CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

MODELOS DE LINGUAGEM DE GRANDE PORTE PARA A IDENTIFICAÇÃO DE CONLUIO EM LICITAÇÕES

Jorge Nelson de Souza Pavão

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Ciência da Computação, Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, como parte dos requisitos necessários à obtenção do grau de mestre.

Orientadores:
Kele Teixeira Belloze
Diego Nunes Brandão

Rio de Janeiro,
Junho de 2026

Modelos de Linguagem de Grande Porte para a identificação de conluio em licitações

Proposta de dissertação de Mestrado em Ciência da Computação, Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, CEFET/RJ.

Jorge Nelson de Souza Pavão

Aprovada por:

Presidente, Prof. Kele Teixeira Belloze, D.Sc. (orientador)

Diego Nunes Brandão, D.Sc. (coorientador)

Eduardo Bezerra da Silva, D.Sc.

Raissa dos Santos Barcellos, D.Sc.

Rio de Janeiro,

Junho de 2026

Ficha catalográfica elaborada pela Biblioteca Central do CEFET/RJ

P337 Pavão, Jorge Nelson de Souza
Modelos de linguagem de grande porte para a identificação de conluio em licitações / Jorge Nelson de Souza Pavão – 2026.
69f : il. (algumas color.) , enc.

Dissertação (Mestrado). Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, 2026.
Bibliografia : f. 63-69.
Orientadora: Kele Teixeira Belloze.
Orientador: Diego Nunes Brandão.

1. Aprendizado de máquina. 2. Processamento de linguagem natural (Ciência da computação). I. Belloze, Kele Teixeira (Orient.). II. Brandão, Diego Nunes (Orient.). III. Título.

CDD 006.31

Elaborada pelo bibliotecário Leandro Mota de Menezes – CRB/7 nº 5281

DEDICATÓRIA

Este trabalho é dedicado à Nina, minha filha, e
ao Boni, meu cachorro, que esteve ao meu lado
nas muitas noites de trabalho nesta pesquisa.

AGRADECIMENTOS

Agradeço à Carolina, minha esposa, pela compreensão quanto ao tempo dedicado ao mestrado e aos meus orientadores, Kele e Diego, que, com suas experiências e conhecimentos, muito me ajudaram neste trabalho.

RESUMO

Modelos de Linguagem de Grande Porte para a identificação de conluio em licitações

Jorge Nelson de Souza Pavão

Orientadores:
Kele Teixeira Belloze
Diego Nunes Brandão

Resumo da Dissertação submetida ao Programa de Pós-graduação em Ciência da Computação do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ como parte dos requisitos necessários à obtenção do grau de mestre.

A prática de conluio em licitações públicas, caracterizada por acordos ilícitos entre empresas concorrentes, gera prejuízos econômicos relevantes, reduz a eficiência das contratações e compromete a confiança nas instituições. Diversos trabalhos têm empregado algoritmos de aprendizado de máquina e variáveis estatísticas para identificar indícios desse tipo de fraude, porém ainda persistem desafios como escassez de dados rotulados, dificuldade de generalização e desempenho inconsistente entre diferentes bases de dados. Neste contexto, esta dissertação investiga o potencial de Modelos de Linguagem de Grande Porte (*Large Language Models* - LLMs) na detecção de indícios de conluio a partir de dados numéricos de processos licitatórios. Foram avaliados modelos de diferentes fornecedores por meio de duas estratégias: engenharia de *prompt* e *fine-tuning* supervisionado. Os experimentos foram conduzidos com três conjuntos de dados reais — Brasil, Estados Unidos e Suíça — e os resultados foram comparados com os de algoritmos tradicionais reportados na literatura. Os resultados mostram que a engenharia de *prompt* não produziu desempenho satisfatório. Com *fine-tuning*, por sua vez, os LLMs superaram os algoritmos tradicionais em apenas um dos três cenários avaliados. Considerando conjuntamente desempenho, custo e complexidade de implementação, conclui-se que, no momento atual, os algoritmos tradicionais apresentam melhor relação custo-benefício na tarefa avaliada, sendo que os LLMs podem atuar como alternativa complementar em cenários específicos nos quais os métodos clássicos não alcancem resultados satisfatórios ou situações que envolvam dados textuais.

Palavras-chave: Conluio, Large Language Model, Engenharia de Prompt, Fine-tuning.

Rio de Janeiro,

Junho de 2026

ABSTRACT

Modelos de Linguagem de Grande Porte para a identificação de conluio em licitações

Jorge Nelson de Souza Pavão

Advisors:
Kele Teixeira Belloze
Diego Nunes Brandão

Abstract of dissertation submitted to Programa de Pós-graduação em Ciência da Computação - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ as partial fulfillment of the requirements for the degree of master.

The practice of collusion in public procurement, characterized by illicit agreements between competing firms, generates significant economic losses, reduces the efficiency of public contracting, and undermines trust in institutions. Several studies have employed machine learning algorithms and statistical variables to identify indications of this type of fraud; however, challenges such as the scarcity of labeled data, difficulties in generalization, and inconsistent performance across different datasets still persist. In this context, this dissertation investigates the potential of Large Language Models (LLMs) for detecting indications of collusion based on numerical data from procurement processes. Models from different providers were evaluated using two strategies: prompt engineering and supervised fine-tuning. The experiments were conducted on three real-world datasets — Brazil, the United States, and Switzerland — and the results were compared with those of traditional algorithms reported in the literature. The findings show that prompt engineering did not produce satisfactory performance. With fine-tuning, the LLMs outperformed traditional algorithms in only one of the three evaluated scenarios. Considering performance, cost, and implementation complexity jointly, it is concluded that, at the present time, traditional algorithms offer a better cost-benefit ratio for the evaluated task, while LLMs may serve as a complementary alternative in specific scenarios where classical methods fail to achieve satisfactory results or in situations involving textual data.

Key-words: Collusion, Large Language Model, Prompt Engineering, Fine-tuning.

Rio de Janeiro,

Junho de 2026

Sumário

I Introdução **1**

I.1 Contextualização e problema 1

I.2 Objetivo 3

I.3 Metodologia e resultados 3

I.4 Organização do trabalho 4

II Referencial Teórico **5**

II.1 Modelos de Linguagem de Grande Porte 5

II.2 Engenharia de prompt 6

II.3 Fine tuning 9

II.3.1 PEFT 11

II.3.2 QLoRA 12

II.4 Métricas 14

II.5 Testes Estatísticos 15

II.5.1 Teste t de Student 15

II.5.2 Teste de Wilcoxon 16

III Trabalhos relacionados **18**

III.1 Aprendizado de máquina para identificação de conluio 18

III.2 Large Language Model para temas semelhantes 20

III.3 Artigo de referência 20

IV Metodologia **22**

IV.1 Conjuntos de dados 23

IV.2 Engenharia de Prompt 29

IV.3 Fine-tuning 30

IV.4 Métricas 31

IV.5 Testes estatísticos 31

V Experimentos e resultados **33**

V.1 Reexecução dos experimentos do artigo de referência. 33
V.2 Engenharia de prompt 35
V.2.1 ChatGPT 36
V.2.2 DeepSeek e Mistral 39
V.3 Fine-tuning 43
V.3.1 Divisão dos dados 43
V.3.2 Descrição do processo de fine-tuning 44
V.3.3 Experimentos com o conjunto de dados dos Estados Unidos 48
V.3.4 Experimentos com o conjunto de dados do Brasil 50
V.3.5 Experimentos com o conjunto de dados da Suíça 55
V.4 Discussão dos resultados 56

VI Conclusão **59**

Referências **63**

Lista de Figuras

II.1 Componentes de um prompt. Fonte: O autor (2026) 7

II.2 Principais técnicas de engenharia de prompt. Fonte: O autor (2026) 8

II.3 Evolução das técnicas de fine-tuning. Fonte: Wu et al. [2025] 10

II.4 Arquitetura dos adapters. Figura adaptada de Alammar and Grootendorst [2024] 12

II.5 Matrizes LoRA. Figura adaptada de Alammar and Grootendorst [2024] 13

II.6 QLoRA. Figura adaptada de Alammar and Grootendorst [2024] 14

IV.1 Metodologia para avaliação dos LLMs na identificação de indícios de conluio. Fonte: O autor (2026) 23

IV.2 Comparação entre a quantidade de registros da classe positiva e negativa. Fonte: O autor (2026) 25

IV.3 Distribuição do número de lances por pregão em três contextos: Brasil, Estados Unidos e Suíça Ticino. Fonte: O autor (2026) 25

IV.4 Distribuição do valor dos lances. Fonte: O autor (2026) 26

IV.5 Correlação entre os atributos do conjunto de dados da Suíça. Fonte: O autor (2026) 27

IV.6 Correlação entre os atributos do conjunto de dados dos Estados Unidos. Fonte: O autor (2026) 28

IV.7 Correlação entre os atributos do conjunto de dados do Brasil. Fonte: O autor (2026) 29

IV.8 Descrição dos experimentos com fine-tuning. Fonte: O autor (2026) 31

V.1 Exemplo de *prompt*. Fonte: O autor (2026) 36

V.2 Comparação do desempenho de acordo com o formato de envio dos dados. Fonte: O autor (2026) 39

V.3 Média do F1 Score dos experimentos com engenharia de *prompt* nos dados do Brasil. As colunas em cinza representam os algoritmos avaliados no artigo de referência. Fonte: O autor (2026) 40

V.4 Exemplo de resposta do ChatGPT 4o Mini no experimento com engenharia de *prompt*. 40

V.5 Exemplo de resposta do Mistral-7B-Instruct-v0.2 no experimento com engenharia de *prompt*. 41

V.6 F1 Score médio dos experimentos com engenharia de *prompt* nos dados dos Estados Unidos. As colunas em cinza representam os algoritmos avaliados no artigo de referência. Fonte: O autor (2026) 41
V.7 Média do F1 Score dos experimentos com engenharia de *prompt* nos dados da Suíça. Fonte: O autor (2026) 42
V.8 Divisão dos dados para o uso no *fine-tuning*. Fonte: O autor (2026) 43
V.9 Funcionalidade da OpenAI utilizada para realização do fine-tuning. Fonte: Interface da plataforma OpenAI, captura realizada pelo autor em 2025. 45
V.10 Comparação do F1 Score dos LLMs com os do artigo de referência utilizando o conjunto de dados dos Estados Unidos. Fonte: O autor (2026) 49
V.11 Comparação do F1 Score dos LLMs com o da nova execução do Gradient Boosting no conjunto de dados dos Estados Unidos. Fonte: O autor (2026) 50
V.12 Comparação do F1 Score dos LLMs com os obtidos no artigo de referência utilizando o conjunto de dados do Brasil. Fonte: O autor (2026) 51
V.13 Distribuição dos F1 Scores alcançados nos experimentos. Fonte: O autor (2026) 51
V.14 Comparação do F1 Score dos LLMs com os obtidos na reexecução dos algoritmos com a divisão dos dados sendo feita em treino, validação e teste. Fonte: O autor (2026) 52
V.15 F1 Score do experimento com a atribuição de peso para as classes durante o treinamento do Mistral 7B. As barras de erro representam ± 1 desvio padrão. Fonte: O autor (2026) 54
V.16 Comparação do F1 Score dos LLMs com os obtidos no artigo de referência utilizando o conjunto de dados da Suíça. Fonte: O autor (2026) 55

Lista de Tabelas

III.1 *Strings* de buscas usadas na pesquisa da literatura 18

IV.1 Campos disponíveis nos conjuntos de dados do Brasil, dos Estados Unidos e da Suíça 24
IV.2 Características dos conjuntos de dados 24

V.1 F1 Score por algoritmo nos conjuntos Brasil, Estados Unidos e Suíça Ticino 33
V.2 Comparação do melhor F1 Score obtido no artigo de referência com os valores alcançados com a engenharia de prompt. 37
V.3 Resultados do F1 Score no conjunto da Suíça com o modelo O4-Mini 38
V.4 Hiperparâmetros do fine-tuning realizado no ChatGPT 4o Mini 46
V.5 Hiperparâmetros do *fine-tuning* feito nos modelos DeepSeek-R1-Distill-Qwen-14B e Mistral-7B-Instruct-v0.2 48
V.6 Melhor LLM em cada um dos experimentos. 56

Lista de Acrônimos

BERT Bidirectional Encoder Representations from Transformers.

CADE Conselho de Administração de Defesa Econômica.

GPT Generative Pre-trained Transformer.

KNN K-Nearest Neighbors.

LLMs Large Language Models.

OCDE Organização para a Cooperação e Desenvolvimento Econômico.

PEFT Parameter-Efficient Fine-Tuning.

QLoRA Quantized Low-Rank Adaptation.

RAG Retrieval-augmented Generation.

Capítulo I Introdução

I.1 Contextualização e problema

O conluio em licitação ocorre quando empresas que deveriam competir de forma legítima entre si conspiram secretamente para restringir a concorrência dos processos de contratação de bens e serviços pela Administração Pública [OCDE, 2021]. Segundo a Organização para a Cooperação e Desenvolvimento Econômico - OCDE, estudos indicam que essa prática pode aumentar em média entre 10% e 20% o preço final [OCDE, 2021], sendo que há casos em que esse percentual pode ser ainda maior. Tal constatação pode ser observada, por exemplo, quando o Conselho Administrativo de Defesa Econômica - CADE identificou um sobrepreço de 25% em uma licitação de portas de segurança giratórias [OCDE, 2021]. Bolotova [2009] sugere que o aumento de preço pode chegar a 50%.

O impacto causado pelo conluio entre empresas fica ainda mais evidente quando é considerado o volume financeiro das contratações públicas. De acordo com dados do Portal de Transparência do Governo Federal[1], em 2025 o valor total dos contratos firmados chegou a R$ 139,36 bilhões. Ressalta-se que esses valores referem-se apenas às contratações do Governo Federal, sem considerar estados e municípios. É razoável inferir que ao incluir no cálculo os 26 estados, o Distrito Federal, bem como os 5.569 municípios brasileiros, esse valor, que já é elevado, passa a ser significativamente maior.

Conluio em contratações públicas é um tipo de fraude que ocorre em todo o mundo. No Brasil, o caso recente que ficou mais conhecido foi o descoberto pela Operação Lava Jato, em que empreiteiras formaram cartéis para fraudar licitações e dividir contratos em troca de propinas a agentes públicos [Bechara and Goldschmidt, 2020]. Nos Estados Unidos, diversas empresas farmacêuticas foram acusadas de combinar preços, dividir mercados e fraudar concorrências públicas [Clark et al., 2022]. Na Coreia do Sul, empresas foram multadas por formação de cartel em contratos de fornecimento de combustível militar e obras públicas[2]. Na Espanha, empresas foram multadas em aproximadamente 127 milhões de euros por conluio em licitações de sinalização ferroviária[3]. No Japão, a maior agência

[1]Informação obtida em https://portaldatransparencia.gov.br/contratos?ano=2024. Último acesso: 14/04/2026

[2]Disponível em https://www.reuters.com/article/business/three-skorean-firms-to-plead-guilty-to-rigging-price-of-us-military-base-fuel-idUSKCN1NJ2NX/ . Último acesso em 12/06/2025.

[3]Disponível em https://www.railjournal.com/financial/spanish-rail-infrastructure-cartel-exposed-and-fined-e127-3m/ . Último acesso em 12/06/2025.

de publicidade e outras cinco empresas foram indiciadas por violarem a lei antimonopólio, em um escândalo de corrupção envolvendo alegações de conluio em licitações durante os Jogos Olímpicos de Tóquio[4].

Além do impacto financeiro, o conluio compromete a qualidade dos bens e serviços contratados, reduz a confiança da população nas instituições públicas e favorece a ineficiência na alocação dos recursos.

No Brasil, o combate ao conluio em licitações é conduzido por diferentes órgãos que atuam de forma complementar. O Conselho Administrativo de Defesa Econômica (CADE) é a principal autoridade responsável por investigar e punir infrações à ordem econômica, incluindo a formação de cartéis. No âmbito do controle interno e externo da Administração Pública, destacam-se a Controladoria-Geral da União (CGU), que apura irregularidades e aplica sanções administrativas com base na Lei Anticorrupção, e o Tribunal de Contas da União (TCU), que fiscaliza os contratos públicos e pode aplicar sanções aos agentes envolvidos. Nos estados, o controle interno e externo é realizado, respectivamente, pelas controladorias e tribunais de contas estaduais. Por fim, o Ministério Público tem papel central na esfera criminal, promovendo ações penais contra os envolvidos. A atuação coordenada entre essas instituições é essencial para a detecção, investigação e punição efetiva das práticas colusivas em processos licitatórios.

As estratégias utilizadas pelas empresas participantes do conluio variam desde propostas fictícias ou de cobertura *(cover bidding)*, supressão ou retirada coordenada de propostas, rodízio entre licitantes vencedores *(bid rotation)*, até a divisão de mercados ou lotes entre as empresas [CADE, 2019]. Tais práticas são, muitas vezes, estruturadas de forma sofisticada para dificultar sua detecção e esconder a existência de um acordo prévio entre os concorrentes.

Neste contexto, identificar indícios de conluio durante a licitação não é uma tarefa trivial e, por isso, vários trabalhos têm utilizado algoritmos de aprendizado de máquina com este objetivo. Atualmente, a abordagem mais utilizada é combinar o uso de algoritmos de aprendizado supervisionado com a adição de variáveis estatísticas *(screens)*, como foi feito por Scoralick et al. [2024], Rodríguez et al. [2022] e Imhof and Wallimann [2021].

Outros métodos utilizados foram o uso de redes neurais [Lima, 2021], algoritmos de aprendizado não supervisionado [Velasco et al., 2021], identificação de padrões frequentes [Ramalho et al., 2020] e grafos [Velasco et al., 2021].

É possível perceber que a identificação de conluio em licitações públicas tem sido objeto de estudo por meio de diversas abordagens da ciência de dados e do aprendizado de máquina, porém alguns desafios persistem, como a escassez de dados rotulados para treinamento dos modelos, a dificuldade de generalização dos modelos e o desempenho insuficiente dos algoritmos tradicionais em alguns

[4]Disponível em https://www.theguardian.com/world/2023/feb/28/japan-to-ad-agency-dentsu-indicted-olympics-bid-rigging-scandal . Último acesso em 12/06/2025.

conjuntos de dados. Frente a esses desafios, o uso de Modelos de Linguagem de Grande Porte (LLMs) - *Large Language Models* em inglês - poderia ser uma alternativa para evitar a necessidade de obtenção de uma grande quantidade de dados para treinamento, bem como para obter um nível maior de precisão na identificação de padrões de conluio em dados em que os algoritmos tradicionais não obtêm resultados satisfatórios.

Embora algoritmos de aprendizado supervisionado e não supervisionado, redes neurais, além de técnicas baseadas em variáveis estatísticas tenham sido amplamente explorados, o uso de Modelos de Linguagem de Grande Porte ainda é incipiente nesse contexto. Não foram encontrados estudos que apresentem evidências de que os LLMs sejam capazes de lidar bem com dados numéricos dessa natureza.

Nesse cenário, LLMs surgem como uma possível alternativa promissora. Sua capacidade de lidar com dados textuais e gerar inferências complexas levanta a hipótese de que, mesmo diante de dados predominantemente numéricos, eles possam superar limitações enfrentadas por modelos tradicionais. Logo, explorar seu potencial pode abrir novos caminhos para a detecção automatizada de indícios de conluio em licitações.

I.2 Objetivo

Este trabalho tem como objetivo investigar o potencial dos LLMs na detecção de indícios de conluio em dados numéricos de processos licitatórios, comparando o resultado dos LLMs com as técnicas mais utilizadas segundo a literatura.

I.3 Metodologia e resultados

Para atingir os objetivos, foi avaliado o desempenho, na tarefa de detecção de conluio em licitações, dos modelos ChatGPT 4o Mini[5] e O4 Mini[6], ambos da OpenAI; do modelo DeepSeek-V3.2-Exp[7] da DeepSeek AI e do Mistral-7B-Instruct-v0.2[8] da empresa Mistral AI utilizando-se técnicas de engenharia de *prompt*. Também foram avaliados os modelos ChatGPT 4o Mini, DeepSeek-R1-Distill-Qwen-14B[9] e Mistral-7B-Instruct-v0.2 por meio do *fine-tuning*. Os resultados foram comparados aos alcançados por métodos tradicionais de aprendizado de máquina em trabalhos anteriores. Os dados utilizados foram aqueles disponibilizados no artigo *Collusion detection in public procurement auctions with machine learning algorithms* [Rodríguez et al., 2022], o qual foi selecionado para comparação dos resultados e que aqui será denominado artigo de referência.

[5]https://openai.com/pt-BR/index/gpt-4o-mini-advancing-cost-efficient-intelligence/ . Último acesso em 24/4/2026

[6]https://openai.com/pt-BR/index/introducing-o3-and-o4-mini/ . Último acesso em 24/4/2026

[7]https://api-docs.deepseek.com/news/news250929 . Último acesso em 24/4/2026

[8]https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2 . Último acesso em 24/4/2026

[9]https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-14B . Último acesso em 24/4/2026

Os resultados apontam que os algoritmos tradicionais de aprendizado de máquina permanecem como a alternativa mais indicada para a detecção de indícios de conluio em dados numéricos de licitações, tanto em termos de desempenho quanto de custo-benefício. Nos experimentos realizados, essas abordagens superaram os Modelos de Linguagem de Grande Porte na maior parte dos conjuntos de dados analisados. Observou-se ainda que estratégias baseadas apenas em engenharia de *prompt* não produziram resultados competitivos. Nos testes com *fine-tuning*, os LLMs alcançaram desempenho superior apenas em um dos três cenários avaliados, não se verificando vantagem consistente em relação aos métodos clássicos. Dessa forma, o uso de LLMs mostra-se mais adequado como abordagem complementar, especialmente em contextos específicos nos quais os algoritmos tradicionais não apresentem resultados satisfatórios e em situação que envolvam dados textuais.

I.4 Organização do trabalho

O texto está estruturado da seguinte forma: o Capítulo II apresenta o referencial teórico que foi usado como base deste trabalho, o Capítulo III contém os trabalhos relacionados, que tratam do uso de aprendizado de máquina para identificação de conluio. O Capítulo IV contém a metodologia e no Capítulo V são descritos os experimentos realizados e seus resultados. Por fim, no Capítulo VI encontram-se as conclusões.

Capítulo II Referencial Teórico

Esse capítulo apresenta uma breve revisão sobre modelos de LLM com atenção especial a engenharia de *prompts* e *fine tuning*, processos importantes para obtenção de modelos mais assertivos. Por fim, são apresentadas as métricas avaliativas e os testes estatísticos que auxiliam na análise dos resultados.

II.1 Modelos de Linguagem de Grande Porte

Modelos de Linguagem de Grande Porte - *Large Language Models* em inglês - são modelos de aprendizado profundo treinados com grandes volumes de dados textuais não rotulados para gerar e interpretar linguagem natural com elevado grau de sofisticação [Benzinho et al., 2024] [Trad and Chehab, 2024]. Esses modelos, que contêm bilhões de parâmetros, utilizam, na maioria das vezes, arquiteturas baseadas em *Transformers*, que permitem entender como diferentes partes de um texto se relacionam entre si, mesmo que estejam distantes. Isso ajuda o modelo a compreender melhor o significado completo das frases [Raiaan et al., 2024] e [Berryman and Ziegler, 2024].

O *Transformer* foi introduzido pela primeira vez em Vaswani et al. [2017]. Essa arquitetura revolucionou o campo do Processamento de Linguagem Natural ao abordar as principais limitações de modelos anteriores, como as Redes Neurais Recorrentes, particularmente sua dificuldade em processar dados sequenciais em paralelo e capturar dependências de longo alcance de forma eficaz. *Transformers* dependem fortemente de mecanismos de atenção, especificamente autoatenção (*self-attention*), que permite ao modelo ponderar a importância de diferentes palavras na sequência de entrada. Essa capacidade permite que o modelo compreenda o contexto e as relações entre os *tokens* simultaneamente, independentemente de sua posição no texto [Amaratunga, 2023]. A arquitetura *Transformer* foi projetada para permitir o processamento paralelo de *tokens*, onde cada *token* de entrada pode fluir por seu próprio caminho de computação, acelerando significativamente o treinamento em comparação com modelos sequenciais.

A arquitetura original do *Transformer* consiste em dois componentes principais: um codificador (*encoder*) e um decodificador (*decoder*). O codificador é responsável por processar o texto de entrada, compreendendo seu contexto e significado, enquanto o decodificador é encarregado de gerar o texto de saída [Berryman and Ziegler, 2024]. Esse design, particularmente com a integração

da atenção, superou uma fragilidade importante do modelo sequência-a-sequência anterior que era a incapacidade de compreender o contexto em longas sequências de texto [Berryman and Ziegler, 2024].

LLMs modernos são na maioria das vezes variantes da arquitetura original dos *transformers*, frequentemente usando apenas os blocos do codificador ou apenas os blocos do decodificador. Modelos apenas com codificador (*Encoder-only models*), como o Bidirectional Encoder Representations from Transformers (BERT), destacam-se na compreensão e representação da linguagem, tornando-os adequados para tarefas como classificação de texto ou busca semântica. Modelos apenas com decodificador (*Decoder-only models*), como a família Generative Pre-trained Transformer (GPT), concentram-se principalmente na geração sequencial de texto, prevendo um token por vez com base no contexto precedente [Ozdemir, 2024].

