao substantivo "João", localizado no início do texto. Além de mais preciso, esse processo é altamente paralelizável, acelerando consideravelmente o treinamento em larga escala (VASWANI *et al.*, 2017).

Autoatenção

O mecanismo de autoatenção pode ser visualizado como uma rede de conexões entre todas as palavras de uma frase, em que a intensidade do peso indica a força da relação contextual. Essa capacidade de modelar relações globais tornou-se um diferencial essencial para superar os gargalos das RNNs.

Peças-chave do Transformer

Para funcionar de forma eficaz, a arquitetura incorpora alguns elementos adicionais:

- Positional Encoding (Codificação Posicional): como o Transformer processa todas as palavras simultaneamente, perde a noção natural de ordem. Para contornar isso, é adicionada uma codificação que informa a posição relativa de cada palavra, preservando a estrutura sequencial do texto (VASWANI *et al.*, 2017).
- Multi-Head Attention (Atenção Multi-Cabeça): em vez de realizar uma única análise de atenção, o modelo utiliza múltiplas "cabeças", cada uma especializada em capturar diferentes tipos de relações. Os resultados dessas cabeças são combinados, produzindo uma representação mais rica (VASWANI *et al.*, 2017).
- Camadas Feed-Forward: após a etapa de atenção, cada posição passa por uma rede neural independente, refinando a informação obtida.

O impacto do Transformer não se limitou a avanços incrementais; ele se tornou o alicerce tecnológico dos Modelos de Linguagem de Larga Escala (LLMs). O GPT (*Generative Pre-trained Transformer*) utilizou a versão autoregressiva do Transformer, treinando para prever a próxima palavra em sequências textuais (RADFORD *et al.*, 2018). Já o BERT (*Bidirectional Encoder Representations from Transformers*) explorou a bidirecionalidade, analisando o contexto à esquerda e à direita de cada palavra, obtendo avanços notáveis em tarefas como pergunta e resposta e análise de sentimentos (DEVLIN *et al.*, 2019).

A escalabilidade dessa arquitetura ficou evidente em modelos como o GPT-3, que, com 175 bilhões de parâmetros, não apenas superou desempenhos anteriores, mas apresentou capacidades emergentes, como a habilidade de realizar tarefas inéditas a partir de poucos exemplos (*few-shot learning*) (BROWN *et al.*, 2020).

Em termos mais amplos, a arquitetura Transformer estabeleceu o paradigma dos modelos de fundação (*foundation models*), que consistem em sistemas pré-treinados em larga escala e posteriormente adaptados para uma variedade de tarefas e domínios (BOMMASANI *et al.*, 2021). Esses modelos constituem a base da revolução atual em IA generativa, ao unirem versatilidade, escalabilidade e desempenho superior em diferentes contextos.

2.2 Modelos Pré-Treinados

O conceito de pré-treinamento em Processamento de Linguagem Natural (PLN) baseia-se na ideia de treinar modelos em grandes volumes de dados textuais para aprender representações gerais da linguagem, que posteriormente podem ser reutilizadas ou ajustadas para tarefas específicas. Uma das primeiras abordagens amplamente difundidas foi o uso de representações distribuídas de palavras, como o Word2Vec, que demonstrou que redes neurais podem aprender representações vetoriais semânticas capazes de capturar relações linguísticas a partir de grandes corpora textuais (MIKOLOV *et al.*, 2013). Esse avanço permitiu que modelos reutilizassem representações previamente aprendidas, reduzindo a necessidade de treinamento completo para cada nova tarefa.

Posteriormente, o conceito foi ampliado com o desenvolvimento de modelos de linguagem mais complexos baseados em redes neurais profundas. Trabalhos como o GPT (RADFORD *et al.*, 2018) e o BERT (DEVLIN *et al.*, 2019) consolidaram a estratégia de *pre-train and fine-tune*, na qual um modelo é inicialmente treinado em larga escala e posteriormente ajustado para aplicações específicas. Esse paradigma tornou-se um dos pilares do desenvolvimento moderno em PLN e abriu caminho para os modelos de linguagem de larga escala utilizados atualmente.

2.3 Fine-Tuning em Modelos de Linguagem

O fine-tuning consiste no processo de adaptação de um modelo previamente treinado em grandes corpora textuais para uma tarefa ou domínio específico. Nesse paradigma, o modelo é inicialmente submetido a um pré-treinamento em larga escala, no qual aprende padrões gerais da linguagem, e posteriormente ajustado com um conjunto de dados menor e mais especializado. Esse procedimento tornou-se uma prática central no desenvolvimento de modelos de linguagem modernos, permitindo reutilizar conhecimento previamente adquirido e reduzir o custo computacional necessário para aplicações específicas (RADFORD *et al.*, 2018; DEVLIN *et al.*, 2019).

A estratégia de pré-treinamento seguida de ajuste fino mostrou-se extremamente eficaz em diversas tarefas de processamento de linguagem natural, como classificação de textos, resposta a perguntas e tradução automática. Modelos baseados na arquitetura Transformer, introduzida por Vaswani et al. (VASWANI *et al.*, 2017), demonstraram que representações aprendidas durante o pré-treinamento podem ser transferidas com grande eficiência para novas tarefas. Essa abordagem tornou-se um dos pilares do desenvolvimento dos modelos de linguagem de larga escala (LLMs), permitindo que arquiteturas previamente treinadas sejam especializadas para domínios específicos por meio de conjuntos de dados relativamente pequenos (BROWN *et al.*, 2020; BOMMASANI *et al.*, 2021).

2.4 LoRA e QLoRA

Apesar das vantagens do fine-tuning tradicional, o ajuste completo de todos os parâmetros de um modelo de linguagem de grande escala pode demandar recursos computacionais elevados. Para mitigar esse problema, foram desenvolvidas técnicas de *parameter-efficient fine-tuning* (PEFT), que permitem adaptar modelos pré-treinados modificando apenas uma pequena fração de seus parâmetros.

Uma das abordagens mais influentes nesse contexto é o LoRA (Low-Rank Adaptation), proposto por Hu et al. (HU *et al.*, 2021). Nessa técnica, em vez de atualizar diretamente os pesos originais do modelo, são introduzidas matrizes adicionais de baixa dimensão que representam atualizações de baixa ordem nos parâmetros das camadas do Transformer. Esse mecanismo permite que o modelo seja adaptado para novas tarefas mantendo a maior parte dos parâmetros congelados, reduzindo significativamente o custo computacional e o consumo de memória.

Posteriormente, Dettmers et al. (DETTMERS *et al.*, 2023) propuseram o QLoRA, uma extensão que combina a técnica LoRA com quantização em baixa precisão. Nesse método, os pesos do modelo são quantizados para representações de menor precisão (tipicamente 4 bits), enquanto os parâmetros adicionais responsáveis pela adaptação permanecem em

maior precisão para garantir estabilidade no treinamento. Essa abordagem permite realizar fine-tuning de modelos de bilhões de parâmetros utilizando hardware mais acessível, mantendo desempenho competitivo em comparação com métodos tradicionais.

Essas técnicas tornaram-se particularmente relevantes no contexto de pesquisas acadêmicas e aplicações industriais, pois possibilitam a especialização de modelos de linguagem de grande escala com recursos computacionais limitados, ampliando o acesso a experimentação e desenvolvimento em inteligência artificial generativa.

2.5 Alinhamento e Questões Éticas

Os *Large Language Models* (LLMs) têm demonstrado capacidades notáveis em tarefas de processamento de linguagem natural, mas também levantam preocupações críticas quanto à segurança, à ética e à confiabilidade de suas respostas. Por serem treinados em grandes volumes de dados da internet, esses modelos podem reproduzir vieses sociais, opiniões tendenciosas e até mesmo informações falsas, comprometendo sua utilização em contextos sensíveis como saúde, direito e educação (BOMMASANI *et al.*, 2021).

Nesse cenário, surge o campo do **alinhamento** (*alignment*), cujo objetivo é garantir que os modelos se comportem de forma coerente com valores humanos e normas sociais. Em outras palavras, busca-se que os sistemas não apenas sejam competentes, mas também responsáveis e seguros.

2.5.1 Reinforcement Learning from Human Feedback (RLHF)

A abordagem mais difundida para alinhamento é o RLHF, proposta por (CHRISTIANO *et al.*, 2017). O método consiste em três etapas principais:

1. Humanos avaliam respostas geradas pelo modelo, classificando-as como mais ou menos adequadas.
2. Essas preferências humanas são transformadas em um sinal de recompensa.
3. O modelo é reajustado para maximizar a recompensa, aprendendo a priorizar respostas consideradas úteis, seguras e verídicas (CHRISTIANO *et al.*, 2017).

Essa técnica pode ser entendida como um processo de ensino indireto, em que o modelo internaliza padrões de comportamento humano por meio de reforço.

2.5.2 InstructGPT

A aplicação prática do RLHF foi evidenciada pelo **InstructGPT**, desenvolvido pela OpenAI. Nesse trabalho, um modelo de apenas 1,3 bilhão de parâmetros, ajustado

com feedback humano, foi preferido por avaliadores em relação ao GPT-3 original, que possuía 175 bilhões de parâmetros. Esse resultado demonstrou que o alinhamento pode ser mais relevante para a qualidade do que simplesmente aumentar a escala dos modelos (OUYANG *et al.*, 2022).

2.5.3 Desafios

Apesar dos avanços, o alinhamento ainda enfrenta obstáculos significativos. Entre eles, destacam-se:

- **Dados de Qualidade**: a coleta de feedback humano confiável e consistente é cara e sujeita a introduzir novos vieses.
- **Custo do Alinhamento**: em alguns casos, o processo de alinhamento leva a perda de desempenho em tarefas úteis, fenômeno descrito como *alignment tax* (OUYANG *et al.*, 2022).
- **Avaliação Multidimensional**: medir a eficácia de um modelo não envolve apenas acurácia, mas também atributos como segurança, justiça e honestidade, o que torna a avaliação complexa e multidisciplinar (WANG *et al.*, 2023).

2.5.4 Questões Éticas

As limitações éticas dos LLMs não se restringem a problemas técnicos. Há riscos de mau uso, como a geração de desinformação em larga escala, manipulação política ou criação de códigos maliciosos (BOMMASANI *et al.*, 2021). Soma-se a isso a falta de transparência sobre os dados de treinamento, dificultando a responsabilização em casos de erros ou viés discriminatório (ZHAO *et al.*, 2023).