Entre as principais técnicas utilizadas para adaptar os LLMs para tarefas específicas, podemos citar a Engenharia de *Prompt*, o *Fine-tuning*, o *Retrieval-augmented generation* (RAG) e a Destilação de Conhecimento. As duas primeiras serão apresentadas com mais detalhes nas seções II.2 e II.3, respectivamente, por terem sido adotadas neste trabalho.

Retrieval-augmented generation é uma abordagem que combina o uso de LLMs com sistemas de recuperação de informações, com o objetivo de melhorar as respostas geradas pelo modelo. No RAG, quando uma pergunta é recebida, primeiro é utilizado o sistema de recuperação de informações para buscar informações relevantes sobre o tema e fornecê-las ao modelo [Parthasarathy et al., 2024]. Essa técnica é bastante útil, pois além de possibilitar que o modelo consulte informações não disponíveis durante o treinamento, permite que ele tenha acesso a informações sempre atualizadas sem a necessidade de treiná-lo novamente.

Destilação do conhecimento, por sua vez, envolve a transferência do conhecimento de um modelo maior e mais complexo para a criação de um modelo menor e mais eficiente em termos de uso de recursos computacionais [Wu et al., 2025].

II.2 Engenharia de prompt

A engenharia de *prompt* é um tema que tem sido bastante estudado atualmente. Schulhoff et al. [2025] fizeram uma revisão sistemática de publicações sobre o assunto e identificaram 58 diferentes técnicas de *prompts* baseadas em texto. Não foram incluídas na lista técnicas que contemplam outros tipos de dados, como áudio e vídeo.

Mas independentemente da técnica adotada, a qual será discutida posteriormente, é possível identificar componentes de um *prompt* que são comuns a quase todas. A Figura II.1 ilustra tais componentes e suas descrições são apresentadas a seguir.



Figura II.1: Componentes de um prompt. Fonte: O autor (2026)

Diretiva é o conjunto de instruções ou questões a que o LLM deve responder. Em alguns casos, ela pode ser implícita [Schulhoff et al., 2025], como no exemplo a seguir.

Sun: Sol Moon:

Exemplos são amostras de entradas e saídas que demonstram ao modelo o comportamento esperado em uma tarefa. Eles ajudam o LLM a inferir o padrão desejado a partir de casos concretos. O prompt apresentado anteriormente ilustra esse uso.

Formato de saída é a forma na qual se deseja que o modelo responda. No presente trabalho, o formato de saída solicitado ao LLM foi o seguinte:

Responda ‘0’ ou ‘1’ para cada um dos 20 registros, sendo 0 para não e 1 para sim. Use ’,’ como separador. Exemplo do formato da resposta: ‘1,1,0’.

Instruções de estilo é um tipo de formato de saída usado para modificar a resposta estilisticamente, não estruturalmente [Schulhoff et al., 2025]. A seguir, é apresentado um exemplo.

Use uma linguagem simples e acessível a estudantes do ensino médio.

Role também conhecido como *persona* define quem o modelo deve ser ao responder. Embora em alguns casos possa ser confundido com as instruções de estilo, são elementos diferentes. O foco aqui é simular um papel ou identidade específica, como apresenta o exemplo a seguir.

Aja como um advogado experiente em direito constitucional.

Segundo Schulhoff et al. [2025], esse componente é frequentemente discutido como uma forma

de melhorar o texto produzido pelo LLM.

Informações adicionais. Algumas vezes é necessário prover informações adicionais ao LLM para a execução de uma tarefa. No *prompt* a seguir, a primeira linha contém a diretiva e as demais, as informações adicionais necessárias para contextualizar a tarefa.

> Crie um roteiro de viagem de 7 dias.
> Destino: Itália (Roma, Florença e Veneza)
> Estilo de viagem: cultural e gastronômica
> Perfil do viajante: casal na faixa dos 30 anos, primeira vez no país.

Conforme mencionado anteriormente, há uma grande variedade de técnicas de engenharia de *prompt*, impulsionada pelo crescente número de estudos sobre o tema. Muitas dessas técnicas são, na verdade, variações de algum tipo de *prompt*. Nesse contexto, é possível agrupá-las conforme os tipos ilustrados na Figura II.2.

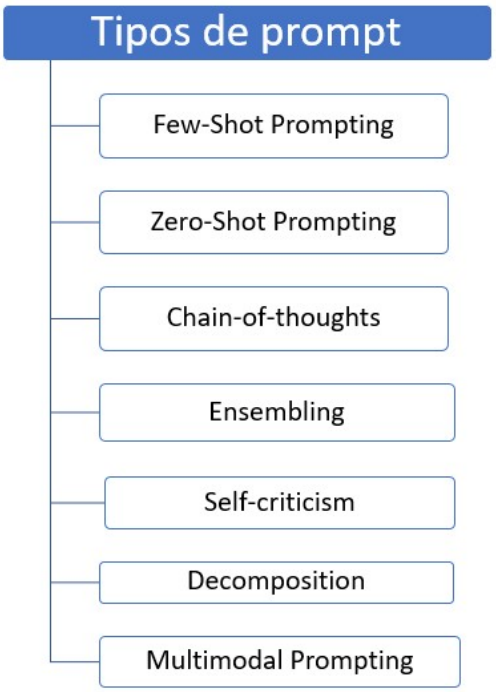


Figura II.2: Principais técnicas de engenharia de prompt. Fonte: O autor (2026)

Few-Shot Prompting é a técnica que consiste em apresentar alguns exemplos no *prompt* para explicar ao modelo o que se deseja. Esses exemplos podem ser usados tanto para ensinar ao LLM o padrão de resposta desejado quanto para auxiliá-lo a interpretar as questões [Berryman and Ziegler, 2024].

Essa abordagem surgiu a partir do trabalho de Brown et al. [2020], o qual mostrou que, com apenas alguns exemplos, é possível melhorar muito a resposta produzida pelos LLM.

Zero-Shot Prompting. Diferentemente do anterior, nesse tipo de *prompt* não são apresentados

exemplos. Essa estratégia pode ser usada quando o usuário deseja uma resposta rápida, sem a necessidade de passar informações adicionais, ou quando o assunto é tão amplo que prover exemplos poderia direcionar a resposta do modelo [Pawar et al., 2024].

Chain-of-Thought é uma abordagem baseada na ideia de dividir o problema em uma sequência de problemas menores a serem resolvidos pelo modelo antes de produzir a resposta final. Wei et al. [2022] demonstraram que essa técnica é capaz de produzir melhoria significativa no desempenho de tarefas que demandam um nível mais aprofundado de raciocínio, como problemas matemáticos, problemas que necessitam de raciocínio simbólico e até mesmo questões mais complexas que necessitam da análise de conhecimentos do senso comum.

Ensembling é um processo em que primeiramente são utilizados vários prompts para solução de um mesmo problema e, após essa etapa, a resposta final é produzida com base na agregação das respostas anteriores. Em muitos casos, é escolhida a resposta mais frequente [Schulhoff et al., 2025].

Self-criticism. Essa técnica é divida em três etapas, na primeira é solicitado ao LLM que produza a resposta para o assunto desejado. A segunda consiste em solicitar ao próprio LLM que avalie a resposta produzida anteriormente e na última etapa essa avaliação é utilizada pelo LLM para a geração da resposta final [Schulhoff et al., 2025].

Decomposition é uma categoria de técnicas que tem como principal característica dividir um problema complexo em problemas menores e mais simples de serem resolvidos. Nesse ponto, elas são similares às do tipo *chain-of-thoughts* [Schulhoff et al., 2025]. A diferença é que aqui as subtarefas são apresentadas de forma mais explícita e estruturada. Dois exemplos de trabalhos nessa linha são Khot et al. [2023] e Patel et al. [2022].

Multimodal Prompting são prompts que podem conter outros elementos além de texto, como imagens ou áudio, por exemplo.

II.3 Fine tuning

O *fine-tuning* é uma abordagem que contempla um treinamento adicional de um LLM, com um conjunto de dados customizado, para adaptá-lo a tarefas ou domínios específicos [Anisuzzaman et al., 2025]. Essa técnica tem se mostrado mais eficaz em tarefas que exigem alta precisão, especialmente em contextos onde os dados apresentam padrões difíceis de serem capturados apenas por meio de técnicas de engenharia de *prompts*.

A principal finalidade do *fine-tuning* é utilizar o conhecimento amplo obtido durante o pré-treinamento e ajustá-lo para que o modelo compreenda melhor os detalhes e exigências de uma tarefa ou área específica. Esse processo permite que os LLMs se especializem em contextos particulares, melhorando seu desempenho em objetivos definidos, sem comprometer o conhecimento geral que já foi assimilado na etapa de pré-treinamento [Wu et al., 2025]. A figura II.3, que foi extraída de

Wu et al. [2025], apresenta as diversas técnicas de *fine-tuning* existentes e sua evolução ao longo do tempo.

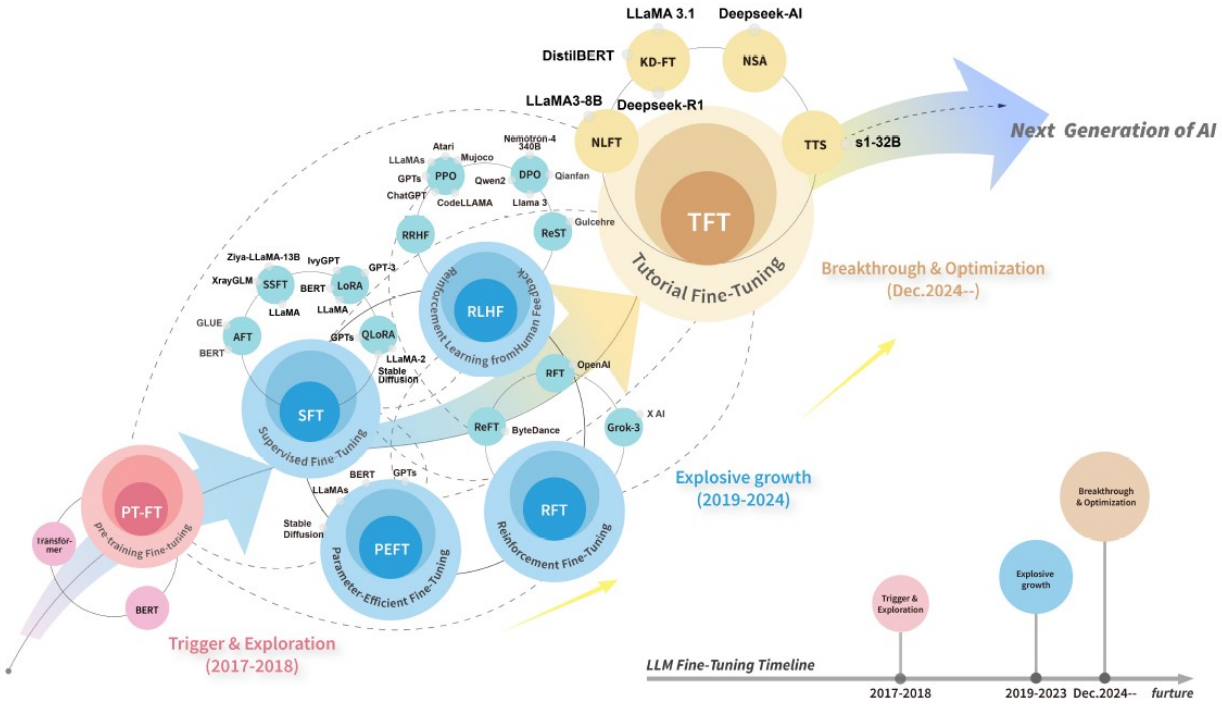


Figura II.3: Evolução das técnicas de fine-tuning. Fonte: Wu et al. [2025]

Como pode ser visto na figura, o *fine-tuning* de LLM é um tema recente e em plena expansão, o qual em poucos anos experimentou uma grande evolução na quantidade de técnicas existentes. Entre os principais conceitos que são abordados em vários trabalhos da área tem-se o *Supervised Fine-Tuning* (SFT), o *Parameter-Efficient Fine-Tuning (PEFT)* e o *Reinforcement Learning from Human Feedback (RLHF)*.

Supervised Fine-Tuning envolve a adaptação de um LLM para tarefas específicas por meio de um novo treinamento com dados rotulados. Ele pode ser usado tanto para especializar o LLM em uma única tarefa quanto em múltiplas tarefas [Wu et al., 2025]. *Parameter-Efficient Fine-Tuning (PEFT)*, que será explicado em mais detalhes na Seção II.3.1, é um tipo de técnica em que apenas uma pequena parte dos parâmetros do LLM são ajustados durante o *fine-tuning*. Essa abordagem demanda uma capacidade computacional muito menor do que a necessária para um *fine-tuning* completo [Anisuzzaman et al., 2025]. *Reinforcement Learning from Human Feedback (RLHF)* é um método em que o *fine-tuning* é baseado no julgamento humano para direcionar o comportamento desejado [Anisuzzaman et al., 2025].

Um dos riscos do *fine-tuning* é a ocorrência de *overfitting*, por este motivo, o ajuste dos hiperparâmetros deve ser realizado com cautela para que isso não aconteça [Anisuzzaman et al., 2025]. Os principais hiperparâmetros que precisam ser ajustados nesse processo são: taxa de aprendizado,

tamanho do lote e *epochs*.

A taxa de aprendizado define a velocidade na qual o modelo se adapta ao problema. Geralmente, os modelos utilizam algoritmos do tipo *gradient descent* para minimizar a função de perda. A taxa de aprendizado controla quanto os pesos do modelo são atualizados a cada iteração da otimização. Taxas de aprendizado pequenas tendem a aumentar o tempo de treinamento por demandarem um maior tempo de convergência. Por outro lado, taxas de aprendizado muito elevadas podem fazer com que o processo de otimização ultrapasse o mínimo, impedindo a convergência adequada [Parthasarathy et al., 2024].

Tamanho do lote se refere a um subconjunto dos dados de treinamento que são usados para atualizar os pesos do modelo. Durante o processo de treinamento, normalmente não é processado todo o conjunto de dados de uma vez, pois pode ser computacionalmente custoso, ao invés disso, os dados são divididos em subconjuntos menores. Esse hiperparâmetro define o tamanho do subconjunto, ou seja, define quantos registros são processados antes dos pesos do modelo serem alterados [Parthasarathy et al., 2024].

Epoch, ou 'ciclo de treinamento' em português, determina a quantidade de vezes que o processo de treinamento passa por todo o conjunto de dados [Yang and Shami, 2020]. Um valor demasiadamente elevado nesse hiperparâmetro pode resultar em *overfitting*. Em contrapartida, valores muito pequenos podem fazer o modelo não aprender o suficiente no treinamento.

II.3.1 PEFT

Embora o *fine-tuning* possa trazer benefícios significativos em diversas tarefas, a realização do *fine-tuning* completo nos LLMs atuais, que possuem bilhões de parâmetros, demanda um longo tempo e uma elevada capacidade computacional, o que, em muitos casos, inviabiliza o processo. Para superar essa limitação, foram desenvolvidas técnicas que buscam realizar o ajuste do modelo de forma mais eficiente. Esse conjunto de técnicas é conhecido como Parameter-Efficient Fine-Tuning - PEFT.

Um dos primeiros trabalhos desse tipo foi Houlsby et al. [2019] que mostrou que foi capaz de obter um desempenho com diferença de apenas 0,4% em relação ao *fine-tuning* completo, adicionando apenas 3,6% de parâmetros por tarefa. Ou seja, ajustando uma pequena parte do modelo, foi possível alcançar resultados similares a um *fine-tuning* completo. A técnica proposta neste artigo é baseada em *Adapters* que são pequenos blocos inseridos entre camadas do modelo original. A Figura II.4 ilustra essa ideia.

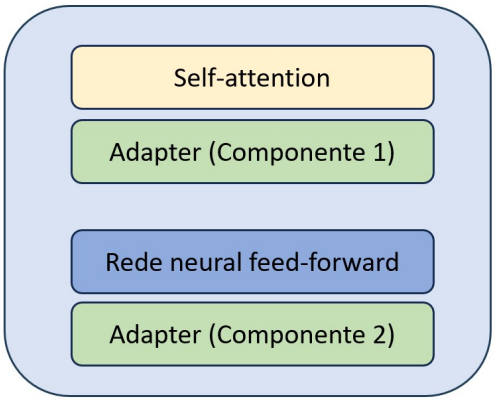


Figura II.4: Arquitetura dos adapters. Figura adaptada de Alammar and Grootendorst [2024]

No *fine-tuning*, apenas os *adapters* são treinados, o modelo original permanece inalterado. Como os *adapters* são blocos relativamente pequenos, isso faz com que a quantidade de pesos atualizados durante o ajuste fino do modelo seja significativamente menor.

Após esse artigo, diversas técnicas do tipo PEFT foram desenvolvidas, alguns exemplos são: LoRA (*Low-Rank Adaptation*) [Hu et al., 2022], *Prompt Tuning* [Lester et al., 2021], *Prefix Tuning* [Li and Liang, 2021], BitFit [Ben-Zaken et al., 2022] e $(IA)^3$ (*Infused Adapter by Inhibiting and Amplifying Inner Activations*) [Liu et al., 2022]. Neste trabalho, foi utilizado o QLoRA, que será descrito a seguir.

II.3.2 QLoRA

O método QLoRA (*Quantized LoRA*) foi apresentado inicialmente no artigo *QLoRA: Efficient Finetuning of Quantized LLMs* [Dettmers et al., 2023]. Antes de explicá-lo, é necessário apresentar o funcionamento do LoRA.

LoRA (*Low-Rank Adaptation*) é uma técnica que atualiza apenas uma parte dos parâmetros do modelo. Isso é possível, pois em vez de atualizar as grandes matrizes de pesos do modelo original, o LoRA identifica uma versão mais compacta da matriz de pesos do LLM e a introduz no modelo. No *fine-tuning*, apenas essas matrizes compactas são ajustadas [Anisuzzaman et al., 2025].

A Figura II.5 apresenta a ideia da substituição de matrizes realizada pelo LoRA.

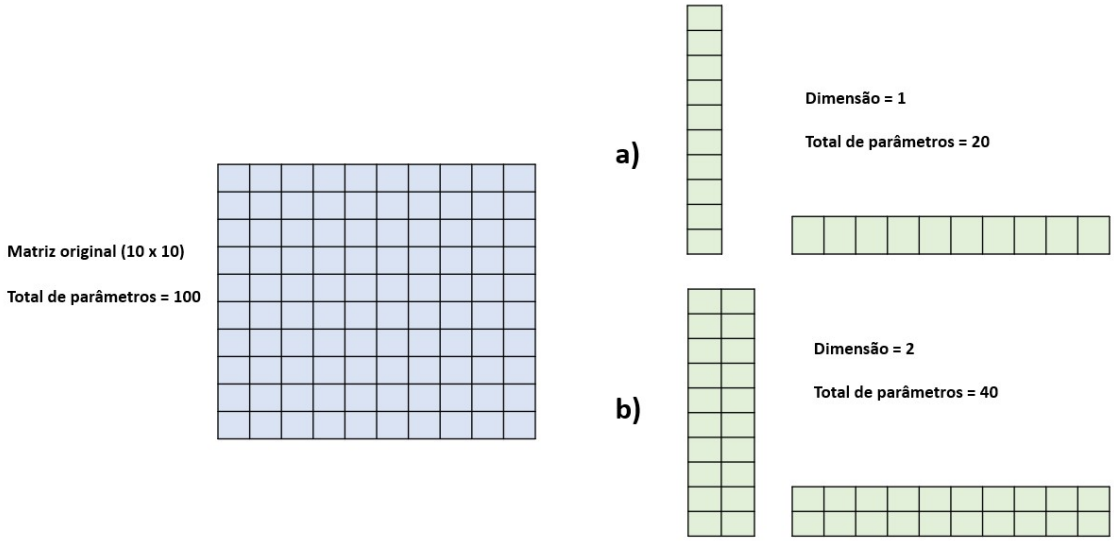


Figura II.5: Matrizes LoRA. Figura adaptada de Alammar and Grootendorst [2024]

A matriz original pode ser substituída por um par de matrizes bem menores. O tamanho dessas matrizes varia de acordo com a dimensão escolhida. É possível notar que tanto no exemplo a) quanto no b) a quantidade de parâmetros a ser ajustada reduz significativamente em relação à matriz original.

Este é apenas um exemplo para ilustrar o funcionamento do método. Na prática, os LLMs contêm diversas matrizes que podem chegar a 150 milhões de parâmetros cada [Alammar and Grootendorst, 2024].

A abordagem adotada pelo LoRA só é possível, pois os modelos de linguagem possuem uma dimensão intrínseca muito baixa [Aghajanyan et al., 2021]. Isso permite encontrar matrizes de baixa dimensão capazes de aproximar as grandes matrizes de peso dos LLMs [Alammar and Grootendorst, 2024].

Agora que foram apresentados os conceitos do LoRA, fica mais fácil compreender o QLoRA (*Quantized LoRA*). É possível tornar o LoRA ainda mais eficiente ao reduzir a quantidade de memória ocupada pelos pesos originais do modelo antes de projetá-los em matrizes menores. Os pesos de um LLM são valores numéricos com determinada precisão, a qual pode ser expressa pelo número de bits, como float64 ou float32. Ao reduzir a quantidade de bits para representar um valor, obtém-se um resultado menos preciso. Contudo, essa redução no número de bits também diminui as exigências de memória do modelo [Alammar and Grootendorst, 2024].

Em Dettmers et al. [2023], os autores do QLoRA criaram uma forma de reduzir a quantidade de bits necessária para representar os pesos sem desviar muito do modelo original. A Figura II.6 ilustra a diferença entre o funcionamento do QLoRA e uma simples compressão na quantidade de bits utilizada para representar os pesos.

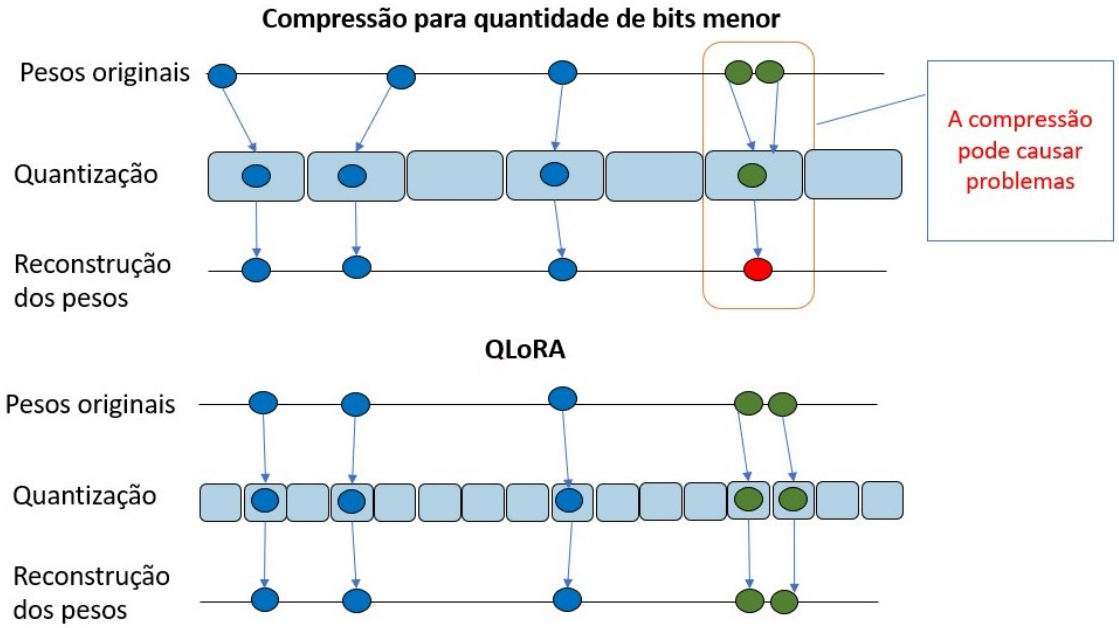


Figura II.6: QLoRA. Figura adaptada de Alammar and Grootendorst [2024]

Note que a redução na quantidade de bits pode fazer com que valores diferentes sejam representados pelo mesmo número, o que diminui a precisão no momento da reconstrução dos pesos. O QLoRA é capaz de minimizar esse problema, o que permite maiores taxas de compressão e, por consequência, maior economia de recursos computacionais. Com o QLoRA, é possível carregar o modelo original com quantização de 4 bits, o que reduz drasticamente a quantidade de memória necessária.

II.4 Métricas

Conforme será visto no capítulo IV, a métrica utilizada nesse trabalho foi a F1 Score, porém, para explicá-la, é necessário apresentar antes os conceitos de *Precision* e *Recall*.

Precision mede a proporção de instâncias corretamente classificadas como positivas entre todas as que o modelo previu como positivas [Han et al., 2011]. Em outras palavras, a métrica responde a seguinte pergunta: Dentre todas as vezes que o modelo disse que era positivo, qual o percentual de acerto? A métrica é calculada conforme a Equação II.1, onde TP (*True Positive*) se refere à quantidade de casos que o modelo classificou corretamente como positivo e FP (*False Positive*) à quantidade de casos em que o modelo errou ao classificar como positivo.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{II.1}$$

Recall mede a capacidade do modelo de encontrar todas as instâncias positivas existentes. Ou seja, do total de registros positivos existentes, quantos por cento o modelo conseguiu identificar corretamente? A métrica é calculada pela Equação II.2, na qual TP (*True Positive*) se refere à

quantidade de casos que o modelo classificou corretamente como positivo e FN (*False Negative*) à quantidade de casos em que o modelo errou ao classificar como negativo [Han et al., 2011].

$$\text{Recall} = \frac{TP}{TP + FN} \tag{II.2}$$

F1 Score é calculada como a média harmônica entre Precision e Recall [Han et al., 2011]. Por atribuir peso idêntico às duas, é especialmente útil em situações em que ambas são igualmente importantes. A Equação II.3 indica como o F1 Score é calculado.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{II.3}$$

II.5 Testes Estatísticos

Para complementar a análise das métricas de desempenho obtidas nos experimentos e verificar se as diferenças observadas entre os resultados dos modelos são estatisticamente significativas, foram empregados testes estatísticos inferenciais. A utilização desses testes permite avaliar se as variações nos valores de F1 Score decorrem de diferenças reais de desempenho ou apenas de flutuações amostrais associadas às múltiplas execuções realizadas com diferentes sementes e divisões de dados. Assim, adotaram-se procedimentos paramétricos e não paramétricos, selecionados de acordo com a verificação prévia de seus pressupostos, com destaque para o Teste t de Student e o Teste de Wilcoxon, amplamente utilizados na literatura para comparação de desempenho de modelos de aprendizado de máquina, os quais são descritos a seguir.

II.5.1 Teste t de Student

O Teste t de Student constitui um dos procedimentos estatísticos paramétricos mais tradicionais e consolidados na literatura, sendo amplamente empregado para verificar se diferenças observadas entre médias são estatisticamente significativas. Seu desenvolvimento remonta ao trabalho seminal de William Sealy Gosset, publicado sob o pseudônimo Student em 1908, quando o autor buscava avaliar a qualidade de pequenos lotes de produção na cervejaria Guinness [Bruce and Bruce, 2017]. A partir dessa formulação inicial, o teste ganhou relevância nas ciências experimentais, sociais e aplicadas, consolidando-se como ferramenta fundamental na análise inferencial.

A lógica subjacente ao Teste t fundamenta-se na distribuição t de Student, uma distribuição contínua que se aproxima da distribuição normal padrão à medida que o tamanho da amostra aumenta, mas que incorpora maior variabilidade para amostras pequenas, situação em que a incerteza sobre a variância populacional é mais pronunciada [Bruce and Bruce, 2017] [Caldwell, 2010]. A estatística t é operacionalmente expressa como a razão entre a diferença observada (entre uma média amostral

e um valor teórico, ou entre médias de dois grupos) e o erro padrão dessa diferença. Quanto maior o valor absoluto da estatística t, menor a probabilidade de que a diferença encontrada decorra do acaso, sob a hipótese nula [Bruce and Bruce, 2017][Moore et al., 2015].

A aplicação do teste pressupõe algumas condições: (i) a variável de interesse deve ser contínua e aproximadamente distribuída de forma normal; (ii) as observações devem ser independentes; e (iii) no caso de comparações entre grupos distintos, as variâncias devem ser homogêneas (pressuposto de homocedasticidade) [Caldwell, 2010]. Embora a robustez do Teste t permita certa tolerância à violação moderada desses pressupostos, a verificação prévia dessas condições — por meio de estatísticas descritivas, testes de normalidade ou gráficos exploratórios — é recomendada para assegurar validade inferencial [Moore et al., 2015].

Existem três variantes principais do Teste t, cada uma adequada a uma configuração específica de problema. O Teste t de uma amostra é utilizado quando se deseja comparar a média de uma amostra com um valor de referência conhecido, como padrões normativos, *benchmarks* ou valores esperados teoricamente. O Teste t para amostras independentes, por sua vez, é empregado quando o objetivo é comparar médias de dois grupos independentes entre si, enquanto o Teste t pareado é indicado para situações em que as observações são dependentes. Por exemplo, medidas repetidas no mesmo indivíduo, ou avaliações pré e pós-intervenção [Cohen, 1988].

No campo da ciência de dados e da avaliação de modelos de aprendizado de máquina, o Teste t de Student desempenha papel relevante ao permitir investigar se diferenças observadas em métricas de desempenho, como F1 Score, acurácia ou precisão, refletem variações sistemáticas ou apenas flutuações aleatórias decorrentes da variabilidade amostral. Ao comparar desempenhos obtidos em replicações experimentais, validações cruzadas ou em relação a valores reportados na literatura, o Teste t fornece uma base estatística sólida para sustentar conclusões sobre a superioridade, equivalência ou inferioridade entre modelos. Assim, sua utilização contribui para garantir rigor metodológico na análise de resultados, evitando inferências precipitadas a partir de diferenças que, embora aparentes, podem não ser estatisticamente significativas.

II.5.2 Teste de Wilcoxon

O Teste de Wilcoxon é um procedimento estatístico não paramétrico utilizado como alternativa ao Teste t de Student quando o pressuposto de normalidade não é atendido. Proposto originalmente por Wilcoxon [1945], o teste baseia-se nas ordenações (*ranks*) das diferenças observadas entre pares de valores, em vez de trabalhar diretamente com as médias e variâncias. Essa característica o torna especialmente adequado para amostras pequenas ou situações em que a escala de medida não é intervalar [Anderson et al., 2011].

Na sua forma mais comum — o Teste de Postos Sinalizados de Wilcoxon (*Wilcoxon Signed-Rank*

Test) — o método é aplicado para comparar duas amostras pareadas ou uma amostra com um valor de referência, avaliando se a mediana das diferenças entre as observações difere significativamente de zero. Assim como o Teste t pareado, o Teste de Wilcoxon busca identificar se a diferença observada entre dois conjuntos de medidas é estatisticamente relevante, porém sem exigir que os dados sigam uma distribuição normal [Anderson et al., 2011]. Em estudos que envolvem métricas de desempenho de modelos, como o F1 Score, o uso desse teste assegura robustez inferencial, permitindo a verificação da significância das diferenças mesmo em cenários com distribuições não gaussianas.

Capítulo III Trabalhos relacionados

A identificação de conluio em licitações não é um tema novo e já foi pesquisado sob diversos aspectos, no entanto, até o melhor do conhecimento dos autores e pesquisas na literatura, não foi identificado nenhum trabalho que tenha utilizado Modelos de Linguagem de Grande Porte para esse fim. Neste sentido, este capítulo apresenta uma revisão dos principais trabalhos relacionados à detecção de conluio em licitações públicas, abrangendo tanto abordagens tradicionais baseadas em aprendizado de máquina quanto pesquisas recentes que empregaram *Large Language Models* em problemas análogos de detecção de fraudes. Além disso, descreve-se o artigo de referência utilizado como base comparativa neste estudo, destacando suas contribuições, métodos e resultados, de modo a contextualizar o posicionamento do presente trabalho frente ao estado da arte.

As pesquisas por trabalhos relacionados foram conduzidas na base Scopus no período de novembro de 2024 a janeiro de 2025 e atualizadas em abril de 2026. A Tabela III.1 consolida as *strings* de busca usadas na pesquisa da literatura e a quantidade de trabalhos retornados em cada uma.

Tabela III.1: *Strings* de buscas usadas na pesquisa da literatura

Strings de busca	**Artigos encontrados**
"Large Language Model"AND conluio	0
llm AND conluio	0
"Large Language Model"AND collusion	8
llm AND collusion	8
collusion AND brazil	6
+ Filter: Computer Science	
Collusion AND algorithm	1251
+ Filter: Computer Science	
Collusion AND chatgpt	1
+ Filter: Computer Science	
"large language model"AND bid-rigging	0
LLM AND bid-rigging	0
"machine learning"AND bid-rigging	13
"large language model"AND fraud	254
"llm"AND fraud	147

III.1 Aprendizado de máquina para identificação de conluio

Entre as técnicas adotadas em trabalhos anteriores, é possível citar o uso de algoritmos clássicos de aprendizado supervisionado, redes neurais, aprendizado não supervisionado, grafos, identificação de padrões frequentes e criação de variáveis estatísticas. Entende-se por algoritmos clássicos de

aprendizado supervisionado aqueles que são consolidados e amplamente conhecidos e utilizados na comunidade quando se pretende treinar modelos de aprendizado de máquina, como o *Random Forest*, Regressão Logística e *Support Vector Machine*.

Em Lima [2021], por exemplo, foram utilizados os algoritmos *K-Nearest Neighbors* (KNN), *Random Forest*, *Naive Bayes*, entre outros, para criar modelos de identificação de licitações com risco de conluio a partir de dados textuais extraídos do Diário Oficial da União.

Souza [2023] também utilizou esse tipo de algoritmo para treinar modelos de identificação de conluio em licitações. Porém, nesse caso, foram utilizados dados do Portal de Compras do Governo Federal e do CADE para criar um conjunto com licitações anotadas como com risco de conluio. Os melhores resultados foram obtidos com algoritmos do tipo *ensemble methods*, *Extra Trees*, *Random Forest* e *AdaBoost*.

O uso de algoritmos clássicos combinados com a criação de variáveis estatísticas (*screens*) é a abordagem adotada na maior parte dos trabalhos produzidos nos últimos anos. Nessa linha, podemos citar Huber and Imhof [2019], Imhof and Wallimann [2021], Huber et al. [2022], Silveira et al. [2022], Wallimann et al. [2023], Scoralick et al. [2024], Fazekas et al. [2026], os quais utilizaram variáveis estatísticas como Desvio Padrão, Coeficiente de Variação, *Spread*, Assimetria, *Kurtosis*, entre outras, para ampliar a capacidade dos algoritmos de identificar padrões de conluio.

Alguns trabalhos que utilizaram algoritmos clássicos também fizeram uso de Redes Neurais. É o caso de Lima [2021], que utilizou modelos de Redes Neurais Profundas (DNN) e Redes Neurais Bidirecionais em sua tarefa de classificação, assim como Souza [2023] que usou *Multi-Layer Perceptron*. Ainda no contexto de redes neurais, Xu et al. [2024] propuseram um modelo que combina *deep learning* com *Case-Based Reasoning* (CBR), buscando aliar desempenho preditivo e interpretabilidade. A arquitetura utiliza um autoencoder para aprender relações não lineares entre os atributos e uma camada de protótipos para explicar as decisões por similaridade com casos anteriores. Avaliado em uma base de licitações públicas do Japão, o método superou modelos tradicionais de aprendizado de máquina.

Outra técnica adotada em trabalhos anteriores foi o aprendizado não supervisionado. Nessa linha, é possível citar Velasco et al. [2021], que utilizaram o algoritmo DBSCAN para identificar licitantes com endereço em comum, e Silveira et al. [2023] que usaram o algoritmo *Gaussian Mixture Clustering Model (GMCM)* como parte de uma abordagem que pretendia identificar padrões de conluio sem usar dados anotados.

Abordagens baseadas em grafos foram utilizadas por Velasco et al. [2021], citado no parágrafo anterior; Carvalho and Filho [2024] que utilizaram um banco de dados orientado a grafos para identificação de relações entre participantes de licitações do estado do Ceará e Pompeu and Filho [2025], que usaram redes neurais baseadas em grafos para identificação de indícios de conluio em

dados também do estado do Ceará.

Embora menos comum, foi encontrado um trabalho que buscou resolver o problema por meio da identificação de padrões frequentes, é o caso de Ramalho et al. [2020] que usaram o algoritmo a Priori para a identificação de situações suspeitas de conluio em contratações do estado da Paraíba.

Um desafio recorrente na literatura é a necessidade de bases detalhadas contendo todos os lances submetidos e informações sobre os participantes. Buscando contornar essa limitação, Tas [2024] desenvolveu um método capaz de identificar indícios de conluio utilizando apenas dados de resultado das licitações, como preço contratado e número de participantes. Para isso, o autor empregou o algoritmo *Kernel Regularized Least Squares* (KRLS), usado para estimar, em nível de contrato, o efeito da concorrência sobre os preços e, a partir disso, construir uma medida contínua de risco de conluio.

III.2 Large Language Model para temas semelhantes

A aplicação de LLMs para solução de problemas nas mais diversas áreas é um tema que tem sido cada vez mais pesquisado. Sendo assim, embora não tenham sido encontrados trabalhos que tenham avaliado o uso de LLMs para identificação de conluio, alguns trabalhos abordaram o uso dessas ferramentas para temas relacionados à identificação de outros tipos de fraude.

Por exemplo, Korkanti [2024] e Boskou et al. [2024] avaliaram o uso de LLM para a identificação de fraudes financeiras. Enquanto o primeiro adotou uma estratégia que incluía um treinamento adicional do LLM para especializá-lo na identificação das fraudes (*fine-tuning*), mais especificamente em transações de cartão de crédito, o segundo utilizou técnicas de engenharia de prompt para avaliar a capacidade dos LLMs na identificação de informações fraudulentas em relatórios financeiros.

Sayed et al. [2024], por sua vez, exploraram o uso de LLMs para identificação de fraudes em *E-commerce* utilizando o GPT-4 para identificação em tempo real de transações suspeitas.

Por fim, o envio de *e-mails* maliciosos é uma técnica muito utilizada por criminosos para o cometimento de diversos tipos de fraudes. O uso de LLMs para a identificação deste tipo de *e-mail* foi estudado por Hewett and Leeke [2022], enquanto Chataut et al. [2024] focaram na produção de respostas automáticas para esses *e-mails*, a fim de obter informações sobre os fraudadores.

III.3 Artigo de referência

O presente trabalho contribui para o tema ao comparar o desempenho dos Modelos de Linguagem de Grande Porte com o trabalho de Rodríguez et al. [2022]. No artigo intitulado *Collusion detection in public procurement auctions with machine learning algorithms*, os autores testaram 11 algoritmos de aprendizado de máquina (SGD, Extra Trees, Random Forest, AdaBoost, Gradient Boosting, SVC, K Neighbors, MLP, Bernoulli Naive Bayes, Gaussian Naive Bayes e Gaussian Process) em

conjuntos de dados reais de licitações envolvendo conluios já comprovados em cinco países (Brasil, Itália, Japão, Suíça e EUA). Os autores realizaram experimentos com e sem a inclusão de variáveis estatísticas para avaliar o desempenho dos modelos.

Os resultados mostraram que os algoritmos do tipo ensemble: *Extra Trees, Random Forest, Gradient Boosting e Ada Boost*, apresentaram os melhores desempenhos na detecção de conluios. Notou-se, no entanto, que não houve um único algoritmo melhor em todos os casos, o que indica que a escolha do algoritmo depende da característica dos dados. Outra conclusão evidenciada pelo trabalho foi a de que a adição de variáveis estatísticas pode aumentar a eficácia dos modelos.

Este artigo, que será denominado daqui em diante como artigo de referência, foi selecionado por ter sido citado por diversos trabalhos sobre identificação de conluio, por ter utilizado algoritmos reconhecidos e bastante adotados no treinamento de modelos de aprendizado de máquina, e por ter seguido a boa prática de disponibilizar os códigos-fonte e os dados, o que permitiu a compreensão completa do que foi feito e a reexecução dos testes.

Capítulo IV Metodologia

Os LLMs avaliados no trabalho foram o ChatGPT 4o mini e o O4 mini, da OpenAI, o DeepSeek R1-Distill-Qwen-14B e o DeepSeek-V3.2-Exp, da DeepSeek AI, e o Mistral-7B-Instruct-v0.2 da Mistral AI.

O ChatGPT 4o Mini é a versão compacta do GPT-4o e foi desenhado para ser rápido e barato, ideal para tarefas cotidianas de chat, automação e agentes que precisam de alta velocidade e baixo custo[1]. Essa versão foi escolhida em detrimento do ChatGPT 4o por ter um menor custo e apresentar resultados competitivos em muitas tarefas. O modelo O4 Mini, por sua vez, é mais voltado para tarefas complexas que demandam um maior raciocínio. Ele utiliza a abordagem *chain-of-thought*, dividindo o problema em etapas menores antes de produzir a resposta final[2].

O DeepSeek R1-Distill-Qwen-14B é um modelo de linguagem de grande porte desenvolvido pela DeepSeek AI com foco em raciocínio complexo e solução de problemas multietapas. Ele é uma versão destilada do modelo DeepSeek R1, que conta com 671 bilhões de parâmetros, e foi projetado para oferecer alto desempenho com menor custo computacional. Com 14 bilhões de parâmetros, o R1-Distill-Qwen-14B combina a arquitetura Qwen da Alibaba com técnicas de destilação e raciocínio supervisionado, permitindo que o modelo reproduza o comportamento analítico do R1 em um formato mais leve [Guo et al., 2025].

O DeepSeek-V3.2-Exp é uma variante experimental da família de modelos DeepSeek-V3.2, projetada para explorar e validar otimizações de eficiência em cenários de contexto longo por meio da introdução de um mecanismo de atenção esparsa (DeepSeek Sparse Attention), que reduz os requisitos computacionais durante o treinamento e a inferência, mantendo desempenho competitivo em tarefas de linguagem natural [DeepSeek-AI, 2025]. Essa abordagem de atenção esparsa seleciona um subconjunto de tokens relevantes para o cálculo de atenção, diminuindo a complexidade de processamento em relação à atenção densa tradicional, o que torna o modelo mais eficiente para lidar com sequências extensas de texto [DeepSeek-AI, 2025]. Essa versão foi escolhida por ser a mais recente do DeepSeek no momento em que os testes foram realizados.

O Mistral-7B-Instruct-v0.2, desenvolvido pela Mistral AI, é um modelo de linguagem aberto que, com seus 7 bilhões de parâmetros, se destaca pelo equilíbrio entre desempenho e acessibilidade.

[1] https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Último acesso em 13/04/2025
[2] https://openai.com/index/introducing-o3-and-o4-mini/. Último acesso em 19/04/2025

Baseado em uma arquitetura Transformer otimizada, ele emprega técnicas como a atenção em grupo (Grouped-Query Attention) e a janela deslizante (Sliding Window Attention) para uma inferência mais rápida e eficiente em termos de memória [Jiang et al., 2023]. Reconhecido por sua robustez em tarefas gerais de linguagem, geração de código e raciocínio básico, o Mistral 7B foi incluído na avaliação como um representante de modelos compactos de alta qualidade, servindo como um ponto de referência para eficiência e desempenho em sua classe de tamanho.

A Figura IV.1 apresenta de forma macro as etapas da metodologia adotada para avaliação dos LLMs.

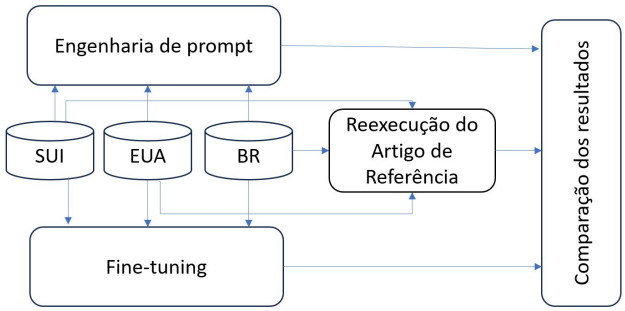


Figura IV.1: Metodologia para avaliação dos LLMs na identificação de indícios de conluio. Fonte: O autor (2026)

A seguir, são apresentados: os conjuntos de dados selecionados, os experimentos com engenharia de *prompt* e *fine-tuning*, bem como as métricas e testes estatísticos utilizados para avaliar os resultados. Mais informações sobre a reexecução dos algoritmos do artigo de referência estão disponíveis na Seção V.1.

IV.1 Conjuntos de dados

Para os testes, foram selecionados três conjuntos de dados do artigo de referência: Brasil, Estados Unidos e Suíça Ticino[3]. Todos são compostos apenas por campos numéricos relacionados aos lances ofertados nas licitações. A tabela IV.1 apresenta os campos disponíveis em cada um deles.

[3]Disponíveis em `https://ars.els-cdn.com/content/image/1-s2.0-S0926580521004982-mmc2.zip`. Último acesso: 15/2/2026

Tabela IV.1: Campos disponíveis nos conjuntos de dados do Brasil, dos Estados Unidos e da Suíça

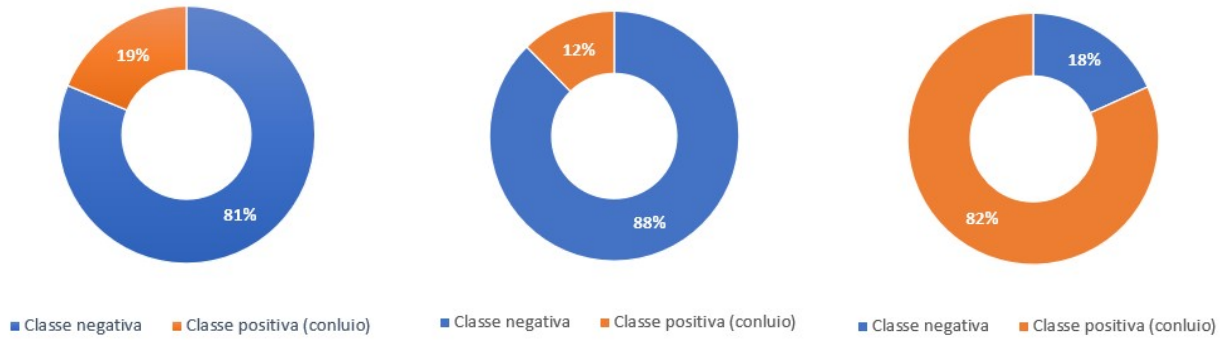
Brasil	**Estados Unidos**	**Suíça Ticino**
- Código identificador da licitação	- Código identificador da licitação	- Código identificador da licitação
- Valores dos lances submetidos	- Valores dos lances submetidos	- Valores dos lances submetidos
- Vencedor da licitação	- Vencedor da licitação	- Vencedor da licitação
- Total de lances por licitação	- Total de lances por licitação	- Total de lances por licitação
- Data da licitação	- Data da licitação	- Data da licitação
- Identificador da empresa	- Identificador da empresa	- Quantidade de participantes no consórcio
- Valor estimado antes da licitação	- Valores dos lances ajustados pela inflação	
- Local de realização da licitação	- Valores ajustados por índice de preço do leite	
- Estado da federação onde ocorreu a licitação		
- Coeficiente de Variação (CV)	- Coeficiente de Variação (CV)	- Coeficiente de Variação (CV)
- *Spread* / Amplitude Relativa dos Lances (SPD)	- *Spread* / Amplitude Relativa dos Lances (SPD)	- *Spread* / Amplitude Relativa dos Lances (SPD)
- Diferença Relativa entre os Dois Menores Lances (DIFFP)	- Diferença Relativa entre os Dois Menores Lances (DIFFP)	- Diferença Relativa entre os Dois Menores Lances (DIFFP)
- Distância Relativa (RD)	- Distância Relativa (RD)	- Distância Relativa (RD)
- Assimetria (SKEW)	- Assimetria (SKEW)	- Assimetria (SKEW)
- Estatística Kolmogorov–Smirnov (KSTEST)	- Estatística Kolmogorov–Smirnov (KSTEST)	- Estatística Kolmogorov–Smirnov (KSTEST)

Esses conjuntos de dados foram escolhidos por apresentarem o melhor (Brasil) e o pior (Estados Unidos) resultado no artigo de referência. Como terceira opção, foi selecionado um dos conjuntos de dados da Suíça presentes no artigo, o chamado Suíça Ticino. Este conjunto foi selecionado devido a outro aspecto importante que foi considerado no momento da escolha dos conjuntos de dados, o custo. Os conjuntos da Itália, do Japão e da Suíça SG&GR eram maiores do que o dos Estados Unidos e tornariam o custo dos experimentos ainda mais elevado. Por esse grupo de fatores, optou-se por utilizar apenas os três. A Tabela IV.2 consolida algumas informações relevantes para compreensão dos dados.

Tabela IV.2: Características dos conjuntos de dados

	Brasil	Estados Unidos	Suíça Ticino
Total de registros	683	7.004	1.629
Registros da classe positiva (conluio)	128	866	1.332
Quantidade de licitações	101	3.754	224
Campos nulos	Não	Não	Não

Um aspecto que merece ser mencionado é a diferença de tamanho dos conjuntos de dados: o dos Estados Unidos é 10 vezes maior do que o do Brasil. O aparente desbalanceamento dos dados, que pode ser visto ao comparar a quantidade de registros da classe positiva com o total de registros, é outro ponto que chama a atenção. A Figura IV.2 permite visualizar melhor esse ponto.



(a) Brasil (b) Estados Unidos (c) Suíça Ticino

Figura IV.2: Comparação entre a quantidade de registros da classe positiva e negativa. Fonte: O autor (2026)

Percebe-se que os três conjuntos são fortemente desbalanceados, sendo que no Brasil e nos Estados Unidos a classe preponderante é a negativa, enquanto no Suíça Ticino é a positiva.

A quantidade de lances por pregão é outra característica que varia bastante entre os conjuntos. A Figura IV.3 mostra que, enquanto no Brasil a distribuição é mais igualitária entre as diferentes quantidades, no Suíça Ticino a maioria das licitações contém entre 5 e 10 lances e nos Estados Unidos praticamente todas as licitações possuem entre 1 e 4 lances.