O alinhamento por meio de técnicas como o RLHF representa um avanço significativo para tornar os LLMs mais seguros e benéficos. Contudo, permanece como um dos maiores desafios da área equilibrar a capacidade técnica desses sistemas com a necessidade de transparência, ética e responsabilidade social. O futuro da IA dependerá não apenas de modelos mais poderosos, mas também de modelos mais alinhados com os valores humanos (OUYANG *et al.*, 2022; WANG *et al.*, 2023).

3 Materiais e Métodos

Este capítulo descreve os recursos computacionais, os procedimentos metodológicos e as ferramentas utilizadas para a instanciação prática deste trabalho. O objetivo foi realizar um *fine-tuning* do modelo Mistral-7B, especializado no tema de Modelos de Linguagem de Larga Escala (LLMs), a partir de um conjunto de artigos científicos previamente selecionados. O modelo selecionado foi escolhido pela grande liberdade nos termos de uso do mesmo, além de ter sido um bom modelo considerando alguns benchmarks analisados no momento da escolha.

3.1 Materiais

3.1.1 Hardware e Ambiente de Execução

O treinamento foi conduzido em ambiente local, utilizando uma GPU NVIDIA RTX A4000, com 16 GB de memória VRAM dedicada. O sistema operacional empregado foi Windows 10, com suporte às bibliotecas CUDA. As principais bibliotecas utilizadas incluíram `PyTorch`, `Transformers`, `PEFT` (Parameter-Efficient Fine-Tuning) e `BitsAndBytes`, todas em versões compatíveis com CUDA 12.8.

3.1.2 Base de Dados

A base de dados foi composta por 26 trabalhos acadêmicos relacionados à inteligência artificial generativa e modelos de linguagem. Os arquivos, originalmente em formato PDF, passaram por processo de extração textual (*Optical Character Recognition* — OCR, quando necessário) e subsequente pré-processamento. Primeiro foram utilizados os artigos selecionados para referência no momento de produção deste trabalho, mas depois de um primeiro momento viu-se a necessidade também de utilizar livros e demais trabalhos citados nos artigos já selecionados anteriormente.

O conteúdo resultante foi segmentado em blocos de até 4096 tokens, para manter o tamanho ideal de contexto do modelo selecionado, priorizando divisões por parágrafos e sentenças, e estruturado no formato `JSONL`. O conjunto final totalizou 873 instâncias no padrão instrução–resposta, adotando a sintaxe usual em modelos instruídos:

```
[INST] instrução [/INST] resposta
```

3.2 Métodos

3.2.1 Pré-processamento dos Dados

O pipeline de pré-processamento incluiu:

1. Extração textual dos PDFs, com detecção e tratamento de arquivos criptografados ou baseados em imagem.
2. Limpeza do texto, removendo espaços duplicados, caracteres de controle e corrigindo hifens de quebra de linha.
3. Segmentação em blocos de até 4096 tokens, priorizando parágrafos e, quando necessário, sentenças.
4. Estruturação no formato `JSONL`, totalizando 873 exemplos no padrão instrução–resposta.

3.2.2 Configuração do Modelo

O modelo base utilizado foi o **Mistral-7B-v0.3**, carregado localmente. A adaptação foi realizada por meio da técnica **LoRA** (*Low-Rank Adaptation*), aplicada às camadas de projeção de atenção (`q_proj`, `v_proj`). Nessa abordagem, pequenas matrizes de baixa dimensão são adicionadas aos pesos originais do modelo, permitindo ajustar o comportamento da rede sem modificar diretamente todos os parâmetros do modelo base (HU *et al.*, 2021; DETTMERS *et al.*, 2023). Os principais hiperparâmetros utilizados foram:

- Rank (r) = 16: representa a dimensão das matrizes de baixa ordem utilizadas para aproximar a atualização dos pesos do modelo. Valores menores reduzem o número de parâmetros treináveis e o consumo de memória, enquanto valores maiores aumentam a capacidade de adaptação do modelo. O valor $r = 16$ é amplamente utilizado na literatura por oferecer um bom equilíbrio entre eficiência computacional e capacidade de representação.
- Alpha = 32: corresponde ao fator de escala aplicado às atualizações produzidas pelas matrizes LoRA. Esse parâmetro controla a intensidade da adaptação introduzida no modelo. Um valor maior permite maior influência das atualizações sobre os pesos originais, enquanto valores muito elevados podem causar instabilidade no treinamento. O valor 32 é frequentemente adotado como configuração padrão por proporcionar estabilidade e boa capacidade de adaptação.
- Dropout = 0,05: define a taxa de regularização aplicada às camadas LoRA durante o treinamento. O objetivo é reduzir o risco de *overfitting*, especialmente quando o conjunto de dados de treinamento é relativamente pequeno. Um valor de 0,05

introduz uma leve regularização sem comprometer a capacidade de aprendizado do modelo.

Esses valores são comumente adotados em experimentos com técnicas de *parameter-efficient fine-tuning*, pois proporcionam um equilíbrio adequado entre desempenho do modelo, estabilidade de treinamento e eficiência computacional.

Adicionalmente, empregou-se **quantização em 4 bits** (`nf4`), por meio da biblioteca `BitsAndBytes`, caracterizando o treinamento como do tipo QLoRA. (DETTMERS *et al.*, 2023)

Os principais hiperparâmetros definidos foram:

- Tamanho do lote por dispositivo (*batch size*) = 4: define o número de amostras processadas simultaneamente em cada passo de treinamento. Um valor reduzido foi adotado devido às limitações de memória da GPU utilizada, garantindo estabilidade no treinamento sem exceder a capacidade de VRAM disponível.
- Acumulação de gradientes = 2: permite acumular os gradientes de múltiplos lotes antes da atualização dos pesos do modelo. Na prática, isso simula um tamanho de lote efetivo maior (equivalente a 8 amostras), mantendo baixo consumo de memória.
- Taxa de aprendizado inicial = $1{,}5 \times 10^{-5}$: controla a intensidade das atualizações aplicadas aos parâmetros do modelo durante o treinamento. Valores pequenos são normalmente utilizados em *fine-tuning* de LLMs para evitar alterações bruscas nos pesos previamente treinados.
- Decaimento de peso = 0,01: atua como um mecanismo de regularização, penalizando valores excessivamente grandes nos parâmetros do modelo. Essa técnica ajuda a reduzir o risco de sobreajuste (*overfitting*) durante o treinamento.
- Comprimento máximo da sequência = 2048 tokens: define o tamanho máximo de contexto que o modelo pode processar em cada entrada. Esse valor permite capturar dependências textuais mais longas sem ultrapassar os limites de memória da GPU.
- Número de passos = 1000 (aproximadamente 6,85 épocas): representa o número total de atualizações de gradiente realizadas durante o treinamento. Esse valor foi suficiente para permitir a convergência do modelo dentro do conjunto de dados disponível.
- Precisão mista: FP16: utiliza representação numérica de 16 bits durante o treinamento, reduzindo o consumo de memória e acelerando os cálculos na GPU, sem perdas significativas de desempenho.

- Otimizador: `paged_adamw_8bit`: variante otimizada do algoritmo AdamW que utiliza quantização em 8 bits e gerenciamento paginado de memória. Essa abordagem reduz significativamente o consumo de memória durante o treinamento de modelos de grande escala.

Durante o processo, apenas 6,8 milhões de parâmetros foram ajustados, equivalendo a 0,0939% do total do modelo, preservando a eficiência típica da técnica LoRA. O tempo total de execução foi de aproximadamente 7 horas e 28 minutos, com perda final (*loss*) média de 1,7286.

3.2.3 Procedimentos de Inferência

Após o treinamento, o modelo foi avaliado qualitativamente por meio de inferências diretas no formato padrão:

```
[INST] instrução [/INST]
```

Os parâmetros de geração empregados foram: `temperature = 0.7`, `top_p = 0.9`, penalização por repetição = 1,1 e limite máximo de novos tokens definido conforme a tarefa.

3.3 Avaliação

Embora não tenha sido realizada divisão explícita entre treino, validação e teste, propõe-se, como continuidade do trabalho, a adoção das seguintes métricas de avaliação:

- **Perplexity (PPL)**: métrica padrão em modelos de linguagem, que reflete a capacidade de prever sequências textuais (BROWN *et al.*, 2020).
- **Métricas de similaridade** (BLEU, ROUGE): recomendadas em tarefas de tradução e sumarização (VASWANI *et al.*, 2017).
- **Avaliação qualitativa humana**: comparação das respostas geradas com o conteúdo dos artigos de treino, verificando coerência, fidelidade e ausência de alucinações.
- **Critérios de alinhamento** (*helpfulness, honesty, harmlessness*): inspirados em trabalhos como o InstructGPT, priorizando respostas úteis, verídicas e não prejudiciais (OUYANG *et al.*, 2022).

O presente trabalho realizou apenas a avaliação qualitativa humana, detalhada no capítulo seguinte.

3.3.1 Disponibilidade do Código

O código-fonte completo desenvolvido neste trabalho, incluindo os scripts de extração textual, pré-processamento, segmentação, treinamento e procedimentos de inferência do modelo Mistral-7B com QLoRA, encontra-se disponível publicamente para fins de reprodutibilidade científica no repositório:

https://github.com/AngeloKiser/LLMAplicacation

4 Desenvolvimento

Este capítulo apresenta o processo de execução prática do trabalho, desde a preparação dos dados até a realização do treinamento do modelo Mistral-7B. As etapas descritas refletem o fluxo metodológico adotado, incluindo a extração e limpeza textual, a segmentação em blocos, a construção do dataset no formato JSONL e a adaptação do modelo via *fine-tuning*.