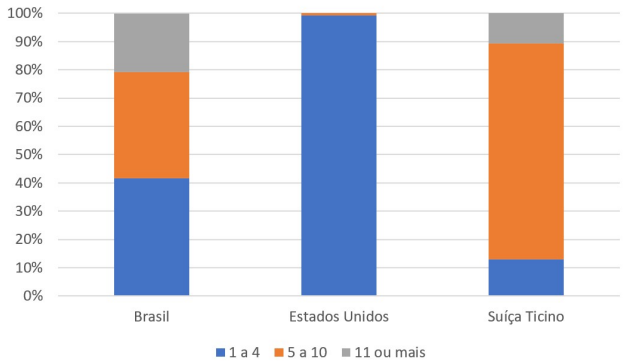
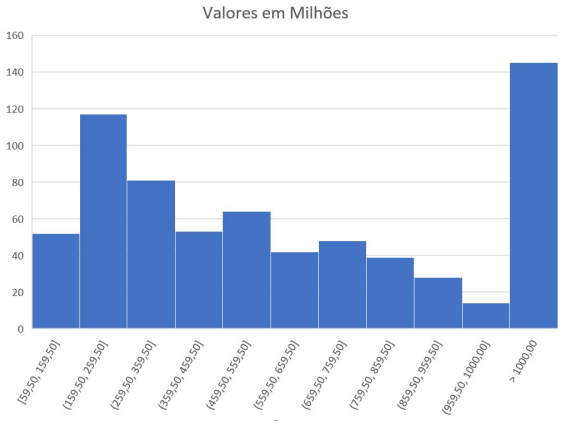
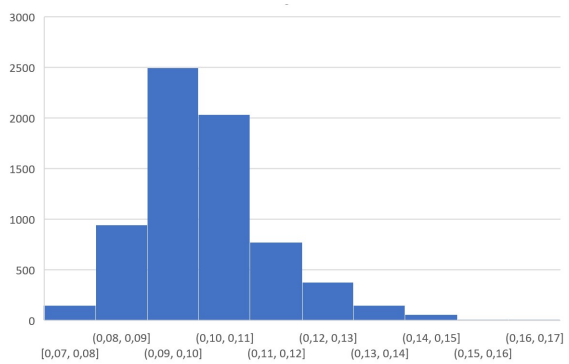


Figura IV.3: Distribuição do número de lances por pregão em três contextos: Brasil, Estados Unidos e Suíça Ticino. Fonte: O autor (2026)

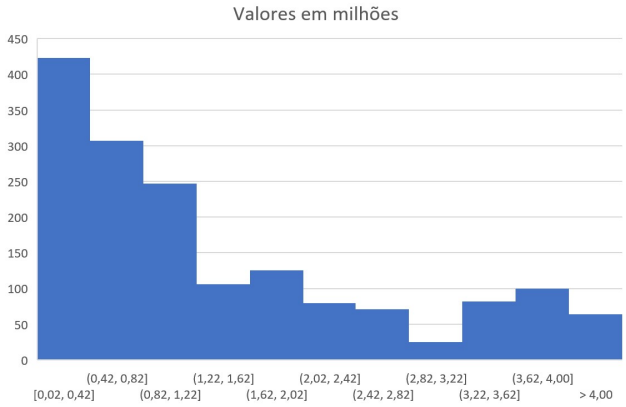
O valor dos lances é outro campo relevante que possui características distintas entre os conjuntos de dados. Na Figura IV.4 é possível perceber que, enquanto nos Estados Unidos o valor varia entre 7 e 17 centavos, no conjunto de dados da Suíça os valores começam em 200 mil e chegam a mais de 4 milhões. No Brasil, os valores são ainda maiores, começando com 59 milhões e chegando a mais de 1 bilhão, sendo que a maior parte dos valores se encontra nesta última faixa.



(a) Brasil



(b) Estados Unidos



(c) Suíça

Figura IV.4: Distribuição do valor dos lances. Fonte: O autor (2026)

Com base nas informações apresentadas, é possível concluir que os conjuntos de dados têm características bem diferentes, o que é desejado para trazer maior robustez aos resultados dos experimentos.

Continuando com a análise exploratória dos dados, a fim de apresentar mais informações sobre as características de cada um dos conjuntos de dados, apresenta-se, a seguir, a análise de correlação entre os atributos para cada um dos conjuntos. Começando pelo da Suíça, Figura IV.5.

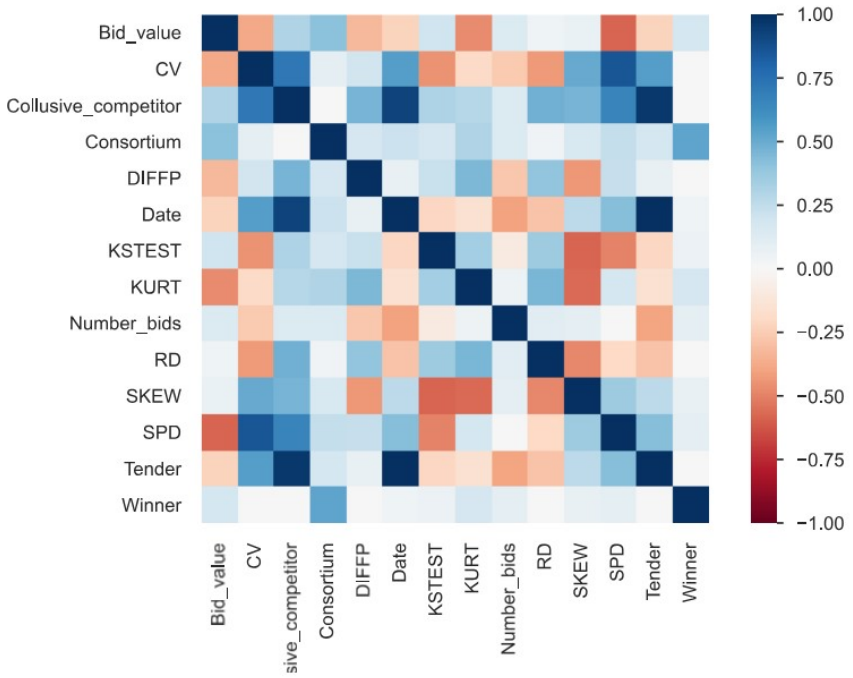


Figura IV.5: Correlação entre os atributos do conjunto de dados da Suíça. Fonte: O autor (2026)

A matriz de correlação mostra que a variável *Winner* apresenta baixa correlação com os demais atributos, indicando que a vitória na licitação não parece depender de uma relação linear simples com as variáveis analisadas. Já a variável *Collusive_competitor* apresenta correlações mais relevantes com atributos como CV e SPD, sugerindo que medidas relacionadas à dispersão dos lances podem estar associadas à ocorrência de conluio.

As correlações envolvendo *Tender* devem ser interpretadas com cautela, pois essa variável representa apenas o código identificador da licitação. Assim, sua correlação numérica não indica necessariamente uma relação estatística substantiva, mas pode refletir a concentração de casos em determinadas licitações. Esse ponto reforça a importância de separar os dados por licitação nos conjuntos de treino e teste, evitando vazamento de informação.

Por fim, como *Collusive_competitor* e *Winner* são variáveis binárias, os coeficientes de correlação devem ser interpretados como uma indicação exploratória de associação linear, e não como evidência suficiente para explicar, isoladamente, a ocorrência de conluio.

No conjunto de dados dos Estados Unidos, Figura IV.6, a matriz de correlação mostra uma associação muito forte entre *Bid_value*, *Bid_value_inflation_raw_milk_price_adjusted_bid* e *Bid_value_without_inflation*. Esse resultado é esperado, pois essas variáveis representam diferentes versões do valor da proposta. Por esse motivo, elas carregam informações muito semelhantes. Também se observa uma correlação relevante entre CV, SPD e *Number_bids*. Essas variáveis parecem capturar características relacionadas à dispersão e à estrutura competitiva da licitação. Isso sugere que aspectos como variação dos lances e quantidade de propostas podem estar associados ao comporta-

mento observado no processo licitatório.

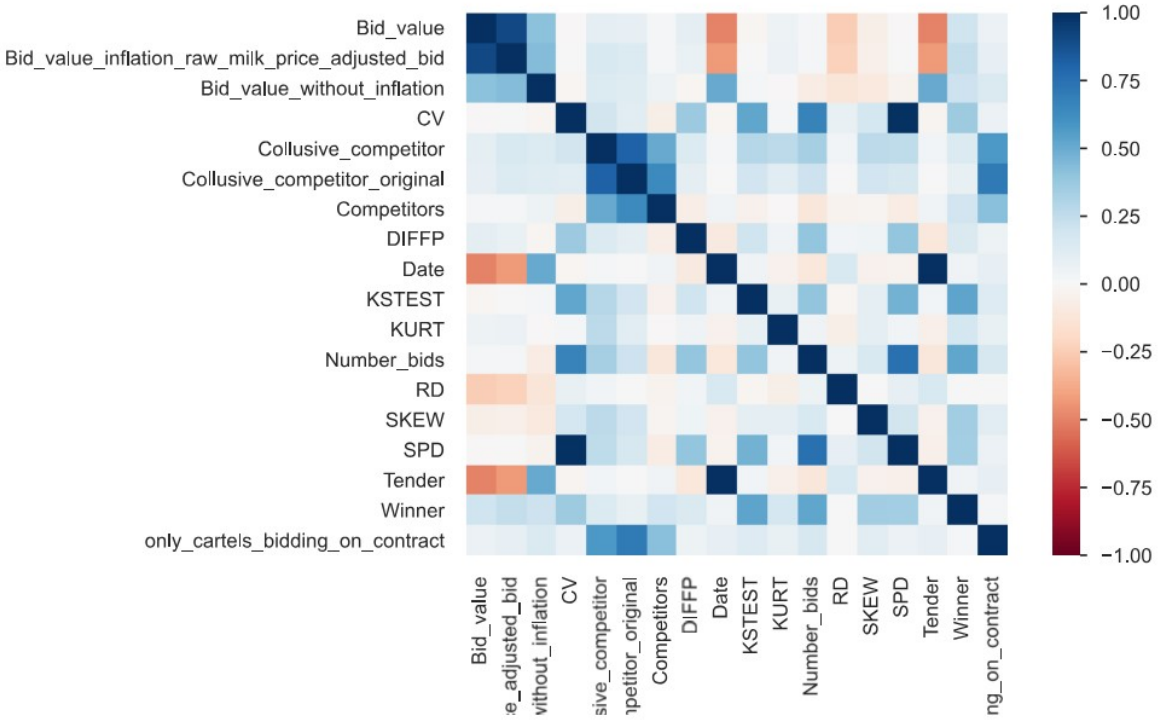


Figura IV.6: Correlação entre os atributos do conjunto de dados dos Estados Unidos. Fonte: O autor (2026)

A variável *Collusive_ competitor* apresenta correlação positiva com *Collusive_ competitor_ original* e também com *only_ cartels_ bidding_ on_ contract*, o que é coerente com o significado dessas variáveis. Esse padrão indica que há consistência entre as informações relacionadas à presença de participantes em conluio. Como essas variáveis estão diretamente ligadas ao rótulo, a única utilizada no treinamento foi a primeira, as demais foram excluídas.

Assim como no gráfico anterior, as correlações envolvendo *Tender* devem ser interpretadas com cautela, pois essa variável representa apenas o identificador da licitação. A correlação de *Tender* com variáveis de valor ou data pode refletir a forma como os dados foram organizados ou codificados, e não necessariamente uma relação estatística relevante. Por fim, a variável *Winner* continua apresentando correlações relativamente baixas com a maior parte dos atributos. Isso indica que a vitória na licitação não parece ser explicada por uma relação linear simples com uma única variável. Portanto, a predição desse resultado provavelmente depende de relações mais complexas entre os atributos.

No conjunto de dados do Brasil, Figura IV.7, a matriz de correlação indica forte associação entre *Bid*, *Bid_ value* e *Pre-Tender Estimate (PTE)*, o que é esperado, uma vez que propostas de maior valor tendem a estar associadas a licitações com maior valor estimado.

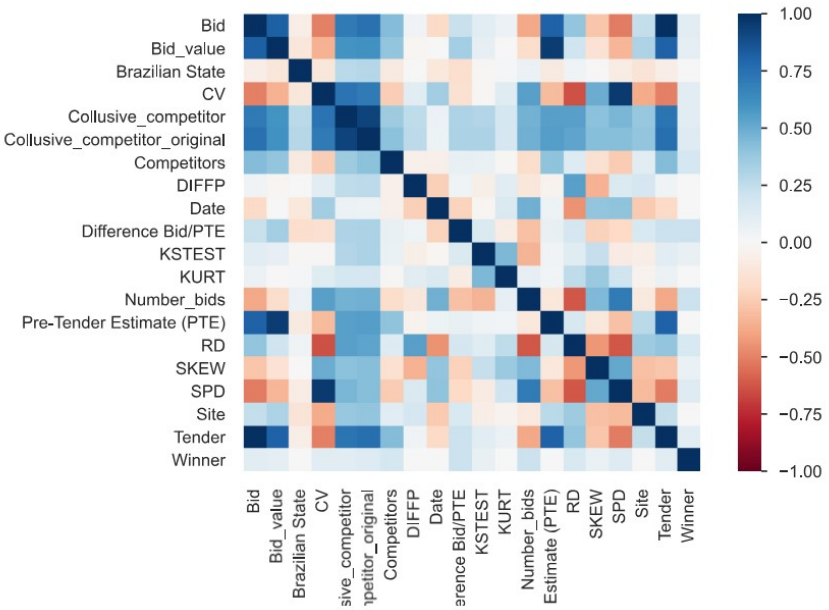


Figura IV.7: Correlação entre os atributos do conjunto de dados do Brasil. Fonte: O autor (2026)

Além disso, CV, *Number_ bids*, SPD e *Pre-Tender Estimate (PTE)* apresentam correlações relevantes com *Collusive_ competitor*, sugerindo que características da estrutura dos lances, como dispersão, quantidade de lances e valor estimado da licitação, podem estar associadas à ocorrência de conluio. Por outro lado, *Winner* apresenta baixa correlação com a maioria dos atributos, indicando que a vitória na licitação não parece ser explicada por uma relação linear simples com as variáveis analisadas. Por fim, as correlações envolvendo *Tender* devem ser interpretadas com cautela, pois essa variável corresponde apenas ao identificador da licitação.

IV.2 Engenharia de Prompt

Nos experimentos com engenharia de *prompt*, os modelos ChatGPT 4o Mini, O4 Mini, DeepSeek-V3.2-Exp e Mistral-7B-Instruct-v0.2 foram avaliados por meio da técnica de *Few-Shot Prompting*, utilizando exemplos rotulados extraídos do próprio conjunto de dados. Inicialmente, cada base foi dividida em conjuntos de treino (80%) e teste (20%), sendo que os exemplos apresentados ao modelo foram selecionados apenas a partir da partição de treino. Em cada execução, foram escolhidos registros da classe positiva (conluio) para compor a seção de exemplos do *prompt*, que também incluía a descrição dos campos do conjunto de dados, os registros a serem classificados e instruções explícitas sobre o formato de resposta esperado.

Os dados de teste não foram enviados de uma só vez. Como foi observado que *prompts* muito extensos levavam os modelos a retornar quantidades incorretas de rótulos, os registros foram divididos em múltiplos envios, com no máximo 20 instâncias por *prompt*. Para cada modelo, foram conduzidos experimentos com 10 exemplos, contudo, nos modelos da OpenAI, por terem sido ava-

liados primeiro, foram feitos experimentos também com 100 exemplos. Por padrão, os dados foram incluídos no *prompt* no formato CSV, mas foi conduzido um experimento com os dados apresentados no formato JSON para avaliar o impacto do formato dos dados no nível de acerto do modelo.

Em cada experimento, foram executadas 10 rodadas com diferentes divisões treino e teste, sendo o desempenho final definido pela média do F1 Score obtido nas execuções.

IV.3 Fine-tuning

O segundo conjunto de experimentos envolveu a aplicação de *fine-tuning* nos modelos ChatGPT 4o Mini, DeepSeek R1-Distill-Qwen-14B e Mistral-7B-Instruct-v0.2, utilizando dados rotulados para adaptar seus parâmetros internos à tarefa de detecção de conluio. Diferentemente dos experimentos com engenharia de *prompt*, no ajuste fino foram utilizados todos os dados disponíveis para treinamento, com o objetivo de maximizar a capacidade de especialização dos modelos no domínio analisado. A divisão dos dados seguiu o método *Three-Way Holdout*, com separação estratificada em treino, validação e teste.

Cada experimento foi repetido com diferentes sementes aleatórias, sendo o resultado final de cada modelo definido como a média do F1 Score obtido em 10 execuções independentes. No caso do ChatGPT 4o Mini, o *fine-tuning* foi realizado por meio da funcionalidade específica disponibilizada pela OpenAI. Já para os modelos DeepSeek R1-Distill-Qwen-14B e Mistral-7B-Instruct-v0.2, o ajuste fino foi realizado com técnicas de *Parameter-Efficient Fine-Tuning* (PEFT) e QLoRA, utilizando as bibliotecas Hugging Face em ambiente Google Colab Pro, devido às exigências computacionais do processo.

Além do experimento padrão do *fine-tuning*, foram conduzidos experimentos adicionais para investigar fatores específicos. No conjunto de dados dos Estados Unidos, diante do fato de que os LLMs ajustados apresentaram resultados muito superiores aos reportados no artigo de referência, foi incluído um teste adicional não previsto inicialmente: o treinamento de um novo modelo com o algoritmo *Gradient Boosting*, com reimplementação e ajuste de hiperparâmetros fora do código original do artigo. O objetivo foi verificar se a diferença observada decorria da superioridade dos LLMs ou de limitações na configuração do algoritmo no artigo de referência.

No conjunto de dados do Brasil, foi realizada inicialmente uma análise complementar motivada pelo fato de que, devido às diferenças na forma de divisão dos dados entre os experimentos, os algoritmos tradicionais acabaram sendo treinados com uma quantidade maior de dados do que os LLMs. Em um cenário de escassez de dados, como é o caso desta base, essa diferença poderia impactar de forma relevante o desempenho dos modelos. Em seguida, foi conduzido um experimento adicional com o modelo Mistral 7B no qual foi aplicada ponderação diferenciada das classes na função de perda durante o treinamento, atribuindo maior peso à classe minoritária, com o objetivo

de reduzir o viés do modelo em favor da classe majoritária e mitigar os efeitos do desbalanceamento.

A Figura IV.8 resume os procedimentos descritos acima para facilitar a compreensão.

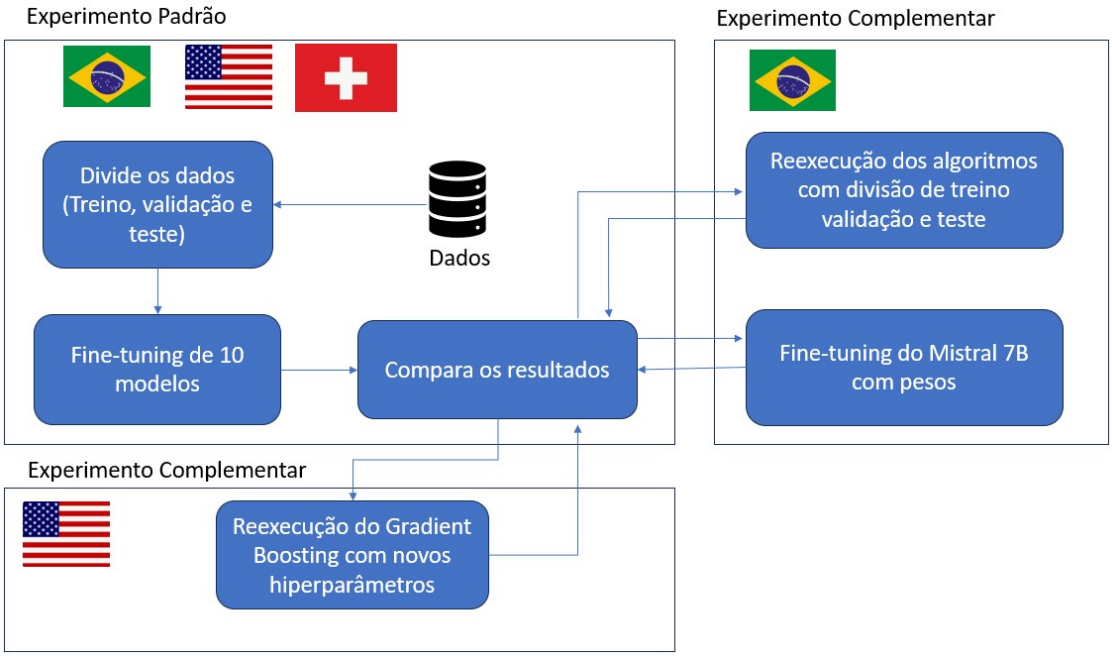


Figura IV.8: Descrição dos experimentos com fine-tuning. Fonte: O autor (2026)

IV.4 Métricas

A métrica selecionada para comparação do desempenho foi a F1 Score, que foi escolhida pois, no problema em análise, tão importante quanto o modelo identificar o máximo possível de ocorrências de conluio é o fato de ele gerar o mínimo possível de falsos positivos. Nesse contexto, onde as métricas *precision* e *recall* são igualmente importantes, a métrica F1 Score é a escolha ideal por consistir na média harmônica das duas.

No artigo de referência, embora o código-fonte disponibilizado indique que a métrica F1 Score foi calculada, os resultados não foram explicitamente apresentados. Por este motivo, foi necessário reexecutar o código do artigo de referência. A Seção V.1 apresenta o F1 Score obtido por cada um dos algoritmos durante a reexecução do código-fonte disponibilizado no artigo de referência.

IV.5 Testes estatísticos

Para verificar se as diferenças observadas nos valores de F1 Score entre os experimentos realizados e os resultados apresentados no artigo de referência são estatisticamente significativas, aplicou-se o Teste t de Student de uma amostra. Em todos os casos, antes de sua aplicação, foram verificados os pressupostos do teste, a saber: (i) independência entre as observações, (ii) normalidade da distribuição dos valores e (iii) natureza contínua da variável analisada.

Na prática, foi avaliada apenas a normalidade da distribuição dos resultados, uma vez que a forma como os experimentos foram desenhados já garante os pressupostos (i) e (iii). Isso ocorre pois os experimentos do artigo de referência e deste trabalho são independentes e a métrica F1 Score é de natureza contínua.

Para verificação do pressuposto de normalidade, aplicou-se o teste de Shapiro–Wilk [Shapiro and Wilk, 1965], que avalia a hipótese nula de que a amostra provém de uma distribuição normal. O teste calcula a estatística W, baseada na associação entre os valores observados (ordenados) e os quantis esperados sob normalidade, sendo que valores de W próximos de 1 indicam maior compatibilidade com a distribuição normal [Shapiro and Wilk, 1965]. A decisão foi tomada a partir do p-valor do teste: quando $p \geq 0{,}05$, não se rejeitou a hipótese de normalidade; quando $p < 0{,}05$, considerou-se que a suposição de normalidade não foi atendida, optando-se por uma alternativa não paramétrica.

Nos casos em que a suposição de normalidade não foi atendida, empregou-se o teste não paramétrico de Wilcoxon para uma amostra.

Capítulo V Experimentos e resultados

Conforme explicado no capítulo de metodologia, a avaliação dos LLMs foi realizada por meio de duas abordagens: engenharia de *prompt* e *fine-tuning*. Neste capítulo, serão apresentados primeiro os resultados da reexecução dos experimentos do artigo de referência, os quais foram usados como base de comparação. Em seguida, serão detalhados os experimentos feitos com engenharia de *prompt* e, por fim, com *fine-tuning*.

V.1 Reexecução dos experimentos do artigo de referência.

No artigo de referência, apesar de o código-fonte disponibilizado incluir o cálculo da métrica F1 Score, os autores não apresentaram esses resultados de forma explícita; mostraram apenas o valor da Acurácia, Falsos Positivos, Falsos Negativos e Acurácia Balanceada. Diante disso, tornou-se necessário executar novamente o código original, mais especificamente os experimentos que utilizaram todos os campos de cada conjunto de dados juntamente com a adição das variáveis estatísticas. Os resultados encontram-se na Tabela V.1.

Tabela V.1: F1 Score por algoritmo nos conjuntos Brasil, Estados Unidos e Suíça Ticino

Algoritmo	**F1 Score (Brasil)**	**F1 Score (Estados Unidos)**	**F1 Score (Suíça Ticino)**
AdaBoostClassifier	0,648	0,152	**0,939**
BernoulliNB	0,248	0,080	0,879
ExtraTreesClassifier	0,697	0,199	**0,938**
GaussianNB	0,543	0,206	0,892
GaussianProcessClassifier	0,000	0,138	0,013
GradientBoostingClassifier	**0,794**	**0,302**	0,917
KNeighborsClassifier	0,493	0,020	0,849
MLPClassifier	0,525	0,000	0,890
RandomForestClassifier	**0,729**	**0,274**	0,925
SGDClassifier	0,370	0,023	0,822
SVC	0,684	0,229	0,641

Nos conjuntos de dados do Brasil e dos Estados Unidos, o algoritmo *Gradient Boosting* obteve os melhores resultados, seguido de perto pelo *Random Forest*. Contudo, os valores ficaram bem abaixo da Acurácia Balanceada que foi de 0,906 (Brasil) e 0,616 (Estados Unidos) com o *Gradient Boosting* e, no caso do *Random Forest*, 0,860 (Brasil) e 0,616 (Estados Unidos). No conjunto de dados da Suíça, os resultados de F1 Score foram substancialmente mais elevados para a maioria dos algoritmos avaliados, com destaque, nesse caso, para o AdaBoost e o ExtraTrees. Esse resultado

geral melhor sugere que o conjunto suíço apresenta características mais favoráveis à discriminação entre as classes.