4.1 Preparação dos Dados

Os 26 trabalhos selecionados foram obtidos em formato PDF. Para a extração textual, foi desenvolvido um código em Python baseado na biblioteca `PyPDF2`, com checagem de criptografia e leitura página a página. Esse procedimento garantiu a conversão segura do conteúdo em texto bruto, passível de manipulação posterior.

```python
def extract_pdf_text(pdf_path): # Extrai texto de um PDF de forma segura
    text = ""
    try:
        with open(pdf_path, 'rb') as pdf_file:
            reader = PyPDF2.PdfReader(pdf_file)
            if reader.is_encrypted: # Verifica criptografia
                try:
                    reader.decrypt('')
                except Exception as e:
                    print(f" Erro de descriptografia: {e}")
                    return ""
            for page in reader.pages: # Processa cada página
                page_text = page.extract_text()
                if page_text:
                    text += page_text + "\n"
        return text.strip()
    except Exception as e:
        print(f" Erro fatal: {e}")
        return ""
```

Listing 1: Extração textual

Na etapa de pré-processamento, aplicou-se uma rotina de limpeza automatizada para remover espaços duplicados, caracteres não imprimíveis e hifens inseridos em quebras de linha. Esse processo resultou em textos mais consistentes, reduzindo ruídos que poderiam comprometer a tokenização. O trecho de código a seguir é o responsável por essa limpeza:

```python
def clean_text(text): # Limpeza avançada do texto
    text = re.sub(r'\s+', ' ', text)  # Espaços duplicados
    text = re.sub(r'[\x00-\x1F\x7F-\x9F]', '', text)  # Não imprimíveis
    text = re.sub(r'-\s*\n\s*', '', text)  # Hifens no final da linha
    return text.strip()
```

Listing 2: Limpeza de texto no pré-processamento

Em seguida, o texto foi segmentado em blocos (*chunks*) respeitando o limite máximo de 4096 tokens. A divisão priorizou parágrafos inteiros e, em casos necessários, sentenças, de modo a preservar a coerência semântica. Essa estratégia possibilitou o aproveitamento mais eficiente da janela de contexto do modelo.

```python
def chunk_text(text, tokenizer): # Divide o texto em chunks sem quebrar sentenças
    paragraphs = [p.strip() for p in text.split('\n\n') if p.strip()]
    chunks = []
    current_chunk = ""
    current_length = 0
    for para in paragraphs:
        tokens = tokenizer.encode(para, add_special_tokens=False)
        para_length = len(tokens)
        if para_length > CONFIG['max_seq_length']:
            sentences = re.split(r'(?<=[.!?])\s+', para)
            for sent in sentences:
                sent_tokens = tokenizer.encode(sent, add_special_tokens=False)
                sent_length = len(sent_tokens)
                if current_length + sent_length <= CONFIG['max_seq_length']:
                    current_chunk += sent + " "
                    current_length += sent_length
                else:
                    if current_chunk:
                        chunks.append(current_chunk.strip())
                    current_chunk = sent + " "
                    current_length = sent_length
        else:
            if current_length + para_length <= CONFIG['max_seq_length']:
                current_chunk += para + "\n\n"
                current_length += para_length
            else:
                if current_chunk:
                    chunks.append(current_chunk.strip())
                current_chunk = para + "\n\n"
                current_length = para_length
    if current_chunk:
        chunks.append(current_chunk.strip())
    return chunks
```

Listing 3: Segmentação do texto

Por fim, os blocos foram estruturados em um arquivo no formato `JSONL`, seguindo a sintaxe recomendada para a variante *Instruct* do Mistral-7B. O formato adotado obedeceu ao padrão:

```
[INST] instrução [/INST] resposta
```

De forma periódica, foram inseridos exemplos artificiais de instruções como:

```
1 {"text": "[INST] Analise este texto acadêmico: Texto acadêmico inserido aqui...
  ↪  [/INST]"}
```

Listing 4: Exemplo de instâncias JSONL

O resultado foi um dataset final contendo 873 instâncias de treino.

4.2 Treinamento do Modelo

A etapa de treinamento exigiu a configuração completa do ambiente. Inicialmente, foi tentado o uso de uma GPU NVIDIA RTX 1000 ADA, com 6 GB de memória VRAM, mas esse recurso mostrou-se insuficiente para o processo de ajuste de parâmetros, ocasionando erros de memória e interrupções.

Após ajustes nos hiperparâmetros e reestruturação do código para otimização do consumo de GPU, foi possível realizar o treinamento de forma estável em uma GPU NVIDIA RTX A4000, com 16 GB de VRAM. Esse ambiente, detalhado no capítulo de Materiais e Métodos, ofereceu a capacidade necessária para suportar o processo de *fine-tuning*.

O modelo base utilizado foi o **Mistral-7B-v0.3**, adaptado por meio da técnica *Low-Rank Adaptation* (LoRA), combinada com quantização em 4 bits (*QLoRA*).

Os hiperparâmetros incluíram: *batch size* de 4 por dispositivo, acumulação de gradientes igual a 2, taxa de aprendizado inicial de 1.5e(-5), decaimento de peso de 0,01 e comprimento máximo de sequência de 2048 tokens.

Apenas 6,8 milhões de parâmetros (0,0939% do total do modelo) foram efetivamente ajustados, preservando a eficiência do treinamento.

O treinamento foi conduzido por 1000 passos, equivalentes a aproximadamente 6,8 épocas sobre o *dataset* de 873 exemplos. O processo totalizou cerca de 7 horas e 28 minutos de execução, resultando em uma perda média final de 1,7286. Durante a execução, observou-se estabilidade no valor da função de perda, sem oscilações significativas, o que indicou convergência adequada frente ao conjunto de dados empregado.