Voltando aos conjuntos de dados do Brasil e dos Estados Unidos, o fato de valores de F1 Score terem sido tão abaixo da acurácia balanceada pode ser explicado pelo forte desbalanceamento dos dados e por uma baixa precisão dos modelos. Para explicitar melhor o motivo, primeiramente é necessário apresentar como é calculada a Acurácia Balanceada. A Acurácia Balanceada (*Balanced Accuracy*) é uma métrica de avaliação amplamente utilizada em problemas de classificação com conjuntos de dados desbalanceados, nos quais as classes não estão representadas de forma uniforme.

Formalmente, a Acurácia Balanceada é definida como a média das taxas de acerto individuais de cada classe, isto é, a média dos valores de *recall* (ou sensibilidade) calculados separadamente para cada classe [Rodríguez et al., 2022]. Seja L o número de classes do problema, a métrica pode ser expressa como:

$$\text{Balanced Accuracy} = \frac{1}{L}\sum_{l=1}^{L}\text{Recall}_l \tag{V.1}$$

onde o *recall* da classe l é dado por:

$$\text{Recall}_l = \frac{TP_l}{TP_l + FN_l} \tag{V.2}$$

em que TP_l representa o número de verdadeiros positivos da classe l, e FN_l o número de falsos negativos dessa mesma classe.

No caso particular de problemas de classificação binária, a acurácia balanceada pode ser interpretada como a média aritmética entre a sensibilidade (taxa de verdadeiros positivos) e a especificidade (taxa de verdadeiros negativos). Nessa situação, a métrica pode ser expressa de forma equivalente por [Grandini et al., 2020]:

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \tag{V.3}$$

Com base nessa definição e na do F1 Score, que foi apresentada na Seção II.4, é possível buscar uma interpretação para os resultados. Como a classe negativa é muito mais frequente do que a classe positiva (conluio), os modelos tendem a aprender principalmente os padrões da classe negativa, o que pode dificultar a correta identificação dos casos positivos. Entretanto, a diferença observada entre a Acurácia Balanceada e o F1 Score não decorre diretamente da frequência das classes, mas da forma como cada métrica é definida. A Acurácia Balanceada é calculada a partir do recall de cada classe e não incorpora a métrica de precisão (precision). Assim, é possível que ela permaneça em nível moderado quando o modelo apresenta recall razoável para a classe positiva, mesmo que produza muitos falsos positivos. Nessa situação, a precisão se reduz e, consequentemente, o F1 Score

— que combina precisão e recall — assume valores mais baixos, refletindo de forma mais rigorosa a limitação do modelo na detecção efetiva de conluio.

Para tornar essa diferença mais clara, considere um exemplo numérico hipotético em um cenário desbalanceado. Suponha um conjunto de dados com 1.000 registros, dos quais 100 pertencem à classe positiva (conluio) e 900 à classe negativa (não conluio). Considere que determinado modelo classificou corretamente 70 dos 100 casos de conluio (verdadeiros positivos) e deixou de identificar 30 (falsos negativos). Ao mesmo tempo, classificou incorretamente 300 casos negativos como positivos (falsos positivos) e classificou corretamente 600 como negativos (verdadeiros negativos).

Nessa situação, o recall da classe positiva é igual a $70/(70 + 30) = 0{,}70$, enquanto o recall da classe negativa é $600/(600 + 300) = 0{,}67$. A Acurácia Balanceada, definida como a média dos recalls das duas classes, resulta, portanto, em aproximadamente 0,685. Observa-se que, segundo essa métrica, o desempenho do modelo é moderado.

Entretanto, a precisão (precision) da classe positiva é dada por $70/(70 + 300) \approx 0{,}19$, indicando que apenas uma pequena fração das instâncias classificadas como conluio corresponde, de fato, a casos positivos. Como o F1 Score é a média harmônica entre precisão e recall, seu valor também se reduz significativamente nesse cenário, situando-se em torno de 0,30. Assim, mesmo com Acurácia Balanceada moderada, o F1 Score evidencia desempenho insatisfatório na identificação confiável dos casos de conluio, devido ao grande número de falsos positivos.

Em problemas do tipo que estamos trabalhando, um modelo que identifica bem apenas a classe negativa ou que apresenta muitos falsos positivos não pode ser considerado adequado, pois sua utilidade depende justamente da capacidade de reconhecer corretamente os casos positivos. Por este motivo, considera-se mais adequado utilizar como base de comparação com o artigo de referência os valores da métrica F1 Score.

V.2 Engenharia de prompt

Os experimentos com engenharia de *prompt* envolveram a avaliação dos modelos ChatGPT 4o Mini e O4 Mini, ambos da OpenAI, do modelo DeepSeek-V3.2-Exp, da DeepSeek AI, e do Mistral-7B-Instruct-v0.2 da empresa Mistral AI.

Os testes com os modelos da OpenAI e com o DeepSeek foram implementados em Python 3.11 e utilizaram a API das empresas para consultar os modelos. Os códigos foram executados em uma máquina equipada com processador Intel Core i7-7500U, 16 GB de memória RAM e sistema operacional Windows 10. No caso do Mistral, foi utilizado o ambiente do Google Colab PRO. A seguir, são descritos os procedimentos adotados e os resultados alcançados por cada modelo.

V.2.1 ChatGPT

Few-Shot Prompting foi a abordagem escolhida para avaliar se, com a engenharia de *prompt*, os LLMs seriam capazes de produzir resultados satisfatórios na detecção de conluio. Os testes foram realizados da seguinte forma.

Inicialmente, os dados foram divididos em treino e teste, sendo 80% para treino e 20% para teste. Dos dados de treino, foram selecionados 10 registros classificados como conluio para serem apresentados ao LLM como exemplos de registros da classe positiva. Os dados de teste foram todos enviados ao modelo para classificação. Aqui cabe uma observação: foram escolhidos inicialmente 10 registros, pois, como o próprio nome indica, a técnica é baseada em poucos exemplos. A intenção foi verificar se um baixíssimo número de exemplos - quando comparado à quantidade usualmente utilizada para o treinamento de modelos tradicionais de aprendizado de máquina - seria o suficiente.

Nos primeiros experimentos, que foram realizados com o conjunto de dados do Brasil, identificou-se que os modelos não retornavam a quantidade correta de elementos na resposta. Por exemplo, eram enviados 137 registros para classificação, mas a resposta continha 138. Foi possível notar que, ao reduzir a quantidade de registros enviados para classificação em cada prompt, os modelos passaram a responder com a quantidade correta. Optou-se, então, por dividir os dados de teste em vários prompts com no máximo 20 registros para classificação.

A figura V.1 apresenta um exemplo de *prompt* utilizado nos experimentos.

```
Considere um dataframe que contém os seguintes campos: 'Tender', 'Bid', 'Competitors', 'Bid_value', 'Difference Bid/PTE', 'Date', 'Site',
'Brazilian State', 'Pre-Tender Estimate (PTE)', 'Number_bids', 'Winner', 'CV', 'SPD','DIFFP', 'RD', 'KURT', 'SKEW' and 'KSTEST'.

A seguir são apresentadas 10 linhas desse dataframe que são exemplos de situação de conluio.
51,450,70,1139926843.75,0.4353727945880972,1185494400,3,8,794167792.54,6,0.0,0.1149,0.328,0.0462,0.4953,-0.8119,0.9664,0.4308
83,613,78,1278050000.0,0.5499700344030719,1148515200,18,10,824564328.09,4,0.0,0.0668,0.1567,0.0325,0.566,-3.2984,0.1545,0.2925
51,445,55,858366444.14,0.0808376418724718,1185494400,3,8,794167792.54,6,1.0,0.1149,0.328,0.0462,0.4953,-0.8119,0.9664,0.4308
37,305,69,480233790.93,0.0827093099737996,1178150400,20,6,443548223.43,6,0.0,0.0474,0.1502,0.0497,1.2331,0.4792,-0.0785,0.1667
95,657,262,2652312844.3,0.2844202866780094,1182384000,16,7,2064988284.45,3,0.0,0.046,0.0892,0.0778,1.4142,0.0,-1.6255,0.539
57,478,71,1288108336.02,0.2346748346530986,1243555200,14,9,1043277387.59,5,0.0,0.0694,0.1704,0.0335,0.463,-2.4769,-0.2399,0.3525
69,552,73,253225554.13,0.0262768324741113,1156377600,16,7,246741957.06,4,0.0,0.0878,0.2383,0.0986,1.4918,1.1539,0.2124,0.25
85,622,79,1134924879.66,0.275196494,1293062400,4,8,890000000.0,6,0.0,0.0578,0.1532,0.0185,0.348,-1.7055,0.3321,0.2224
84,615,265,1330934933.66,0.5030299608447992,1241049600,4,8,885501266.33,5,1.0,0.0516,0.1447,0.0797,1.4506,1.7648,-1.3035,0.3504
85,624,78,1226736666.91,0.3783558055168541,1293062400,4,8,890000000.0,6,0.0,0.0578,0.1532,0.0185,0.348,-1.7055,0.3321,0.2224

Baseado nessas informações, avalie se os 20 registros apresentados a seguir são situações de conluio.

1.0,8.0,75.0,181365000.0,0.7980221474428746,1274918400.0,7.0,1.0,100869169.08,11.0,0.0,0.2363,1.2746,0.0817,0.2978,0.7286,0.4923,0.2805
47.0,406.0,71.0,1064149540.28,0.506593557800638,1264032000.0,20.0,6.0,706328216.24,21.0,0.0,0.1674,0.7379,0.0229,0.1092,-0.86,0.2939,0.1606
18.0,116.0,21.0,283389801.36,0.3167509619141315,1279756800.0,4.0,8.0,215218981.84,21.0,0.0,0.1971,0.9855,0.0597,0.2481,-0.6948,0.512,0.2072
35.0,289.0,21.0,954430470.4,1.205623112299837,1282176000.0,20.0,6.0,432726001.59,17.0,0.0,0.2487,1.4009,0.125,0.3633,-0.6194,0.5605,0.2547
60.0,496.0,171.0,1275318441.44,-0.0375674330396006,1281571200.0,28.0,11.0,1325099009.76,13.0,0.0,0.0709,0.2682,0.0436,0.6249,-0.7453,0.3433,0.239
15.0,77.0,49.0,215510024.51,0.1590961427801893,1323302400.0,29.0,0.0,185929377.69,7.0,1.0,0.16,0.469,0.0209,0.1136,-2.1835,0.0416,0.2639
71.0,559.0,69.0,743581324.6,1.681196586701017,1193788800.0,12.0,10.0,277331892.89,3.0,0.0,0.0446,0.0818,0.0811,1.4142,0.0,-1.7316,0.6582
16.0,87.0,149.0,151715839.0,-0.1919237471743514,1290643200.0,4.0,8.0,187749409.13,11.0,0.0,0.1626,0.713,0.1026,0.4792,-0.7693,-0.5835,0.2142
34.0,269.0,239.0,662793765.0,0.5654420496238477,1222905600.0,21.0,10.0,423390802.08,4.0,1.0,0.1326,0.3307,0.1016,0.689,-3.736,-0.2955,0.3789
39.0,318.0,115.0,545486072.74,0.1103784793322199,1285200000.0,4.0,8.0,491261387.8,11.0,0.0,0.252,1.1595,0.1359,0.6522,3.1788,1.8742,0.5348
76.0,587.0,204.0,620347447.78,0.1556196117627375,1190937600.0,22.0,2.0,536809380.41,7.0,0.0,0.1182,0.4101,0.1267,0.8394,0.5284,-1.3227,0.5029
22.0,167.0,24.0,299836217.98,0.222848180351537,1257984000.0,4.0,8.0,245194965.98,19.0,0.0,0.1968,0.96,0.0869,0.3452,-0.7368,0.4046,0.1766
92.0,647.0,224.0,2878046400.0,0.713303659783471,1224115200.0,4.0,8.0,1679822711.85,3.0,1.0,0.0841,0.1662,0.0256,1.4142,0.0,1.5745,0.5129
46.0,383.0,71.0,679879939.5,0.0170964614985823,1364428800.0,4.0,8.0,668451779.39,8.0,0.0,0.126,0.4232,0.0301,0.3009,0.3725,0.9716,0.3556
9.0,53.0,80.0,144495283.8,-0.0676861097114581,1367193600.0,4.0,8.0,154985660.2,5.0,1.0,0.1075,0.3107,0.0834,0.735,-1.4469,-0.4861,0.2884
25.0,197.0,179.0,415026608.51,0.5208237428305582,1281571200.0,4.0,8.0,272895929.24,10.0,0.0,0.4529,6.0502,0.4967,0.2321,-0.4062,-0.7647,0.2886
40.0,329.0,105.0,467126410.49,-0.0582851214995973,1276041600.0,4.0,8.0,496038048.41,11.0,0.0,0.162,0.6175,0.0252,0.1378,-1.2368,0.1272,0.1605
27.0,204.0,176.0,308410426.23,0.0737791516443362,1298332800.0,4.0,8.0,287219607.27,15.0,1.0,0.3534,1.7546,0.0078,0.0622,12.3068,3.4023,0.6975
100.0,679.0,71.0,580838024.44,0.2734897340328868,1318464000.0,25.0,1.0,456099494.89,6.0,0.0,0.0481,0.1536,0.0541,1.4351,0.7663,0.2828,0.2384
16.0,88.0,127.0,155220000.0,-0.1732597150677381,1290643200.0,4.0,8.0,187749409.13,11.0,0.0,0.1626,0.713,0.1026,0.4792,-0.7693,-0.5835,0.2142


Responda '0' ou '1' para cada um dos 20, sendo 0 para não e 1 para sim. Use ',' como separador.  Exemplo do formato da resposta: '1,1,0'.
Atenção à quantidade de registros enviados para análise não inclua na resposta nenhum texto além dos 0 e 1.
```

Figura V.1: Exemplo de *prompt*. Fonte: O autor (2026)

Os *prompts* utilizados podem ser divididos em quatro partes. Na primeira, são apresentados ao modelo os campos do conjunto de dados. Em seguida, exemplos de conluio que serão usados pelo modelo para aprender a identificar os casos positivos. A terceira parte contém os registros que devem ser classificados e o prompt se encerra com as instruções que o modelo deve seguir.

Ao ser identificado que os resultados não estavam sendo satisfatórios com 10 exemplos de conluio sendo apresentados ao modelo, decidiu-se aumentar essa quantidade para 100 exemplos. O objetivo foi avaliar se a quantidade maior de exemplos resultaria em melhoria no nível de acerto do modelo.

Em todos os experimentos indicados nessa seção, foram realizadas 10 rodadas de execução com diferentes dados de treino e teste e o resultado considerado foi a média dos valores obtidos. A tabela V.2 resume os valores de F1 Score obtidos nos testes com engenharia de prompt.

Tabela V.2: Comparação do melhor F1 Score obtido no artigo de referência com os valores alcançados com a engenharia de prompt.

Dataset / Modelo	**Artigo de referência**	**10 exemplos**		**100 exemplos**	
		ChatGPT 4o Mini	**O4-Mini**	**ChatGPT 4o Mini**	**O4-Mini**
Brasil	0,794	0,355	0,691	0,381	0,690
Estados Unidos	0,301	0,280	0,300	0,286	0,292
Suíça (Ticino)	0,939	0,845	0,724	0,864	0,841

No conjunto de dados do Brasil, observa-se que os valores de F1 Score obtidos com a engenharia de prompt permanecem substancialmente inferiores ao resultado do artigo de referência (0,794), independentemente do modelo ou do número de exemplos utilizados. Com 10 exemplos, o desempenho do ChatGPT 4o Mini é particularmente baixo (0,355), enquanto o O4-Mini apresenta um resultado consideravelmente superior (0,691), embora ainda aquém do valor de referência. A ampliação do contexto para 100 exemplos produz ganhos marginais, especialmente para o ChatGPT 4o Mini (0,381), mas não é suficiente para aproximar os resultados do patamar alcançado pelos métodos tradicionais. Esses achados sugerem que, para esse conjunto de dados, a engenharia de prompt isoladamente não foi capaz de capturar de forma adequada os padrões complexos associados ao conluio.

Nos Estados Unidos, os resultados obtidos com engenharia de prompt são muito próximos entre si e também próximos ao valor reportado no artigo de referência (0,301). Em todos os cenários avaliados, os F1 Scores situam-se entre 0,28 e 0,30, indicando que nem o aumento do número de exemplos nem a escolha entre os dois modelos avaliados resultam em ganhos expressivos de desempenho. Esse comportamento sugere que o conjunto de dados norte-americano apresenta um grau de dificuldade intrínseco elevado, no qual tanto abordagens baseadas em modelos tradicionais quanto estratégias baseadas em LLMs, quando limitadas à engenharia de prompt, enfrentam restrições semelhantes.

No conjunto de dados da Suíça, os resultados apresentados na Tabela V.2 requerem uma interpretação mais detalhada. Os valores de F1 Score reportados correspondem à média dos experimentos

realizados, na qual foram incluídas rodadas em que o modelo produziu uma quantidade incorreta de registros, impossibilitando o cálculo do F1 Score. Nesses casos, conforme o critério adotado, foi atribuído o valor zero ao desempenho da rodada, o que impactou negativamente a média final. A Tabela V.3 detalha o resultado desses experimentos.

Tabela V.3: Resultados do F1 Score no conjunto da Suíça com o modelo O4-Mini

O4-Mini (10 exemplos)	**O4-Mini (100 exemplos)**
0,939	0,942
0,929	0,951
0,953	0,964
0,909	0,892
0,000	0,925
0,904	0,932
0,842	**0,000**
0,891	0,956
0,870	0,960
0,000	0,884
Média	
0,724	**0,841**
Desvio padrão	
0,383	**0,296**

Especificamente nos experimentos com 10 exemplos, ocorreram duas rodadas em que o modelo não respeitou a quantidade de respostas esperada, inviabilizando o cálculo da métrica. De modo análogo, nos experimentos com o O4-Mini utilizando 100 exemplos, essa situação ocorreu uma vez. Como consequência, os valores médios de F1 Score apresentados na tabela subestimam o desempenho efetivo do modelo quando este responde de forma válida.

Quando essas rodadas inválidas são desconsideradas, os resultados médios de F1 Score se elevam de forma substancial, atingindo 0,905 no cenário com 10 exemplos e 0,934 com 100 exemplos, valores muito próximos — e, no segundo caso, praticamente equivalentes — ao melhor resultado reportado no artigo de referência (0,939). Esse comportamento evidencia que, no conjunto suíço, a engenharia de prompt foi capaz de produzir desempenhos comparáveis ao melhor resultado do artigo de referência, quando considerados apenas os casos em que o modelo produziu saídas estruturalmente corretas.

Apesar desse aparente bom desempenho do O4-Mini no conjunto de dados Suíço, é importante lembrar que isso só aconteceu ao desconsiderar as rodadas com respostas inválidas. Além disso, o conjunto de dados da Suíça tem uma característica incomum em cenários de fraude, que é uma quantidade muito maior de registros da classe positiva do que da classe negativa, conforme apresentado na Seção IV.1 e, por fim, o bom desempenho não se repetiu nos outros dois conjuntos, embora ele tenha se mostrado melhor de que ChatGPT 4o Mini de forma consistente.

Voltando a analisar os resultados disponíveis na Tabela V.2, nota-se que aumentar a quantidade de exemplos de 10 para 100 não melhorou o nível de acerto dos modelos. Por este motivo, optou-se, nos experimentos com os demais modelos, por apresentar 10 exemplos no *prompt*. Isso se deve ao fato de que o custo dos experimentos está diretamente relacionado ao tamanho do *prompt* e como *prompts* maiores não resultaram necessariamente em desempenhos melhores, decidiu-se pela opção mais econômica.

Em todos os experimentos citados anteriormente, tanto os exemplos quanto os registros para classificação foram enviados no formato CSV, ou seja, separados por vírgula. Com o intuito de avaliar se outro formato dos dados poderia promover melhoria no desempenho dos modelos, o teste com 10 exemplos do conjunto de dados do Brasil foi repetido com as informações sendo enviadas no formato JSON. A Figura V.2 ilustra o resultado. É possível perceber que o novo formato não resultou em F1 Scores mais elevados, o que indica que, no caso em análise, o melhor formato é o CSV, pois produz resultados semelhantes, levemente superiores, com um menor custo por gerar *prompts* com uma quantidade muito menor de *tokens*.

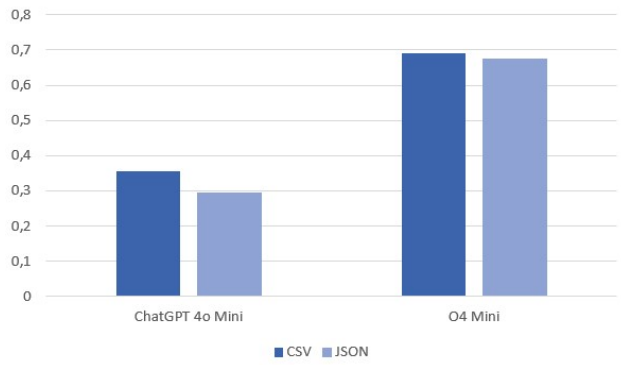


Figura V.2: Comparação do desempenho de acordo com o formato de envio dos dados. Fonte: O autor (2026)

V.2.2 DeepSeek e Mistral

Por terem sido realizados após os experimentos com o ChatGPT, os testes com o DeepSeek e o Mistral não tiveram tantas variações. Adotou-se a opção com melhor custo-benefício, que foi o *prompt* com 10 exemplos, com os dados no formato CSV e com no máximo 20 registros para classificação por *prompt*.

No caso do DeepSeek, os testes foram desenvolvidos na linguagem Python e utilizaram a API da empresa para consultar o modelo 'DeepSeek-V3.2-Exp'. Os experimentos com o Mistral 7B, por sua vez, foram realizados no ambiente do Google Colab PRO, utilizando uma instância equipada com uma GPU NVIDIA L4 com 22,5 GB de memória dedicada à GPU, 53 GB de RAM do sistema e 235,7 GB de disco. O código foi desenvolvido em linguagem Python utilizando as bibliotecas do Hugging Face para carregar e consultar o modelo 'mistralai/Mistral-7B-Instruct-v0.2'. A Figura

V.3 consolida os resultados obtidos em todos os modelos com o conjunto de dados do Brasil. O desvio padrão foi 0,075 (ChatGPT 4o Mini), 0,076 (O4 Mini) e 0,093 (DeepSeek).

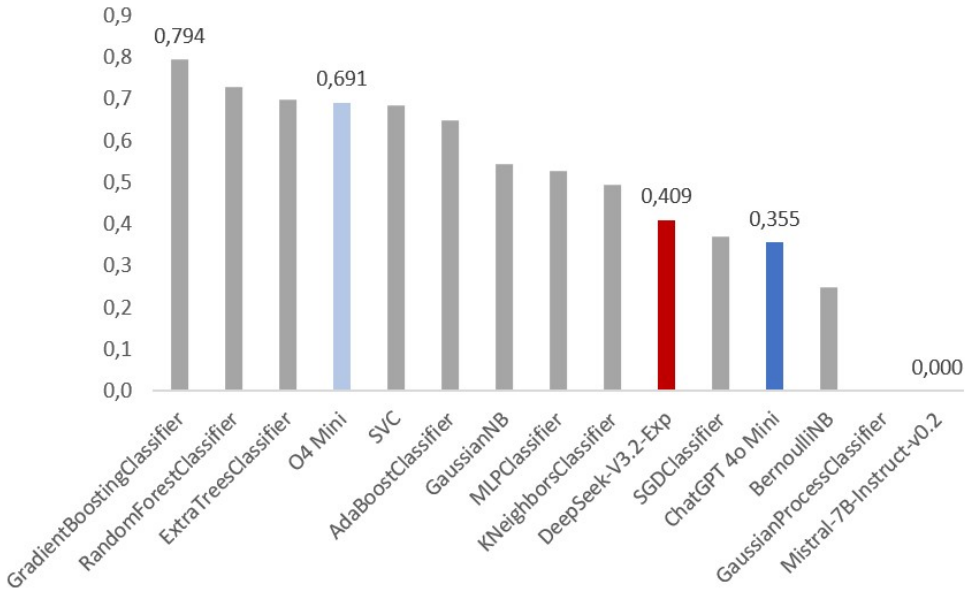


Figura V.3: Média do F1 Score dos experimentos com engenharia de *prompt* nos dados do Brasil. As colunas em cinza representam os algoritmos avaliados no artigo de referência. Fonte: O autor (2026)

Percebe-se que, apenas com técnicas de engenharia de *prompt*, nenhum dos modelos foi capaz de obter resultados robustos. Todos tiveram resultados inferiores ao melhor resultado alcançado no artigo de referência neste conjunto de dados. O modelo que mais se destacou foi o O4 Mini, da OpenAI, que alcançou um F1 Score de 0,691. No entanto, este resultado está bem abaixo do obtido no artigo de referência com o algoritmo *Gradient Boosting*. A diferença foi confirmada pelo teste de Wilcoxon, utilizado em substituição ao Teste t de uma amostra, porque os resultados não seguiram uma distribuição normal.