4.3 Inferência e Testes

Concluída a etapa de treinamento, o modelo foi submetido a testes práticos de inferência, com o objetivo de avaliar qualitativamente a adequação das respostas. As entradas seguiram o formato padronizado `[INST] pergunta [/INST]`, em conformidade com a arquitetura Instruct do Mistral.

As perguntas utilizadas foram elaboradas a partir dos artigos empregados no treinamento, de modo a explorar temas fundamentais como a arquitetura Transformer, o surgimento dos modelos GPT, as capacidades emergentes em grandes modelos de linguagem, o alinhamento via *Reinforcement Learning from Human Feedback* (RLHF), as oportunidades e riscos associados aos modelos de fundação, e os desafios atuais descritos em surveys recentes.

O modelo foi testado lado a lado com a versão base (pré-treinada), permitindo comparação direta. Em diversos casos, observou-se que o modelo base apresentou respostas genéricas, pouco direcionadas ao conteúdo das questões, enquanto o modelo treinado produziu respostas mais densas e repletas de terminologia técnica retirada da literatura acadêmica.

5 Resultados e Limitações

Nesta seção, apresentamos os resultados do treinamento do modelo de linguagem e discutimos suas principais limitações. A análise contempla os resultados quantitativos, a avaliação qualitativa das respostas a seis perguntas de referência (Q1–Q6) e, por fim, as limitações identificadas.

5.1 Resultados Quantitativos

O treinamento foi realizado em uma GPU *NVIDIA RTX A4000* com 16 GB de memória, utilizando a técnica *LoRA*, que possibilita ajustar apenas uma pequena fração dos parâmetros do modelo base.

Foram ajustados aproximadamente 6,8 milhões de parâmetros, o que corresponde a 0,0939% do total de 7,25 bilhões. O processo totalizou cerca de 7 horas e 28 minutos de execução, com 1000 passos de otimização e aproximadamente 6,85 épocas. A taxa de aprendizado iniciou em 1.5e(-5) e decaiu progressivamente até 1e(-8). O valor da função de perda (*loss*) reduziu-se de 1,78 para 1,72, de forma estável e consistente, indicando que o modelo aprendeu padrões relevantes dentro das condições de hardware disponíveis.

5.2 Avaliação Qualitativa das Respostas

A comparação entre o modelo sem treinamento e o modelo após ajuste fino foi realizada a partir de seis questões de referência.

Na **Questão 1**, sobre a arquitetura Transformer e o mecanismo de autoatenção, o modelo sem treinamento apresentou uma resposta superficial, mas ainda explicativa. O modelo treinado, por outro lado, regrediu: retornou apenas uma lista de perguntas em vez de fornecer a explicação solicitada.

Na **Questão 2**, referente ao GPT-1, o modelo sem treinamento mencionou apenas o termo *autoaprendizado*, de forma vaga. Já o modelo treinado apresentou melhora significativa, explicando corretamente a inovação do pré-treinamento generativo seguido de *fine-tuning*, com referência a Radford et al. (2018).

Na **Questão 3**, sobre habilidades emergentes no GPT-3, o modelo sem treinamento forneceu uma resposta extensa, mas especulativa e imprecisa. O modelo treinado, em contraste, definiu o conceito de forma correta, incluindo exemplos como *in-context learning*, em conformidade com Brown et al. (2020).

Na **Questão 4**, acerca do RLHF, o modelo sem treinamento produziu uma resposta confusa e com erro conceitual. O modelo treinado, por sua vez, estruturou adequadamente as três etapas do processo — *supervised fine-tuning*, *reward modeling* e *policy optimization* — com referências a Christiano et al. (2017) e Ouyang et al. (2022).

Na **Questão 5**, que tratava das oportunidades e riscos dos *foundation models*, o modelo sem treinamento respondeu de forma informal, restringindo-se ao ChatGPT. O modelo treinado apresentou uma análise conceitual mais consistente, listando riscos como viés e custo ambiental, além de oportunidades de transferência, com base em Bommasani et al. (2021).

Por fim, na **Questão 6**, sobre os desafios atuais dos grandes modelos de linguagem, a resposta do modelo sem treinamento foi incoerente e sem exemplos concretos. O modelo treinado listou corretamente os principais desafios apontados em surveys recentes, como escassez de dados, custo computacional, viés, robustez, segurança e governança (Zhao et al., 2023; Bommasani et al., 2021).

De forma geral, em cinco das seis questões o modelo treinado apresentou algum progresso, oferecendo respostas mais estruturadas e fundamentadas em literatura acadêmica. Apenas na Q1 houve piora, evidenciando a influência negativa de alguns exemplos de treinamento.

5.3 Limitações

Apesar dos avanços, o estudo revelou limitações importantes:

1. **Qualidade dos dados de treinamento**: o desempenho inadequado na Q1 mostrou que alguns exemplos induziram o modelo a listar perguntas em vez de explicar conceitos. Isso evidencia a necessidade de um pré-processo mais rigoroso além de um volume de dados maior.

2. **Tempo de treinamento**: a execução levou mais de sete horas, evidenciando o custo computacional do processo, mesmo em hardware intermediário.

3. **Melhorias pontuais**: como apenas uma fração reduzida dos parâmetros foi ajustada, os ganhos foram localizados em áreas específicas, não se estendendo de maneira uniforme a todos os tópicos avaliados.

Essa análise mostra que o ajuste fino trouxe alguns ganhos, mas também revelou a importância da qualidade dos dados e da escolha cuidadosa das estratégias de treinamento para evitar limitações semelhantes.

6 Considerações Finais

Este trabalho explorou os fundamentos, a evolução e as aplicações dos Modelos de Linguagem de Larga Escala (LLMs), culminando em um experimento prático de especialização do modelo Mistral-7B-v0.3. A investigação confirmou o impacto transformador da arquitetura Transformer e de paradigmas como o pré-treinamento com ajuste fino, que deram origem aos poderosos modelos de fundação atuais.

A instanciação prática, realizada com a técnica de otimização QLoRA, validou a viabilidade de treinar LLMs em ambientes com recursos limitados. O ajuste de uma ínfima parcela dos parâmetros (0,0939%) resultou em ganhos qualitativos sutís na maioria das questões avaliadas, com respostas mais semelhantes estruturalmente, coerentes e bem fundamentadas. A piora em uma única questão, contudo, salientou a dependência crítica da qualidade dos dados de treinamento.

Conclui-se que os LLMs são peças centrais da infraestrutura de IA moderna, com potencial de impacto em setores cruciais. No entanto, seu avanço deve ser acompanhado pelo enfrentamento de desafios éticos e técnicos, como vieses, custos e governança. Este estudo serve como provocação para entender que técnicas eficientes democratizam o acesso à pesquisa de ponta, permitindo que instituições acadêmicas contribuam significativamente para o campo.

A análise desenvolvida ao longo deste trabalho evidencia que os modelos baseados na arquitetura Transformer permanecem como o paradigma dominante na área de modelos de linguagem de larga escala. Contudo, observa-se que o campo ainda se encontra em uma fase de otimização estrutural dessa arquitetura, marcada por aprimoramentos incrementais voltados à eficiência computacional, ampliação de contexto, redução de custo energético e melhoria de alinhamento.

Inovações recentes, como técnicas de parameter-efficient fine-tuning, quantização, arquiteturas do tipo Mixture of Experts e mecanismos mais eficientes de atenção, indicam que a evolução atual ocorre predominantemente por refinamentos internos do próprio modelo Transformer, e não por sua substituição estrutural.

Paralelamente, observa-se o surgimento de propostas alternativas, como modelos baseados em State Space Models, arquiteturas híbridas recorrentes e variações que buscam reduzir a complexidade quadrática da atenção. Embora essas abordagens apresentem resultados promissores em termos de escalabilidade e eficiência, ainda não existe evidência consolidada de que representem uma ruptura paradigmática capaz de substituir definitivamente os Transformers.

Nesse contexto, sugerem-se como trabalhos futuros: (i) a investigação comparativa entre diferentes arquiteturas emergentes e modelos baseados em Transformer; (ii) a análise do impacto dessas alternativas em tarefas de especialização temática; (iii) a avaliação de eficiência computacional em cenários com recursos limitados; e (iv) o estudo de integração entre modelos de linguagem e mecanismos externos de memória e recuperação de informação.

Conclui-se que o campo encontra-se em um momento de transição exploratória, no qual novas arquiteturas estão sendo testadas, mas ainda sem uma perspectiva clara de consolidação como sucessoras definitivas do paradigma Transformer. Estudos experimentais adicionais serão fundamentais para compreender se tais propostas representam evolução incremental ou uma mudança estrutural mais profunda na construção de sistemas de inteligência artificial generativa.

Pesquisas futuras devem priorizar a expansão dos conjuntos de dados, a adoção de métricas de avaliação mais robustas e a investigação de temas como interpretabilidade e alinhamento ético, pavimentando o caminho para modelos não apenas mais capazes, mas também mais confiáveis e alinhados com o benefício para a sociedade.

Referências

BOMMASANI, Rishi *et al.* *On the Opportunities and Risks of Foundation Models*. 2021. Disponível em: http://arxiv.org/abs/2108.07258v3. Citado 6 vezes nas páginas 3, 4, 13–16.

BROWN, Tom B. *et al.* Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165*, 2020. Disponível em: http://arxiv.org/abs/2005.14165v4. Citado 6 vezes nas páginas 3, 4, 8, 13, 14, 20.

CHRISTIANO, Paul F. *et al.* Deep Reinforcement Learning from Human Preferences. *In:* ADVANCES in Neural Information Processing Systems (NeurIPS). 2017. Disponível em: https://arxiv.org/abs/1706.03741. Citado 3 vezes nas páginas 8, 15.

DETTMERS, Tim *et al.* QLoRA: Efficient Finetuning of Quantized LLMs. *In:* ADVANCES in Neural Information Processing Systems. 2023. Citado 3 vezes nas páginas 14, 18, 19.

DEVLIN, Jacob *et al.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *In:* PROCEEDINGS of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT). 2019. p. 4171–4186. Disponível em: http://arxiv.org/abs/1810.04805v2. Citado 7 vezes nas páginas 3, 4, 8, 10, 13, 14.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735. Citado 3 vezes nas páginas 10, 12.

HU, Edward J. *et al.* LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2021. Citado 2 vezes nas páginas 14, 18.

JELINEK, Frederick. *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press, 1997. ISBN 978-0-262-10066-9. Citado 1 vez na página 10.