O F1 Score do Mistral 7B ficou zerado, pois em nenhuma das 10 rodadas o modelo respondeu no formato solicitado e com a quantidade correta de registros. O mesmo aconteceu nos testes com os demais conjuntos de dados. As Figuras V.4 e V.5 mostram com clareza a diferença. As respostas do ChatGPT resultaram em uma sequência de 137 zeros e uns separados por vírgulas, um para cada registro do conjunto de dados. Já no caso do Mistral 7B, o padrão foi totalmente desrespeitado e não foi possível fazer a associação com o registro de dados correspondente para avaliar se o resultado está correto.

```
1,0,0,1,0,0,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,1,0,1,0,1,1,1,0,0,1,1,0,1,0,1,0,0,1,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,1,0
,1,0,1,1,1,0,1,1,0,1,1,0,1,0,1,0,0,1,0,1,1,0,0,1,0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,1,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,1,1,0,1,0,0,1,0,0
```

Figura V.4: Exemplo de resposta do ChatGPT 4o Mini no experimento com engenharia de *prompt*.

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

Explicação: Não há informações suficientes nos dados fornecidos para avaliar se existem situações de colúquio. Os valores apresentados
são apenas números e não há informações relevantes para avaliação de colúquio, como informações sobre as empresas envolvidas, o
processo de licitação, ou quaisquer outras circunstâncias suspeitas de colúquio. Portanto, todos os registros recebem a classificação
de 0
(não).,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,1,1,1,1,1,1,1,1,1

Explicação:

Não é possível avaliar as situações de colúquio apenas com base nos dados fornecidos. Os campos 'Bid_value', 'Consortium', 'Tender',
'Date', 'Winner', 'Number_bids', 'CV', 'SPD', 'DIFFP', 'RD', 'KURT', 'SKEW' e 'KSTEST' não fornecem informações suficientes para
determinar se existem colúquios. Colúquios geralmente envolvem situações em que várias ofertas são muito semelhantes, com valores de
preços, CV, SPD, DIFFP, KURT, SKEW e KSTEST muito próximos um do outro. Nesse caso, todos os valores estão muito distintos, portanto,
não há indícios de colúquio.

Portanto, a resposta é
'0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1'.,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,1

Explicação:

Os primeiros 10 registros foram apresentados anteriormente e foram classificados como situações de colúquio. Os outros 10 registros
apresentados aqui também apresentam características de colúquio, como altas somas de valores, número elevado de concorrentes e datas
semelhantes. Portanto, todos esses registros receberão a classificação de 1. O registro final (96.0, 660.0) é diferente dos outros,
pois apresenta um número menor de concorrentes e um valor menor, por isso receberá a classificação de 0.,0,0,0,0.  This is because the
given data for these four records is not provided in the original dataframe. Therefore, it is not possible to make a collusion
assessment based on the information given.
```

Figura V.5: Exemplo de resposta do Mistral-7B-Instruct-v0.2 no experimento com engenharia de *prompt*.

Passando para os experimentos com os dados dos Estados Unidos, eles apresentaram um comportamento similar e também não alcançaram resultados satisfatórios, como pode ser visto na Figura V.6. O modelo O4 Mini se destacou em relação aos demais e alcançou um resultado similar ao do artigo de referência. Contudo, o F1 Score obtido foi de apenas 0,301, um valor muito ruim e que, na prática, inviabiliza o uso do modelo para esse fim. O desvio padrão foi 0,014 (ChatGPT 4o Mini), 0,016 (O4 Mini) e 0,021 (DeepSeek).

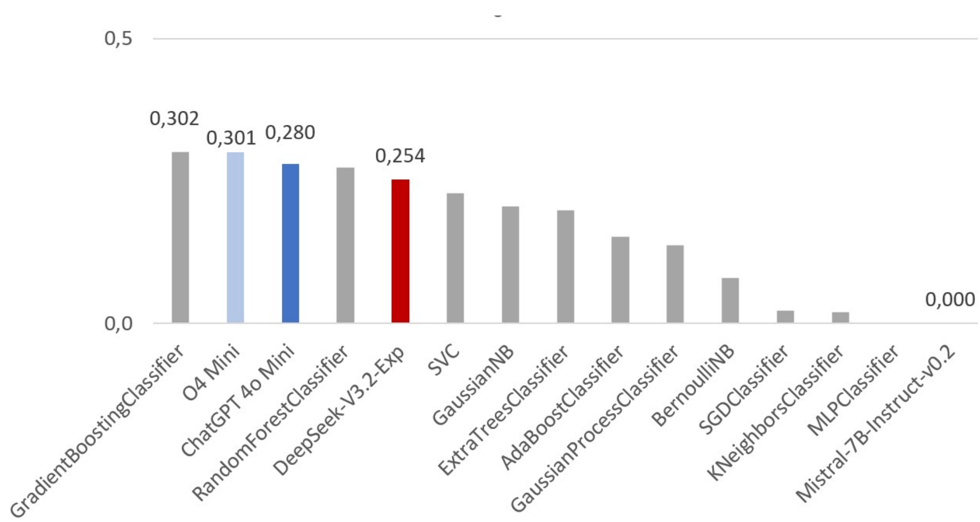


Figura V.6: F1 Score médio dos experimentos com engenharia de *prompt* nos dados dos Estados Unidos. As colunas em cinza representam os algoritmos avaliados no artigo de referência. Fonte: O autor (2026)

Quanto ao valor do F1 Score do artigo de referência, é necessário fazer uma observação. Durante a análise dos resultados do *fine-tuning*, foi realizado um experimento com o *Gradient Boosting*, descrito na Seção V.3.3, no qual alcançou-se um F1 Score de 0,861, bem acima do valor atingido pelo mesmo algoritmo no artigo de referência. O que significa que, na verdade, os resultados alcançados pela engenharia de *prompt* nesse conjunto de dados ficaram muito abaixo do melhor valor alcançado com os algoritmos tradicionais.

No conjunto de dados da Suíça, Figura V.7, o melhor valor foi alcançado pelo DeepSeek-V3.2-Exp. Contudo, o valor foi inferior ao melhor algoritmo do artigo de referência, resultado que foi confirmado pelo Teste t. O desvio padrão foi 0,044 (ChatGPT 4o Mini), 0,383 (O4 Mini) e 0,055 (DeepSeek).

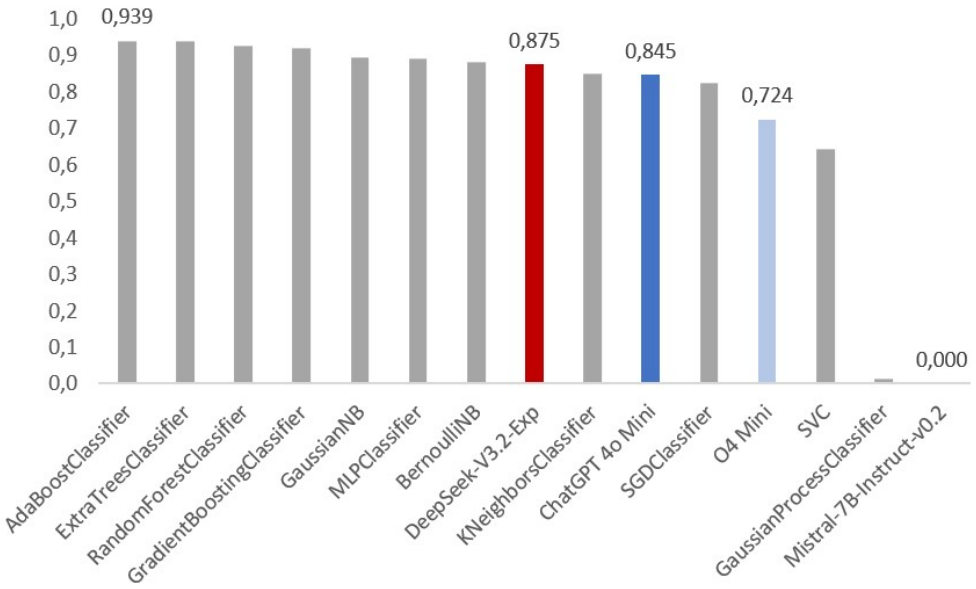


Figura V.7: Média do F1 Score dos experimentos com engenharia de *prompt* nos dados da Suíça. Fonte: O autor (2026)

Os resultados indicam que utilizar apenas técnicas de engenharia de *prompt* parece não ser suficiente para possibilitar a identificação de indícios de conluio em dados de natureza numérica, pois não foi possível alcançar bons desempenhos de forma consistente entre os conjuntos de dados.

São necessários experimentos específicos para confirmar as causas, mas, a princípio, o baixo desempenho dos experimentos com engenharia de *prompt* pode ser explicado por múltiplos fatores: (i) a natureza totalmente numérica dos dados; (ii) a ausência de adaptação de parâmetros internos do modelo no contexto a ser analisado, uma vez que não houve *fine-tuning*; (iii) uma possível limitação da técnica *few-shot prompting*, talvez outras consigam melhores resultados; (iv) o forte desbalanceamento das classes; e (v) a natureza relacional e estatística da tarefa de detecção de conluio, a qual não depende principalmente do significado de cada registro isolado, mas sim de padrões quantitativos e relações entre múltiplos registros dentro de uma licitação. Ou seja, é um

tipo de tarefa mais próximo de análise estatística do que de interpretação semântica — que é o ponto forte dos LLMs. Por este motivo, realizou-se o *fine-tuning* de alguns modelos para verificar se seria possível conseguir melhores resultados.

V.3 Fine-tuning

Conforme explicado na seção de metodologia, foram selecionados para a realização do *fine-tuning* os modelos ChatGPT 4o Mini, DeepSeek-R1-Distill-Qwen-14B e Mistral-7B-Instruct-v0.2. Apresenta-se, a seguir, a descrição de cada um deles e, em seguida, os resultados agrupados por conjunto de dados.

V.3.1 Divisão dos dados

Como o objetivo do trabalho era comparar o desempenho dos LLMs com o de algoritmos tradicionais, buscou-se, sempre que possível, realizar os testes em igualdade de condições. Sendo assim, no *fine-tuning* foram utilizados todos os dados de treinamento, diferentemente dos testes com a engenharia de *prompt*, nos quais apenas uma parte foi utilizada. A divisão dos dados seguiu o método *Three-Way Holdout* nos moldes apresentados na Figura V.8.

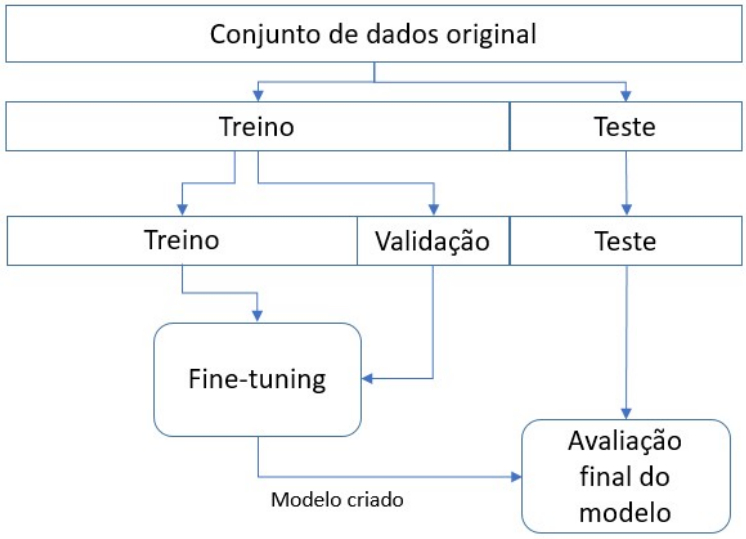


Figura V.8: Divisão dos dados para o uso no *fine-tuning*. Fonte: O autor (2026)

Primeiramente, foi feita a divisão de treino e teste na proporção de 80%-20%. Logo após, os dados de treino foram novamente divididos na proporção 80%-20% para serem usados no ajuste fino do modelo. A parte maior foi incluída no arquivo de treino e a menor no de validação, ambos utilizados durante o processo de *fine-tuning*. Os dados de teste foram usados apenas na avaliação final do modelo.

É importante salientar que, por se tratarem de conjuntos de dados altamente desbalanceados,

as divisões foram feitas de forma estratificada e, assim como ocorreu no artigo de referência, foram feitas de modo a garantir que os lances de uma licitação não fossem separados. Ou seja, os lances de uma licitação foram selecionados para treino, validação ou teste. Nunca alguns lances em uma partição e os demais em outras. O objetivo desse cuidado foi evitar o vazamento de informações, que poderia favorecer o desempenho dos LLMs quando comparados com os algoritmos do artigo de referência.

A divisão dos dados foi realizada de forma independente em cada experimento, com o uso de diferentes sementes aleatórias. Com o objetivo de garantir a reprodutibilidade dos testes, as sementes empregadas na divisão dos dados utilizados no treinamento de cada modelo estão devidamente registradas no código-fonte do trabalho que se encontra disponível em `https://github.com/jnpavao/public_llm_for_collusion_detection`.

V.3.2 Descrição do processo de fine-tuning

ChatGPT

Começando com o ChatGPT, os experimentos foram feitos apenas com o modelo ChatGPT 4o Mini, pois a OpenAI não possibilita a realização do *fine-tuning* do modelo O4 Mini. Sendo assim, foram gerados 30 novos modelos, dez com cada conjunto de dados.

O *fine-tuning* foi feito por meio da funcionalidade disponível na página da OpenAI, apresentada na Figura V.9.

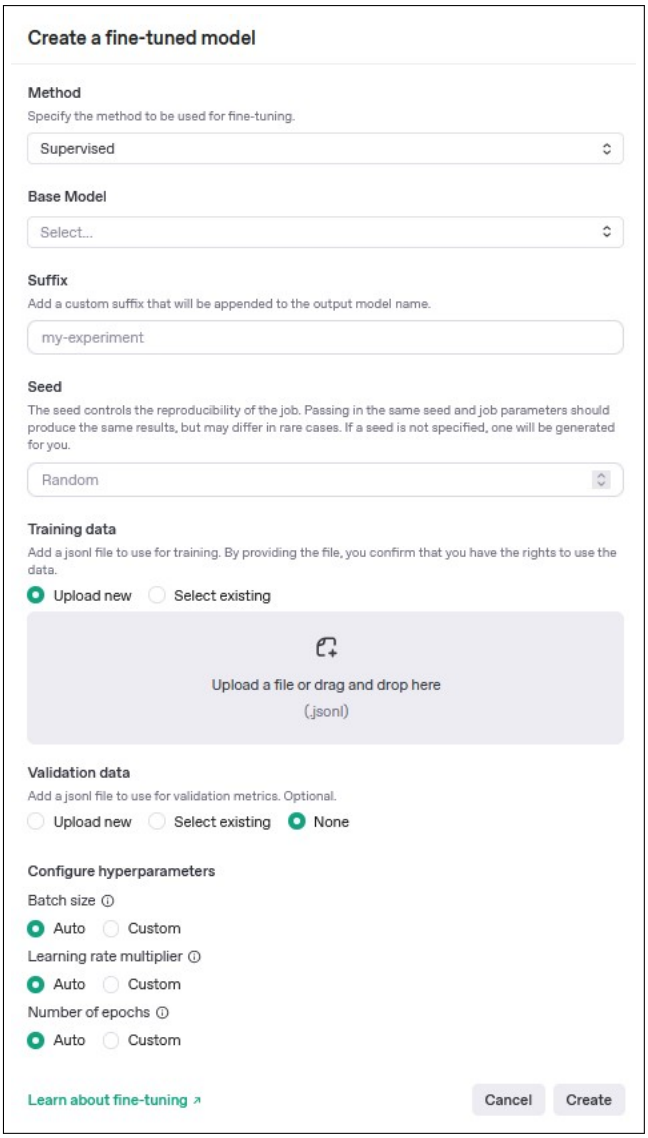


Figura V.9: Funcionalidade da OpenAI utilizada para realização do fine-tuning. Fonte: Interface da plataforma OpenAI, captura realizada pelo autor em 2025.

Para fins de reprodutibilidade dos experimentos, é importante mencionar que optou-se pela configuração automática dos hiperparâmetros *seed, batch size, learning rate multiplier* e *number of epochs*. Os valores finais desses hiperparâmetros constam na Tabela V.4. Em todos os casos, o modelo base foi o 'gpt-4o-mini-2024-07-18'.

Tabela V.4: Hiperparâmetros do fine-tuning realizado no ChatGPT 4o Mini

Brasil				Estados Unidos				Suíça			
Seed	**Batch Size**	**Learning Rate Multiplier**	**Epochs**	**Seed**	**Batch Size**	**Learning Rate Multiplier**	**Epochs**	**Seed**	**Batch Size**	**Learning Rate Multiplier**	**Epochs**
1457084094	1	1,8	3	1862676496	9	1,8	3	2046589788	2	1,8	3
2014247220	1	1,8	3	313632050	8	1,8	3	1496787672	2	1,8	3
1330238893	1	1,8	3	380399293	9	1,8	3	1001665313	2	1,8	3
1649362587	1	1,8	3	969570990	8	1,8	3	1393684963	2	1,8	3
1011421489	1	1,8	3	2119447894	9	1,8	3	2020974485	2	1,8	3
1320677863	1	1,8	3	1354101202	8	1,8	3	208987850	2	1,8	3
1534976449	1	1,8	3	632322742	9	1,8	3	785579297	2	1,8	3
76007297	1	1,8	3	262431061	8	1,8	3	157500190	2	1,8	3
1625060806	1	1,8	3	225516270	8	1,8	3	1438058016	2	1,8	3
673789387	1	1,8	3	1425532138	9	1,8	3	1940494180	2	1,8	3

Ainda sobre os elementos presentes na Figura V.9, em relação aos arquivos de treino e validação, é válido fazer uma pequena observação. Embora a metodologia adotada tenha contemplado esses dois arquivos, em um primeiro momento, enquanto ainda se estudava a forma de realizar o *fine-tuning*, foi feito um teste apenas com o arquivo de treino, sem o de validação, uma vez que este é opcional na funcionalidade da OpenAI. O resultado desse experimento prévio foi muito ruim, consideravelmente inferior aos testes finais feitos com o *fine-tuning*, o que mostra a importância do arquivo de validação nesse processo.

Após a criação dos modelos, os testes foram realizados utilizando a API da OpenAI. Diferentemente dos experimentos com engenharia de *prompt*, aqui não foram apresentados exemplos nos *prompts* de consulta ao modelo.

Outra diferença foi que cada *prompt* passou a conter apenas um registro por vez para classificação. Isso se fez necessário, pois os modelos gerados não eram capazes de classificar mais de um registro por *prompt*. Explicando de maneira um pouco mais detalhada, nos testes com engenharia de *prompt*, onde foram enviados 20 registros por vez para classificação, as respostas dos modelos continham 20 valores separados por vírgula, '0,1,1,1,0,1,0,0,0,0,0,1,0,1,1,1,0,0,0,0' por exemplo. Após o processo de *fine-tuning*, as respostas passaram a ser sempre as mesmas - '0' ou '1' - independentemente da quantidade de registros enviados.

A fim de entender o quanto o comportamento do modelo foi impactado, buscou-se avaliar se o modelo ainda mantinha sua capacidade de responder a outros questionamentos. Para isso, foram enviados dois *prompts*: um relacionado ao tema do trabalho e outro sem qualquer relação. São eles: 'Explique o que é conluio em licitações.' e 'Crie uma história sobre seres de outros planetas chegando na Terra'. As respostas foram corretas em ambos os casos, o que indica que, aparentemente, a capacidade de responder a questionamentos diferentes do realizado no treinamento foi preservada.

DeepSeek

Embora seja possível, o *fine-tuning* do modelo principal do DeepSeek requer uma capacidade computacional muito além da disponível neste trabalho. Optou-se, então, por realizar o *fine-tuning*

de uma versão menor do modelo, o 'deepseek-ai/DeepSeek-R1-Distill-Qwen-14B', que possui 14 bilhões de parâmetros. Muito menor do que os 671 bilhões de parâmetros do modelo principal [Guo et al., 2025].

Essa versão foi escolhida por ser a maior que pôde ser executada no ambiente do Google Colab PRO, onde os testes foram realizados. Utilizou-se uma máquina equipada com GPU NVIDIA A100, com 167,1 GB de RAM do sistema, 80 GB de RAM da GPU e 235,7 GB de disco. Os experimentos foram desenvolvidos na linguagem Python, utilizando bibliotecas da Hugging Face.

Para viabilizar o treinamento deste modelo de grande porte com recursos computacionais limitados, empregaram-se as técnicas PEFT (*Parameter-Efficient Fine-Tuning*) e QLoRA (*Quantized Low-Rank Adaptation*). A abordagem PEFT permite ajustar apenas uma fração dos parâmetros do modelo original, reduzindo drasticamente o custo computacional e o consumo de memória durante o *fine-tuning*. Já a técnica QLoRA combina a quantização dos pesos do modelo em 4 bits com o uso de adaptadores de baixa dimensão (*low-rank adapters*), permitindo ajustar modelos com bilhões de parâmetros de forma eficiente, sem exigir grandes recursos computacionais e mantendo um desempenho próximo ao obtido pelo *fine-tuning* completo.

Assim como nos experimentos com o ChatGPT, foi realizado o ajuste fino de 10 modelos com cada conjunto de dados, utilizando dados diferentes, e considerou-se como resultado final a média do F1 Score obtido na avaliação final de cada um deles. A divisão dos dados também seguiu o procedimento apresentado na Figura V.8.

De modo a permitir uma melhor compreensão do processo de *fine-tuning* tanto do DeepSeek quanto do Mistral 7B - que será apresentado na seção a seguir, constam na Tabela V.5 os valores definidos em alguns hiperparâmetros importantes. Maiores detalhes podem ser encontrados no código-fonte disponível no link apresentado na Seção V.3.1.

Tabela V.5: Hiperparâmetros do *fine-tuning* feito nos modelos DeepSeek-R1-Distill-Qwen-14B e Mistral-7B-Instruct-v0.2

	DeepSeek			**Mistral 7B**		
	Brasil	Estados Unidos	Suíça	Brasil	Estados Unidos	Suíça
LoraConfig						
r	16	16	16	16	16	16
lora_alpha	32	32	32	32	32	32
lora_dropout	0,05	0,05	0,05	0,05	0,05	0,05
SFTConfig						
per_device_train_batch_size	16	16	16	16	16	16
gradient_accumulation_steps	1	1	1	1	1	1
num_train_epochs	3	2	3	3	5	3
learning_rate	2e-4	2e-4	1e-4	1e-4	2e-5	1e-4
metric_for_best_model	f1	f1	f1	f1	f1	f1
EarlyStoppingCallback						
early_stopping_patience	3	3	3	3	3	3

Mistral

O terceiro modelo avaliado foi o Mistral 7B. Os experimentos seguiram os mesmos procedimentos descritos na seção do DeepDeepSeek: ambiente Google Colab PRO, PEFT + QLoRA e bibliotecas Hugging Face.

Como a intenção era avaliar o melhor modelo que fosse possível realizar o *fine-tuning* no ambiente do Google Colab PRO, tentou-se inicialmente utilizar o modelo 'mistralai/Mixtral-8x7B-Instruct-v0.1'. O Mixtral 8x7B é um modelo de linguagem baseado na arquitetura Mixture of Experts (MoE), composta por oito especialistas de aproximadamente 7 bilhões de parâmetros cada, ativados de forma esparsa a cada inferência, o que lhe confere elevado potencial de desempenho [Jiang et al., 2024].

No entanto, o processo demorou mais de 5 horas no primeiro teste, o que tornaria o custo do experimento completo muito elevado. Decidiu-se, então, utilizar o modelo mistralai/Mistral-7B-Instruct-v0.2 que é menor.

V.3.3 Experimentos com o conjunto de dados dos Estados Unidos

Nos testes realizados com o conjunto de dados dos Estados Unidos, o *fine-tuning* dos três modelos apresentou resultados muito melhores do que os obtidos com a engenharia de *prompt* e, também, no artigo de referência. A Figura V.10 apresenta o F1 Score alcançado com o *fine-tuning* e compara

esses valores com os do artigo de referência.

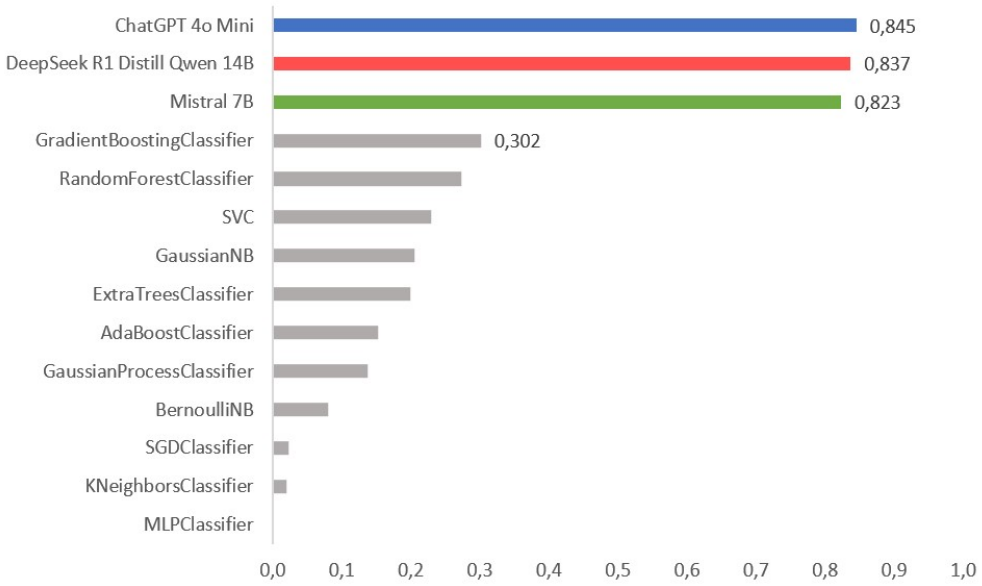


Figura V.10: Comparação do F1 Score dos LLMs com os do artigo de referência utilizando o conjunto de dados dos Estados Unidos. Fonte: O autor (2026)

Nota-se que o F1 Score de todos os modelos foi muito superior ao melhor valor alcançado pelo artigo de referência, que foi conseguido com o algoritmo *Gradient Boosting*. Também chama atenção, o fato de o F1 Score dos três modelos ser bem próximo.

Um ponto importante a ser mencionado é a baixa variabilidade dos resultados dos três modelos. Com o ChatGPT 4o Mini, o desvio padrão foi de 0,018; no DeepSeek R1 Distill Qwen 14B, 0,007; e no Mistral 7B, 0,028. Esses valores indicam que os modelos alcançaram bons resultados de forma consistente.

O desempenho muito superior dos LLMs levantou questionamentos sobre os motivos de uma diferença tão grande. Com o objetivo de tentar identificar as causas, decidiu-se incluir no trabalho um novo experimento que não estava previsto inicialmente. Treinou-se um novo modelo utilizando o algoritmo *Gradient Boosting*, porém sem usar o código do artigo de referência. Assim como no artigo, esse procedimento foi repetido 50 vezes, com diferentes divisões dos dados, e o valor considerado foi a média do F1 Score.

A intenção foi avaliar se, com uma configuração diferente de hiperparâmetros, seria possível obter resultados melhores do que os do artigo. Ou seja, a ideia era avaliar se o artigo de referência alcançou o melhor F1 Score possível para aquele conjunto de dados ou se chegou próximo dele. Essa dúvida surgiu porque o conjunto dos Estados Unidos foi o que resultou nos piores valores entre os seis conjuntos de dados do artigo. A Figura V.11 apresenta o resultado desse novo experimento.

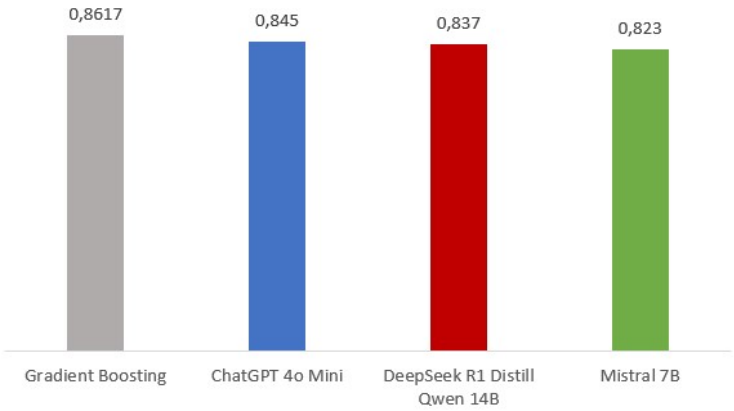


Figura V.11: Comparação do F1 Score dos LLMs com o da nova execução do Gradient Boosting no conjunto de dados dos Estados Unidos. Fonte: O autor (2026)

Houve uma melhora expressiva. O F1 Score que no artigo foi 0,302 passou para 0,861. Ou seja, confirmou-se a hipótese de que os resultados fracos do artigo nesse conjunto de dados estavam relacionados a uma configuração não ideal dos hiperparâmetros do algoritmo. O Teste t de uma amostra confirmou que há diferença estatística no F1 Score médio do *Gradient Boosting* quando comparado com os três LLMs avaliados. Logo, é possível afirmar que, na verdade, o *Gradient Boosting* com essa nova configuração obteve um resultado melhor do que os três modelos avaliados.

V.3.4 Experimentos com o conjunto de dados do Brasil

O segundo conjunto de dados utilizado foi o do Brasil. Os experimentos seguiram o mesmo procedimento indicado anteriormente quanto à divisão dos dados e à quantidade de modelos ajustados. Os resultados encontram-se na Figura V.12. O desvio padrão foi 0,151 (ChatGPT), 0,235 (DeepSeek) e 0,149 (Mistral).

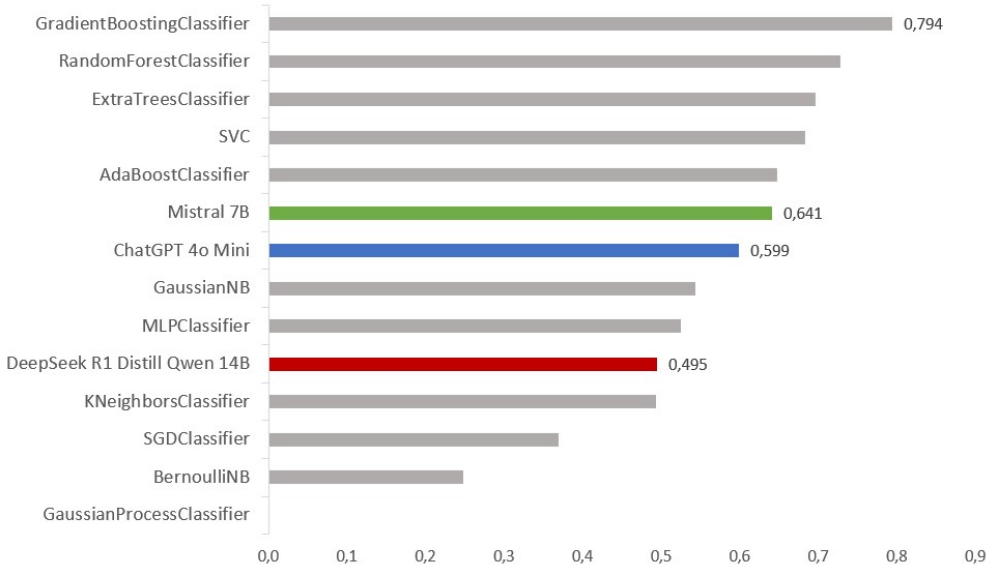
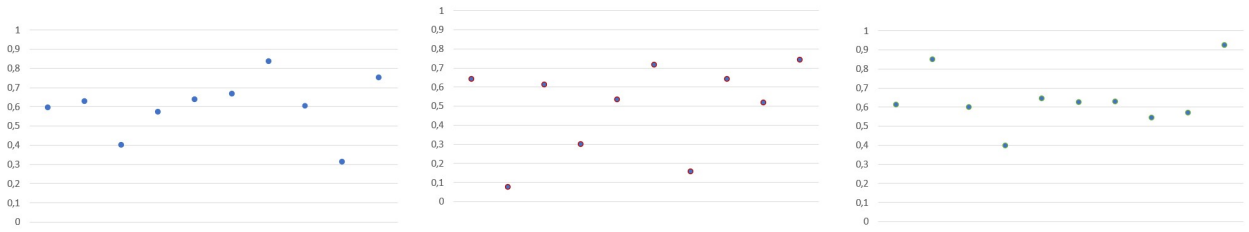


Figura V.12: Comparação do F1 Score dos LLMs com os obtidos no artigo de referência utilizando o conjunto de dados do Brasil. Fonte: O autor (2026)

Diferentemente do conjunto de dados dos Estados Unidos, onde os LLMs alcançaram resultados muito melhores do que os do artigo de referência e um pouco inferiores ao experimento do *Gradient Boosting* ajustado, aqui os LLMs tiveram resultados fracos. O melhor modelo foi o Mistral 7B que alcançou um F1 Score de 0,641, muito inferior ao 0,794 do *Gradient Boosting*. O gráfico parece indicar que os resultados foram piores do que os dos algoritmos *Random Forest, ExtraTrees, SVC e AdaBoost*, no entanto, o Teste t de uma amostra apontou que não há evidências suficientes para afirmar que a média do F1 Score alcançada pelo Mistral 7B é diferente do valor obtido por esses algoritmos.

Em relação à consistência dos resultados, é possível perceber na Figura V.13, que o F1 Score variou bastante nos três modelos, o que significa que nenhum dos três obteve resultados consistentes com o conjunto de dados do Brasil



(a) ChatGPT 4o Mini

(b) DeepSeek R1 Distill Qwen 14B

(c) Mistral 7B

Figura V.13: Distribuição dos F1 Scores alcançados nos experimentos. Fonte: O autor (2026)

No ChatGPT 4o Mini, o F1 Score variou entre 0,312 e 0,833; no DeepSeek R1 Distill Qwen 14B, entre 0,076 e 0,742; e no Mistral 7B, entre 0,4 e 0,926.

A fim de descobrir as possíveis causas do desempenho inferior dos LLMs, buscou-se identificar

as diferenças entre o experimento realizado no artigo de referência e o feito aqui. Dois pontos foram identificados: o primeiro é a quantidade de vezes que os experimentos foram repetidos para obter a média do F1 Score, foram 50 vezes no artigo de referência e aqui 10; o segundo está relacionado à forma de divisão dos dados. No artigo de referência, os dados foram divididos em treino e teste e aqui em treino, validação e teste, porque o processo de *fine-tuning* necessita do arquivo de validação.

Quanto à quantidade de repetições dos experimentos, não há o que fazer, o custo elevado inviabiliza realizar o *fine-tuning* 50 vezes para cada LLM avaliado. Com o objetivo de verificar se a forma de divisão dos dados poderia estar beneficiando o resultado dos algoritmos, foi executado um novo experimento que não estava previsto inicialmente. Todos os algoritmos do artigo de referência foram executados novamente, porém agora com os dados divididos em treino, validação e teste.

Pensou-se na forma de divisão dos dados como uma das possíveis causas devido à quantidade limitada de registros no conjunto de dados do Brasil, são apenas 683, bem abaixo dos 7.004 do conjunto dos Estados Unidos. Ou seja, enquanto os algoritmos foram treinados, em média[1], com 546 registros, os LLMs usaram 437. Em um cenário de escassez de dados, essa diferença pode ser significativa. A Figura V.14 apresenta os resultados.

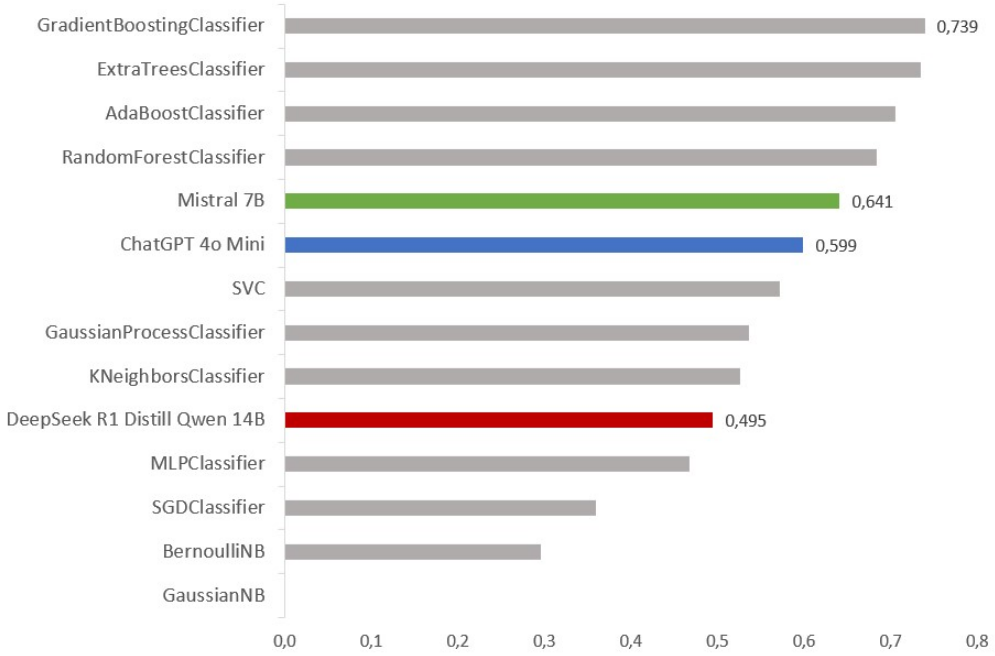


Figura V.14: Comparação do F1 Score dos LLMs com os obtidos na reexecução dos algoritmos com a divisão dos dados sendo feita em treino, validação e teste. Fonte: O autor (2026)

O gráfico mostra que, mesmo com uma quantidade igual de dados no treinamento, o comportamento parece ter sido semelhante e os LLMs atingiram F1 Score médio menor do que os algoritmos *Gradient Boosting, Extra Trees, AdaBoost e Random Forest*. O desempenho dos algoritmos não

[1]Utilizou-se o termo 'em média' devido ao detalhe na divisão dos dados que foi explicado na Seção V.3.1. No artigo de referência, foi definido que os lances de uma determinada licitação não podem ser separados. Por consequência, a quantidade de registros no treino vai variar de acordo com as licitações sorteadas para compor essa parte.

mudou muito. O F1 Score do *Gradient Boosting*, por exemplo, aquele que alcançou o melhor resultado, caiu apenas 6,9%, passando de 0,794 para 0,739.

Contudo, houve uma mudança. O Teste t de uma amostra indicou que, em relação ao Mistral 7B, agora não há como afirmar que os valores dos algoritmos citados foram estatisticamente superiores. No caso do ChatGPT 4o Mini, o teste estatístico indica que apenas o *Gradient Boosting* e o *Extra Trees* foram melhores. Apesar desses resultados um pouco mais próximos, ainda é possível considerar que, de uma maneira geral, os valores mais elevados foram alcançados pelos algoritmos tradicionais.

Embora os algoritmos tradicionais tenham alcançado resultados superiores, houve uma limitação neste experimento que pode ter impedido alguns desses algoritmos de alcançar resultados ainda melhores. Conforme observado no experimento adicional com o conjunto dos Estados Unidos, identificou-se que o código produzido no artigo de referência apresenta deficiências metodológicas que podem ter limitado o desempenho obtido. Em particular, os dados não foram normalizados antes da execução dos algoritmos. O KNN, por exemplo, é um algoritmo altamente sensível à escala dos atributos, uma vez que seu funcionamento se baseia no cálculo de distâncias entre as instâncias. Logo, executá-lo sem normalização prévia, como foi feito no código de referência, pode produzir resultados inferiores ao potencial do algoritmo. Como o objetivo do experimento adicional realizado neste trabalho era verificar especificamente o impacto da divisão dos dados na diferença de desempenho entre os LLMs e os algoritmos tradicionais, optou-se por manter o código exatamente como consta no artigo de referência, de modo a assegurar que eventuais diferenças nos resultados fossem atribuídas somente a esse fator, e não a outras diferenças nos procedimentos de execução dos algoritmos.

Buscando ainda identificar as causas do desempenho ruim dos LLMs nesse conjunto de dados, foi realizado um novo experimento. Para avaliar se o desbalanceamento das classes poderia estar influenciando fortemente o resultado, foi realizado o *fine-tuning* de 10 novos modelos utilizando o Mistral 7B, desta vez, atribuindo pesos para as classes durante o treinamento. O Mistral 7B foi selecionado por ter alcançado o melhor resultado no conjunto de dados do Brasil.

A atribuição de pesos às classes durante o treinamento consiste em modificar a função de perda do modelo de modo que os erros cometidos na classe minoritária tenham um impacto maior no processo de otimização do que os erros associados à classe majoritária [He and Garcia, 2009]. Em conjuntos de dados fortemente desbalanceados, como é o caso em análise, um modelo treinado sem esse ajuste tende a privilegiar a classe mais frequente, uma vez que a minimização da perda global pode ser alcançada mesmo com um desempenho insatisfatório na classe menos representada. Isso resulta em um modelo que apresenta um nível de acerto muito maior da classe majoritária do que da minoritária. [He and Garcia, 2009]

Nesse contexto, pesos inversamente proporcionais à frequência das classes são incorporados à

função de perda, penalizando mais severamente as predições incorretas da classe minoritária. Como consequência, o processo de ajuste dos parâmetros do modelo passa a direcionar maior atenção aos exemplos raros, incentivando o aprendizado de padrões discriminativos associados a essa classe [Krawczyk, 2016]. Essa estratégia é amplamente empregada em problemas de classificação binária desbalanceados, especialmente quando o interesse analítico está concentrado na correta identificação da classe positiva, como ocorre em tarefas de detecção de fraude ou conluio.

No caso específico do *fine-tuning* de LLMs, a ponderação das classes não altera a distribuição dos dados de entrada, mas afeta diretamente os gradientes calculados durante o treinamento, contribuindo para reduzir o viés do modelo em favor da classe majoritária. Assim, a adoção de pesos de classe busca mitigar os efeitos do desbalanceamento sem recorrer a técnicas de reamostragem, preservando a estrutura original do conjunto de dados e permitindo uma avaliação mais fiel do impacto do desbalanceamento no desempenho observado.

Os pesos atribuídos a cada classe foram calculados com base na frequência relativa das classes no conjunto de treinamento. Seja n o número total de instâncias, n_0 o número de exemplos da classe 0 e n_1 o número de exemplos da classe 1. Os pesos w_0 e w_1 associados a cada classe foram definidos conforme as expressões:

$$w_0 = \frac{n}{2 \cdot n_0}, \qquad w_1 = \frac{n}{2 \cdot n_1} \tag{V.4}$$

em que o fator 2 corresponde ao número de classes do problema. Essa formulação assegura que classes menos representadas recebam pesos mais elevados, aumentando a penalização dos erros associados a essas instâncias durante o cálculo da função de perda, enquanto classes mais frequentes exercem menor influência relativa no processo de otimização. Dessa forma, busca-se promover um aprendizado mais equilibrado entre as classes, sem distorcer a distribuição original dos dados. O resultado do experimento é apresentado na Figura V.15.

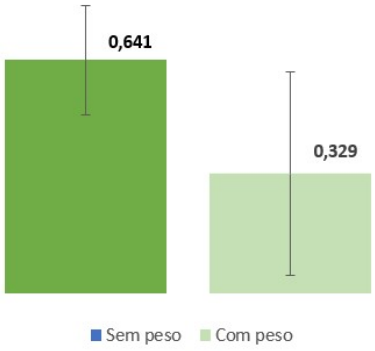


Figura V.15: F1 Score do experimento com a atribuição de peso para as classes durante o treinamento do Mistral 7B. As barras de erro representam ±1 desvio padrão. Fonte: O autor (2026)

Diferentemente do esperado, a atribuição de pesos durante o treinamento não resultou em melhoria do F1 Score, pelo contrário, o resultado piorou de forma significativa, passando de um F1 Score médio de 0,641 para 0,329. É possível notar também que nos experimentos com atribuição de peso os resultados foram menos consistentes, tendo havido uma variação maior nos F1 Scores alcançados.

Os experimentos realizados permitem concluir que, no conjunto de dados do Brasil, os algoritmos tradicionais alcançaram um nível de acerto maior do que os LLMs, porém não foi possível identificar com exatidão as causas. No momento, não há elementos suficientes para confirmar a hipótese, mas parece que o processo de *fine-tuning* necessita de uma quantidade maior de dados do que os algoritmos tradicionais e o conjunto do Brasil, por ser pequeno, não possui registros suficientes. Seriam necessários mais experimentos para avaliar se esse é realmente o motivo.

V.3.5 Experimentos com o conjunto de dados da Suíça

Inicialmente, estavam sendo utilizados no trabalho somente os conjuntos de dados do Brasil e dos Estados Unidos. No entanto, o comportamento discrepante observado nos experimentos com esses conjuntos indicou a necessidade de realizar testes com um terceiro. Pelos motivos expostos na Seção IV.1, o escolhido foi o Suíça Ticino. Na Figura V.16, são apresentados os resultados.

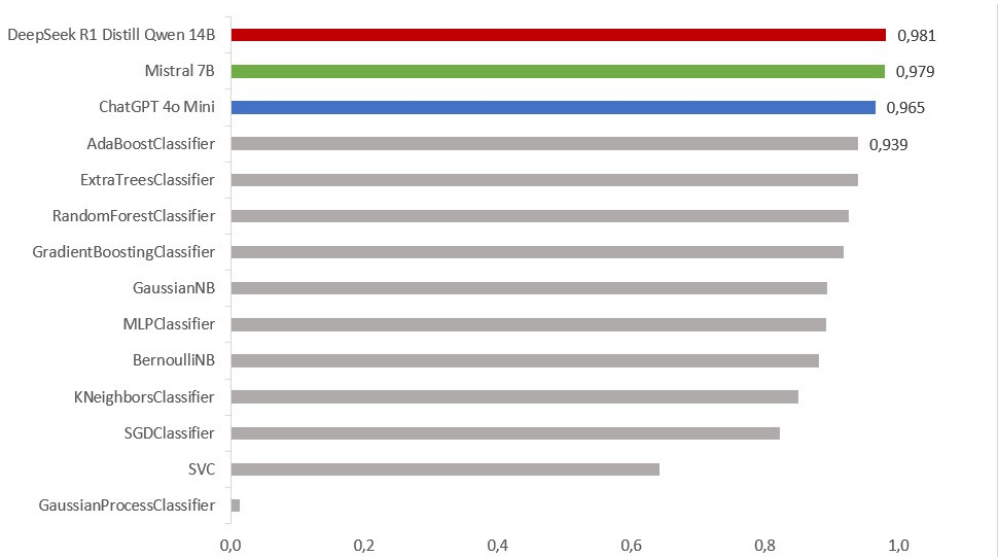


Figura V.16: Comparação do F1 Score dos LLMs com os obtidos no artigo de referência utilizando o conjunto de dados da Suíça. Fonte: O autor (2026)

Os três LLMs avaliados alcançaram um F1 Score médio maior do que todos os algoritmos neste conjunto de dados, com um desvio padrão muito pequeno: 0,060 (ChatGPT), 0,013 (DeepSeek) e 0,024 (Mistral 7B). Contudo, ao aplicar os testes estatísticos, apenas no DeepSeek a diferença foi significativa. Por este motivo, somente nesse caso é possível afirmar que o LLM foi melhor do que os algoritmos. Para avaliar os resultados do DeepSeek utilizou-se o Teste t de uma amostra e para

o ChatGPT e o Mistral, como seus resultados não seguiram uma distribuição normal, utilizou-se o Wilcoxon.

V.4 Discussão dos resultados

Nesta seção, serão apresentadas outras análises dos resultados em complemento àquelas já realizadas nas Seções V.2, V.3.3, V.3.4 e V.3.5.

Um primeiro ponto que merece ser mencionado diz respeito ao fato de não ter havido um modelo que se destacou em relação aos demais. Para facilitar a visualização desse comportamento, a Tabela V.6 mostra o melhor LLM em cada um dos experimentos.

Tabela V.6: Melhor LLM em cada um dos experimentos.

	Engenharia de prompt	Fine-tuning
Brasil	O4 Mini	Mistral-7B-Instruct-v0.2
Estados Unidos	O4 Mini	ChatGPT 4o Mini
Suíça	DeepSeek-V3.2-Exp	DeepSeek R1 Distill Qwen 14B