MIKOLOV, Tomas *et al.* Distributed Representations of Words and Phrases and their Compositionality, 2013. Citado 1 vez na página 13.

OUYANG, Long *et al.* Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022. Disponível em: http://arxiv.org/abs/2203.02155v1. Citado 7 vezes nas páginas 3, 4, 8, 16, 20.

RABINER, Lawrence R. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, v. 77, n. 2, p. 257–286, 1989. DOI: 10.1109/5.18626. Citado 1 vez na página 10.

RADFORD, Alec *et al.* Improving Language Understanding by Generative Pre-Training. *arXiv preprint arXiv:1801.06146*, 2018. Disponível em: http://arxiv.org/abs/1801.06146. Citado 7 vezes nas páginas 3, 4, 8, 10, 13, 14.

SPARCK JONES, Karen. Natural Language Processing: A Historical Review. *In:* ZAMPOLLI, Antonio; CALZOLARI, Nicoletta; PALMER, Martha (ed.). *Current Issues in Computational Linguistics: In Honour of Don Walker*. Pisa; Dordrecht: Giardini / Kluwer, 1994. v. 9-10. (Linguistica Computazionale). p. 1–27. Citado 1 vez na página 10.

TOUVRON, Hugo *et al.* LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. Disponível em: http://arxiv.org/abs/2302.13971v1. Citado 1 vez na página 8.

VASWANI, Ashish *et al.* Attention is all you need. *In:* ADVANCES in Neural Information Processing Systems (NeurIPS). 2017. v. 30. Disponível em: http://arxiv.org/abs/1706.03741v4. Citado 11 vezes nas páginas 3, 4, 8, 10–12, 14, 20.

WANG, Yufei *et al.* Aligning Large Language Models with Human: A Survey. *arXiv preprint arXiv:2307.12966*, 2023. Disponível em: http://arxiv.org/abs/2307.12966v1. Citado 2 vezes na página 16.

ZHAO, Wayne Xin *et al.* A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223*, 2023. Disponível em: http://arxiv.org/abs/2303.18223. Citado 1 vez na página 16.

Glossário

Autoatenção (self-attention) Mecanismo que permite a cada token de uma sequência ponderar sua relação com todos os demais, capturando dependências de curto e longo alcance com alta paralelização.

Transformer Arquitetura baseada exclusivamente em atenção, sem recorrência nem convoluções, composta por blocos com camadas de atenção multi-cabeças, conexões residuais, normalização e redes feed-forward posicionais.

LoRA (Low-Rank Adaptation) Técnica de ajuste fino eficiente que insere adaptadores de baixo posto em camadas do modelo, permitindo treinar apenas uma fração dos parâmetros com menor custo computacional e de memória.

Fine-tuning (ajuste fino) Etapa de especialização de um modelo pré-treinado em um conjunto de dados específico, ajustando parâmetros (ou adaptadores) para um domínio ou tarefa.

Pré-treinamento (pre-training) Treinamento inicial em grandes corpus (tipicamente não rotulados) para aprender representações gerais que serão reutilizadas em múltiplas tarefas.

RLHF (Reinforcement Learning from Human Feedback) Procedimento de alinhamento que envolve (i) ajuste supervisionado inicial, (ii) modelagem de recompensa a partir de preferências humanas e (iii) otimização da política para maximizar a recompensa aprendida.

Função de perda (loss) Medida numérica do erro entre a saída do modelo e o alvo esperado; sua redução contínua durante o treinamento indica aprendizado efetivo.

Norma do gradiente (gradient norm) Magnitude do vetor de gradientes; valores estáveis sugerem treinamento controlado, enquanto valores excessivos podem indicar explosão de gradientes.

Taxa de aprendizado (learning rate) Hiperparâmetro que controla o tamanho do passo nas atualizações dos parâmetros; esquemas com decréscimo tendem a favorecer estabilidade na convergência.

Época (epoch) Passagem completa de todo o conjunto de treinamento pelo modelo.

Passo de otimização (step) Atualização dos parâmetros com base em um lote (*batch*) de exemplos; diversos passos compõem uma época.

Parâmetros treináveis Subconjunto de parâmetros efetivamente atualizado no treinamento (em LoRA, tipicamente uma fração muito pequena do total).

GPU (Graphics Processing Unit) Unidade de processamento massivamente paralela que acelera o treinamento e a inferência de redes neurais.

VRAM (Video RAM) Memória disponível na GPU; limita o tamanho do modelo, do *batch* e a resolução numérica viáveis no treinamento.

Modelos de fundação (foundation models) Modelos de larga escala, pré-treinados de forma ampla e reutilizáveis em diversas tarefas por adaptação leve; combinam alto potencial de transferência com desafios de viés, custo e governança.

Capacidades emergentes Habilidades que aparecem em modelos maiores e não se manifestam em versões menores, como aprendizado no contexto e generalização zero-*shot*.

Aprendizado no contexto (in-context learning) Capacidade de resolver novas tarefas a partir de exemplos contidos no próprio *prompt*, sem atualização explícita de parâmetros.

Mistral-7B Modelo de linguagem de aproximadamente 7 bilhões de parâmetros utilizado como base experimental para ajuste fino neste trabalho.