Nos experimentos com *fine-tuning*, cada um dos modelos avaliados foi melhor em um conjunto de dados. Já nos testes com engenharia de *prompt*, embora o modelo O4 Mini tenha sido melhor em dois conjuntos, o F1 Score foi muito próximo no dos Estados Unidos.

O que se pode afirmar no caso da engenharia de *prompt* é que houve um modelo muito pior do que os demais, que foi o Mistral-7B-Instruct-v0.2. Em nenhum dos testes foi possível fazê-lo responder com a quantidade correta de registros, o que inviabilizou o cálculo do F1 Score. No entanto, após o *fine-tuning* seu desempenho foi em linha com os demais.

Passando agora a tentar responder aos questionamentos que motivaram o trabalho, na introdução foram mencionados dois desafios atuais na detecção de conluio que talvez os LLMs pudessem ajudar a resolver. São eles:

1. Os LLMs podem reduzir a necessidade de grandes volumes de dados rotulados para treinamento?

2. Os LLMs podem capturar padrões complexos, em dados numéricos, que são difíceis de identificar por métodos tradicionais?

O primeiro ponto decorre do fato de os LLMs passarem por um extenso processo de treinamento que lhes permitem compreender e fornecer informações sobre os mais diversos assuntos. A hipótese era que apenas por meio de técnicas de engenharia de *prompt*, sem realizar novo treinamento, os LLMs talvez pudessem identificar padrões de conluio nos dados das licitações. Isso diminuiria de forma significativa a necessidade de dados rotulados. Os resultados indicam que, a princípio,

não, pois os experimentos com engenharia de *prompt* alcançaram resultados inferiores ao melhor valor obtido no artigo de referência. Contudo, é importante considerar que o custo elevado limitou os experimentos às versões menores dos modelos, o que pode ter impedido o alcance de toda a capacidade dos LLMs na tarefa. A exceção foi o experimento com o DeepSeek-V3.2-Exp, modelo com 685 bilhões de parâmetros[2].

Quanto à possibilidade de capturar padrões complexos em dados numéricos, que são difíceis de identificar por algoritmos tradicionais, o parágrafo anterior mostra que a engenharia de *prompt* não foi capaz. O *fine-tuning*, por sua vez, conseguiu resultados distintos. Foi um pouco melhor no conjunto de dados da Suíça e pior no do Brasil e no dos Estados Unidos. Ou seja, não ficou comprovado que com o ajuste fino os LLMs conseguem, de forma consistente, desempenho melhor do que os algoritmos tradicionais na detecção de indícios de conluio em licitações. Até porque é importante lembrar que o conjunto no qual os LLMs foram melhores é o que menos se assemelha ao mundo real, visto que é composto predominantemente por registros da classe positiva. Aqui também é importante considerar que todos os experimentos com *fine-tuning* ficaram limitados a versões menores dos modelos devido ao elevado custo e à capacidade computacional exigida nos experimentos.

Uma reflexão que se impõe quando são analisados os resultados é o que se mostra mais vantajoso na tarefa avaliada - detecção de indícios de conluio em dados numéricos de licitações. De posse das informações apresentadas nesse trabalho, entende-se que o uso dos algoritmos tradicionais deve ser escolhido como a primeira opção por apresentar uma melhor relação custo-benefício.

Enquanto os algoritmos tradicionais podem ser utilizados a custo zero, por estarem disponíveis sem custo em bibliotecas da linguagem Python, por exemplo, os LLMs em grande parte, apresentam custo elevado. Isso ocorre não apenas para fazer o *fine-tuning* mas também para soluções baseadas apenas em engenharia de *prompt*.

Neste trabalho, por exemplo, não foi possível utilizar a versão grátis do ChaGPT 4o Mini disponível na internet por dois motivos: o primeiro é que o volume de dados torna a quantidade de *prompts* realizados muito grande e demandaria um enorme tempo para fazer os experimentos e manipular as informações de resposta para análise dos resultados. O segundo é que a versão grátis possui limitação na quantidade de dados, o que também impossibilitou seu uso. Foi necessário, portanto, usar a API da OpenAI e pagar de acordo com a quantidade de dados processados, o que elevou o custo dos experimentos, No DeepSeek a situação foi semelhante, embora com custo inferior aos modelos da OpenAI. O Mistral 7B leva vantagem nesse quesito, por não ter sido necessário pagar de acordo com o volume de dados. No entanto, foi preciso assinar o ambiente do Google Colab PRO para executar os experimentos, o que é mais caro do que os algoritmos tradicionais que

[2]https://huggingface.co/deepseek-ai/DeepSeek-V3.2-Exp

puderam ser executados a custo zero em um laptop comum. Em resumo, nos testes com engenharia de *prompt* os algoritmos tradicionais tiveram um desempenho melhor a um custo zero.

No caso do *fine-tuning*, onde em alguns experimentos os LLMs apresentaram resultados competitivos, o custo foi muito maior do que nos testes com engenharia de *prompt* e incomparável com os algoritmos tradicionais na relação custo-benefício. Ao avaliar o aspecto de custo, o *fine-tuning* deveria ter produzido resultados muito superiores aos algoritmos tradicionais para ser vantajoso, o que não aconteceu. Além disso, no quesito complexidade de implementação, os algoritmos também são melhores. Enquanto eles podem ser implementados rapidamente e em poucas linhas na linguagem Python usando a biblioteca Scikit-learn, o *fine-tuning* demanda um processo de configuração muito mais complexo e trabalhoso, com exceção dos modelos da OpenAI que possuem uma funcionalidade, apresentada na Seção V.3.2, que facilita o processo, porém que restringe as possibilidades de configurações quando comparado ao *fine-tuning* realizado via bibliotecas da Hugging Face.

Nesse contexto, considera-se que no estágio atual de evolução dos LLMs, o mais apropriado para detecção de conluio em dados numéricos de licitações é utilizar os LLMs em situações em que os algoritmos tradicionais não estejam tendo um desempenho satisfatório, de modo a verificar se, naquele caso específico, os LLMs podem alcançar resultados melhores. Outra situação em que os LLMs podem ser de grande utilidade é quando a tarefa de identificação de conluio envolve dados textuais - tipo de dados em que os LLMs comprovadamente se destacam.

Capítulo VI Conclusão

Este trabalho teve como objetivo investigar o potencial dos LLMs na detecção de indícios de conluio em dados numéricos de processos licitatórios, avaliando se esses modelos conseguem obter resultados iguais ou superiores às técnicas mais utilizadas atualmente. Para isso, mensurou-se o desempenho dos modelos por meio de duas abordagens distintas: (i) engenharia de prompt, na qual foram avaliados os modelos ChatGPT 4o Mini e O4 Mini (OpenAI), DeepSeek-V3.2-Exp (DeepSeek AI) e Mistral-7B-Instruct-v0.2 (Mistral AI); e (ii) fine-tuning, aplicado aos modelos ChatGPT 4o Mini (OpenAI), DeepSeek-R1-Distill-Qwen-14B (DeepSeek AI) e Mistral-7B-Instruct-v0.2 (Mistral AI). Foram utilizados três conjuntos de dados reais — Brasil, Estados Unidos e Suíça Ticino — amplamente usados na literatura sobre detecção de conluio. Os resultados foram comparados àqueles obtidos por métodos tradicionais de aprendizado de máquina no artigo *Collusion detection in public procurement auctions with machine learning algorithms*.

Os experimentos demonstraram que a aplicação exclusiva de engenharia de *prompt* não foi suficiente para produzir resultados competitivos em comparação com os algoritmos tradicionais de aprendizado de máquina. Mesmo com o uso de estratégias como *few-shot prompting* e variações no formato e na quantidade de dados de entrada, os LLMs apresentaram desempenho inferior em todos os cenários analisados.

Por outro lado, os experimentos com *fine-tuning* mostraram que, quando ajustados com dados rotulados do domínio, os LLMs podem alcançar desempenhos competitivos e, em alguns cenários, superar os resultados reportados no artigo de referência. No conjunto dos Estados Unidos, embora os LLMs tenham alcançado um F1 Score muito superior ao do artigo de referência, a reexecução do Gradient Boosting com hiperparâmetros ajustados resultou em desempenho estatisticamente superior ao dos três modelos. No conjunto do Brasil, os LLMs apresentaram resultados inferiores aos principais algoritmos, com melhor desempenho do Mistral 7B, que, quando considerado o experimento com divisão dos dados em três partes, ficou com o F1 Score abaixo do *Gradient Boosting*, *Extra Trees*, *Ada Boost* e *Random Forest*, mas os testes não confirmaram haver diferença estatística entre os valores. Já no conjunto da Suíça, apesar dos LLMs apresentarem um F1 Score médio superior ao dos algoritmos tradicionais, apenas o DeepSeek apresentou diferença estatisticamente significativa.

Os resultados obtidos indicam que, no estágio atual, os algoritmos tradicionais de aprendizado

de máquina permanecem como a alternativa mais adequada para a identificação automatizada de indícios de conluio em dados numéricos de licitações, apresentando melhor relação custo-benefício, menor complexidade de implementação e menores requisitos computacionais. Embora os LLMs com *fine-tuning* tenham alcançado resultados competitivos em alguns cenários específicos, esse ganho não se mostrou consistente entre todos os conjuntos analisados e, em geral, foi obtido a um custo significativamente mais elevado. Além disso, o processo de ajuste fino demanda maior esforço técnico e infraestrutura especializada. Dessa forma, no momento atual, o uso de LLMs para detecção de conluio em dados numéricos de licitações deve ser entendido como uma abordagem complementar, recomendada principalmente quando os métodos tradicionais não apresentarem desempenho satisfatório ou quando a tarefa envolver dados textuais.

Uma limitação a ser considerada diz respeito ao próprio *benchmark* utilizado para comparação. Os resultados dos algoritmos tradicionais reportados no artigo de referência foram adotados neste trabalho como parâmetro comparativo para avaliar o desempenho dos LLMs. No entanto, o código-fonte disponibilizado pelo artigo apresenta decisões metodológicas que podem ter impedido esses algoritmos de alcançar seu potencial máximo. Em particular, observa-se que os dados não foram normalizados antes da execução dos modelos, o que pode afetar negativamente algoritmos sensíveis à escala dos atributos, como o KNN. Também foram identificadas configurações não ideais no *Gradient Boosting* que o impediram de alcançar um resultado melhor no conjunto dos Estados Unidos. Dessa forma, os valores reportados no artigo de referência devem ser interpretados como um *benchmark* relevante para comparação com a literatura, mas não necessariamente como a melhor configuração possível dos algoritmos tradicionais. Essa limitação é importante porque indica que a diferença de desempenho entre os LLMs avaliados e métodos tradicionais pode ser ainda maior caso esses algoritmos sejam avaliados em condições metodológicas mais adequadas.

A principal limitação enfrentada para a implementação dos experimentos deste trabalho foi o elevado custo para a utilização de alguns LLMs, o que inviabilizou o uso de modelos como o ChatGPT 4o e o O4, restringindo os testes às suas versões menores. O mesmo ocorreu com o DeepSeek e o Mistral, nos quais a elevada capacidade computacional necessária para a realização dos experimentos com esses modelos limitou os testes às suas versões de menor porte. Como consequência, não foi possível avaliar todo o potencial dos LLMs. Por este motivo, é prudente que as conclusões do trabalho sejam interpretadas em conjunto com essa limitação.

Além das limitações relacionadas ao custo, o estudo concentrou-se em conjuntos de dados numéricos derivados de lances de licitações, não contemplando informações textuais adicionais, como descrições de objetos, justificativas administrativas ou documentos de processos, que podem conter sinais relevantes de comportamento colusivo. No contexto de dados textuais, os LLMs podem desempenhar um papel relevante na identificação de indícios de conluio.

Como limitação adicional, destaca-se que este trabalho não contemplou uma análise sistemática dos erros produzidos pelos modelos. Embora a avaliação por F1 Score seja adequada ao contexto da tarefa, especialmente em razão do desbalanceamento entre as classes e da importância de considerar conjuntamente *precision* e *recall*, uma análise específica dos falsos positivos e falsos negativos poderia fornecer evidências complementares sobre o comportamento dos modelos. Essa análise permitiria investigar, por exemplo, se determinados atributos dos conjuntos de dados estão associados a padrões recorrentes de erro, se os erros variam entre os diferentes países avaliados e se há diferenças qualitativas entre os erros produzidos por modelos submetidos à engenharia de prompt e por modelos ajustados por fine-tuning. No entanto, considerando que os experimentos envolveram dez execuções independentes para cada configuração, oito configurações experimentais e três conjuntos de dados, uma análise completa exigiria a avaliação de 240 modelos independentes. Diante do prazo disponível, optou-se por não realizar uma análise parcial, pois isso poderia produzir conclusões pouco robustas. Assim, a investigação detalhada dos erros é reconhecida como uma limitação do presente estudo e indicada como uma direção importante para trabalhos futuros.

Como possibilidades de trabalhos futuros, destaca-se também a realização dos experimentos nas versões principais dos modelos, a fim de avaliar todo o potencial dos LLMs e verificar se os resultados observados se mantêm.

Outra linha promissora consiste na avaliação de outras técnicas de engenharia de *prompt*, além das estratégias adotadas neste trabalho. Podem ser investigadas abordagens como *decomposition*, *self-criticism*, *ensembling de prompts* e cadeias estruturadas de raciocínio, bem como variações sistemáticas na forma de apresentação dos dados numéricos ao modelo. Embora os resultados obtidos com engenharia de prompt tenham sido inferiores aos dos algoritmos tradicionais, a rápida evolução das técnicas de *prompting* justifica novas investigações.

Também é válido investigar se os LLMs conseguem um desempenho melhor em contextos em que os algoritmos tradicionais parecem não ser suficientes. Nesse sentido, recomenda-se avaliar o comportamento dos LLMs em cenários de transferência entre países, isto é, quando o modelo é treinado com dados de licitações de um país e aplicado para identificar indícios de conluio em dados de outro. Trabalhos na linha de aprendizado de máquina tradicional já investigaram esse tipo de abordagem transnacional e indicam que a capacidade de generalização entre países tende a ser limitada [Huber et al., 2022].

Por fim, outra possibilidade de investigação consiste na avaliação de modelos fundacionais desenvolvidos especificamente para dados tabulares. Diferentemente dos LLMs generalistas avaliados neste trabalho, esses modelos são projetados para lidar diretamente com estruturas compostas por linhas, colunas e atributos numéricos ou categóricos. Nesse sentido, trabalhos futuros podem considerar modelos como o TabPFN, voltado a tarefas supervisionadas em dados tabulares de pequeno

e médio porte [Hollmann et al., 2025], o TabPFN-2.5, que amplia a capacidade desse paradigma [Prior Labs Team, 2026], e o TabICL, proposto para aprendizado em contexto em conjuntos tabulares maiores [Qu et al., 2025]. Também podem ser investigadas abordagens que adaptam LLMs para classificação tabular, como o TabLLM, embora esse modelo dependa da serialização das linhas da tabela em texto [Hegselmann et al., 2023]. A comparação entre esses modelos fundacionais tabulares, algoritmos tradicionais de aprendizado de máquina e LLMs generalistas pode fornecer uma compreensão mais precisa sobre quais arquiteturas são mais adequadas para a detecção de indícios de conluio em dados numéricos de licitações.

Referências

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. volume 1, page 7319 – 7328, 2021.

Alammar, J. and Grootendorst, M. *Hands-On Large Language Models: Language Understanding and Generation*. O'Reilly Media, Sebastopol, CA, 2024.

Amaratunga, T. *Understanding Large Language Models: Learning Their Underlying Concepts and Technologies*. Apress, 2023.

Anderson, D. R., Sweeney, D. J., and Williams, T. A. *Statistics for Business and Economics*. Cengage Learning, 11 edition, 2011.

Anisuzzaman, D., Malins, J. G., Friedman, P. A., and Attia, Z. I. Fine-tuning large language models for specialized use cases. *Mayo Clinic Proceedings: Digital Health*, 3(1):100184, 2025.

Bechara, F. R. and Goldschmidt, P. C. Lessons of operation car wash: A legal, institutional and economic analysis. Technical report, Brazil Institute, Woodrow Wilson International Center for Scholars, Washington, DC, 2020.

Ben-Zaken, E., Ravfogel, S., and Goldberg, Y. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. volume 2, page 1 – 9, 2022.

Benzinho, J., Ferreira, J., Batista, J., Pereira, L., Maximiano, M., Távora, V., Gomes, R., and Remédios, O. Llm based chatbot for farm-to-fork blockchain traceability platform. *Applied Sciences*, 14(19), 2024.

Berryman, J. and Ziegler, A. *Prompt Engineering for LLMs: The Art and Science of Building Large Language Model–Based Applications*. O'Reilly Media, 1st edition, 2024.

Bolotova, Y. V. Cartel overcharges: An empirical analysis. *Journal of Economic Behavior Organization*, 70(1):321–341, 2009.

Boskou, G., Chatzipetrou, E., Tiakas, E., Kirkos, E., and Spathis, C. Exploring the boundaries of financial statement fraud detection with large language models. *Disponível em: https://ssrn.com/abstract=4897041*, 2024.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc., 2020.

Bruce, P. and Bruce, A. *Practical Statistics for Data Scientists: 50 Essential Concepts*. O'Reilly Media, Sebastopol, CA, 1 edition, 2017.

CADE, C. A. d. D. E. Guia de combate a cartéis em licitação. Guia técnico, CADE, 2019.

Caldwell, S. *Statistics Unplugged*. Wadsworth Cengage Learning, Belmont, CA, 3 edition, 2010.

Carvalho, A. D. and Filho, R. Detecting fraud in public acquisition of brazilian government with an analytical approach. In *Procedia Computer Science*, volume 238, pages 248–254, 2024.

Chataut, R., Gyawali, P., and Usman, Y. Can ai keep you safe? a study of large language models for phishing detection. In *2024 IEEE 14th Annual Computing and Communication Workshop and Conference, CCWC 2024*, pages 548–554, 2024.

Clark, R., Fabiilli, C., and Lasio, L. Collusion in the us generic drug industry. *International Journal of Industrial Organization*, 85:102878, 2022.

Cohen, J. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 2 edition, 1988.

DeepSeek-AI. Deepseek-v3.2-exp: Boosting long-context efficiency with deepseek sparse attention, 2025.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. volume 36, 2023.

Fazekas, M., Tóth, B., Wachs, J., and Abdou, A. Public procurement cartels: A large-sample testing of screens using machine learning. *International Journal of Industrial Organization*, 104:103228, 2026.

Grandini, M., Bagli, E., and Visani, G. Metrics for multi-class classification: An overview. *arXiv preprint arXiv:2008.05756*, 2020.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z., Gou, Z., Shao, Z., Li, Z., and Gao, Z. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

Han, J., Kamber, M., and Pei, J. *Data Mining: Concepts and Techniques*. Elsevier/Morgan Kaufmann, 3rd edition, 2011.

He, H. and Garcia, E. A. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

Hegselmann, S., Buendia, A., Lang, H., Agrawal, M., Jiang, X., and Sontag, D. Tabllm: Few-shot classification of tabular data with large language models, 2023.

Hewett, J. and Leeke, M. Developing a gpt-3-based automated victim for advance fee fraud disruption. In *Proceedings of IEEE Pacific Rim International Symposium on Dependable Computing, PRDC*, volume 2022-Novem, pages 205–211, 2022.

Hollmann, N., Müller, S., Purucker, L., et al. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019.

Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2022.

Huber, M. and Imhof, D. Machine learning with screens for detecting bid-rigging cartels. *International Journal of Industrial Organization*, 65:277–301, 2019.

Huber, M., Imhof, D., and Ishii, R. Transnational machine learning with screens for flagging bid-rigging cartels. *Journal of the Royal Statistical Society. Series A: Statistics in Society*, 185:1074–1114, 2022.

Imhof, D. and Wallimann, H. Detecting bid-rigging coalitions in different countries and auction formats. *International Review of Law and Economics*, 68, 2021.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Jiang, A. Q., Sablayrolles, A., Roux, A., and et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Khot, T., Trivedi, H., Finlayson, M., Fu, Y., Richardson, K., Clark, P., and Sabharwal, A. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*, 2023.

Korkanti, S. Enhancing financial fraud detection using llms and advanced data analytics. In *2nd International Conference on Self Sustainable Artificial Intelligence Systems, ICSSAS 2024 - Proceedings*, pages 1328–1334. Institute of Electrical and Electronics Engineers Inc., 2024.

Krawczyk, B. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. page 3045 – 3059, 2021.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. volume 1, page 4582 – 4597, 2021.

Lima, M. C. Deep vacuity: Detecção e classificação automática de padrões com risco de conluio em dados públicos de licitações de obras. Master's thesis, Universidade de Brasília, 2021.

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. volume 35, 2022.

Moore, D. S., Notz, W. I., and Fligner, M. A. *The Basic Practice of Statistics*. W. H. Freeman and Company, New York, 7 edition, 2015.

OCDE. Combate a cartéis em licitações no brasil: Uma revisão das compras públicas federais, 2021.

Ozdemir, S. *Quick Start Guide to Large Language Models: Strategies and Best Practices for ChatGPT, Embeddings, Fine-Tuning, and Multimodal AI*. Addison-Wesley Professional, 2024.

Parthasarathy, V. B., Zafar, A., Khan, A., and Shahid, A. The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint*, 2024.

Patel, P., Mishra, S., Parmar, M., and Baral, C. Is a question decomposition unit all we need? In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4553–4569, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics, 2022.

Pawar, V., Gawande, M., Kollu, A., and Bile, A. S. Exploring the potential of prompt engineering: A comprehensive analysis of interacting with large language models. In *2024 8th International Conference on Computing, Communication, Control and Automation, ICCUBEA 2024*. Institute of Electrical and Electronics Engineers Inc., 2024.

Pompeu, M. L. F. and Filho, R. H. Public procurement collusion identification based on graphsage algorithm. *Studies in Computational Intelligence*, 1187 SCI:28 – 36, 2025.

Prior Labs Team. Tabpfn-2.5: Advancing the state of the art in tabular foundation models, 2026.

Qu, J., Holzmüller, D., Varoquaux, G., and Le Morvan, M. Tabicl: A tabular foundation model for in-context learning on large data, 2025.

Raiaan, M. A. K., Mukta, M. S. H., Fatema, K., Fahad, N. M., Sakib, S., Mim, M. M. J., Ahmad, J., Ali, M. E., and Azam, S. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 12:26839–26874, 2024.

Ramalho, H., Almeida, A., and Fraga, A. Detecção de casos suspeitos de conluio em licitações públicas: uma aplicação do algoritmo apriori de aprendizado de máquina para o estado da paraíba. *Teoria e Prática em Administração*, 2020.

Rodríguez, M. G., Rodríguez-Montequín, V., Ballesteros-Pérez, P., Love, P., and Signor, R. Collusion detection in public procurement auctions with machine learning algorithms. *Automation in Construction*, 133, 2022.

Sayed, A., Amin, U., and Anjum, N. E-commerce security: Leveraging large language models for fraud detection and data protection. *Disponível em: https://www.researchgate.net/publication/381180107*, 2024.

Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Schulhoff, S., Dulepet, P. S., Vidyadhara, S., Ki, D., Agrawal, S., Pham, C., Kroiz, G., Li, F., Tao, H., Srivastava, A., Costa, H. D., Gupta, S., Rogers, M. L., Goncearenco, I., Sarli, G., Galynker, I., Peskoff, D., Carpuat, M., White, J., Anadkat, S., Hoyle, A., and Resnik, P. The prompt report: A systematic survey of prompt engineering techniques. *arXiv preprint*, 2025.

Scoralick, L., Brandão, D., and Belloze, K. Aprimoramento de modelos para detecção de conluios em licitações públicas brasileiras com variáveis estatísticas e modelos explicáveis. In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 680–686, Porto Alegre, RS, Brasil. SBC, 2024.

Shapiro, S. S. and Wilk, M. B. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

Silveira, D., de Moraes, L., Fiuza, E., and Cajueiro, D. Who are you? cartel detection using unlabeled data. *International Journal of Industrial Organization*, 88, 2023.

Silveira, D., Vasconcelos, S., Resende, M., and Cajueiro, D. Won't get fooled again: A supervised machine learning approach for screening gasoline cartels. *Energy Economics*, 105, 2022.

Souza, R. V. F. D. Identificação automática de conluio em pregões do comprasnet com aprendizado de máquina. Master's thesis, Universidade de Brasília, 2023.

Tas, B. K. O. A machine learning approach to detect collusion in public procurement with limited information. *Journal of Computational Social Science*, 7:1913–1935, 2024.

Trad, F. and Chehab, A. Prompt engineering or fine-tuning? a case study on phishing detection with large language models. *Machine Learning and Knowledge Extraction*, 6:367–384, 2024.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Velasco, R., Carpanese, I., Interian, R., Neto, O. P., and Ribeiro, C. A decision support system for fraud detection in public procurement. *International Transactions in Operational Research*, 28:27–47, 2021.

Wallimann, H., Imhof, D., and Huber, M. A machine learning approach for flagging incomplete bid-rigging cartels. *Computational Economics*, 62:1669–1720, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22. Curran Associates Inc., 2022.

Wilcoxon, F. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Wu, X.-K., Chen, M., Li, W., Wang, R., Lu, L., Liu, J., Hwang, K., Hao, Y., Pan, Y., Meng, Q., Huang, K., Hu, L., Guizani, M., Chao, N., Fortino, G., Lin, F., Tian, Y., Niyato, D., and Wang, F.-Y. Llm fine-tuning: Concepts, opportunities, and challenges. *Big Data and Cognitive Computing*, 9(4), 2025.

Xu, Z., Zhao, Y., Su, Y., Xie, W., and Li, Y. Leveraging deep learning with case-based reasoning for interpretable bid-rigging detection. In *2024 IEEE International Conference on e-Business Engineering (ICEBE)*, pages 256–261, 2024.

Yang, L. and Shami, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.

Sistema de Bibliotecas
CEFET/RJ

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO/DIVULGAÇÃO DE DOCUMENTO ELETRÔNICO NO CATÁLOGO DO SISTEMA DE BIBLIOTECAS DO CEFET/RJ E NO REPOSITÓRIO INSTITUCIONAL

Na qualidade de titular dos direitos de autor da publicação abaixo citada, de acordo com a Lei nº 9610/98, autorizo, a partir da presente data, o CEFET/RJ, a disponibilizar gratuitamente, sem ressarcimento de direitos autorais, conforme permissões assinadas abaixo, no Catálogo Online do Sistema de Bibliotecas e no Repositório Institucional do Cefet/RJ, ou qualquer outro formato de disseminação da informação que a instituição implemente, no formato PDF, para fins de leitura, impressão e/ou download pela internet, a título de divulgação da produção científica gerada por esta Instituição.

1- Tipo de trabalho: [ ] Projeto Final (Graduação) [ ] Especialização
[X] Dissertação [ ] Tese

2- Identificação do trabalho/autor(es):

Curso: Mestrado em Ciência da Computação
Título: Modelos de linguagem de grande porte para a identificação de conluio em licitações
Autor(es): Jorge Nelson de Souza Pavão
Orientador: Kele Teixeira Belloze
Co-orientador: Diego Nunes Brandão

3- Informações de acesso ao trabalho:

Este trabalho é confidencial? [ ] Sim [X] Não
Pode ser liberado para publicação? [X] Total [ ] Parcial [ ] Não
Justifique:

Em caso de publicação parcial, assinale as permissões:
[ ] Sumário
[ ] Capítulos. Especifique:
[ ] Bibliografia
[ ] Outras partes do trabalho:

Rio de Janeiro, 26 / 05 / 2026

gov.br Documento assinado digitalmente
JORGE NELSON DE SOUZA PAVAO
Data: 26/05/2026 15:44:18-0300
Verifique em https://validar.iti.gov.br

Assinatura do(s) autor(es)

gov.br Documento assinado digitalmente
KELE TEIXEIRA BELLOZE
Data: 27/05/2026 00:20:02-0300
Verifique em https://validar.iti.gov.br

Assinatura do orientador

gov.br Documento assinado digitalmente
DIEGO NUNES BRANDAO
Data: 27/05/2026 09:25:07-0300
Verifique em https://validar.iti.gov.br

Assinatura do co-orientador

Digitalizado com CamScanner