

Análise Comparativa de Técnicas Avançadas de Deep Learning para Classificação de Sentimentos em Avaliações Multidomínio

Luciano Ayres Farias de Carvalho

*Centro de Informática (CIn), Universidade Federal de Pernambuco – UFPE,
Recife, PE, Brasil*

Email: lafc@cin.ufpe.br

Resumo

Este trabalho apresenta um estudo comparativo de técnicas avançadas de Deep Learning aplicadas à classificação de sentimento em avaliações de produtos multissetoriais em português. Foi desenvolvida uma solução completa denominada **NLP-Sentinel**, englobando desde o processamento dos dados reais de e-commerce (dataset **Olist Store**) até a construção de modelos de aprendizado profundo e uma aplicação web de análise de sentimento. Avaliou-se o desempenho de modelos de linguagem pré-treinados (BERT, RoBERTa, DistilBERT) fine-tunados para classificação binária positivo/negativo, em comparação com abordagens baseadas em *embeddings* semânticos combinados com classificadores tradicionais (SVM) e com modelos de última geração de *Large Language Models* (LLMs) em *zero-shot*. Os resultados mostram que o modelo BERT fine-tunado obteve a melhor performance dentre os supervisionados (acurácia ~94%, *F1-score* ~94%), superando marginalmente outros transformers e significativamente métodos clássicos. Abordagens híbridas como **Sentence-BERT + SVM** também alcançaram desempenho competitivo (~93% *F1*), evidenciando uma alternativa de menor custo computacional. Adicionalmente, o modelo generativo GPT-4 (OpenAI) atingiu 100% de acurácia em um cenário de teste limitado sem treinamento prévio, demonstrando o potencial de LLMs com *in-context learning*. O relatório descreve a metodologia, os experimentos conduzidos, a aplicação desenvolvida e discute qualitativa e quantitativamente os resultados obtidos. Concluímos destacando que é possível alcançar alta precisão na análise de sentimentos em português por diferentes estratégias, cabendo considerar trade-offs entre desempenho e custo computacional conforme o contexto de aplicação.

Palavras-chave

Processamento de Linguagem Natural; Análise de Sentimentos; BERT; Transformers; *Large Language Models*; Classificação Binária; Deep Learning; Português.

Introdução

A análise de sentimentos em textos de avaliações de produtos é uma tarefa crucial para empresas e consumidores, permitindo extrair insights sobre a satisfação do cliente de forma automatizada. Com o crescimento de plataformas de e-commerce multissetoriais, existem enormes volumes de **avaliações de consumidores** em formato de texto, tornando inviável a análise manual. Assim, técnicas de **Deep Learning** voltadas ao **Processamento de Linguagem Natural (PLN)** têm ganhado destaque por oferecerem melhor desempenho na classificação de sentimentos (polaridade positiva, negativa ou neutra) em comparação com abordagens baseadas em regras ou aprendizado de máquina tradicional.

Modelos de linguagem **pré-treinados** revolucionaram a área de PLN ao fornecer representações contextuais ricas. Em especial, o modelo **BERT** (Bidirectional Encoder Representations from Transformers) introduzido por Devlin *et al.* ([1810.04805] [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#)) ([1810.04805] [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#)) trouxe avanços significativos ao permitir **fine-tuning** para uma variedade de tarefas de linguagem natural com desempenho superior aos métodos anteriores. Desde então, várias variantes e aprimoramentos têm surgido, como o **RoBERTa** (uma otimização robusta do pré-treinamento do BERT) ([RoBERTa: A Robustly Optimized BERT Pretraining Approach](#)) e o **DistilBERT** (versão distilada e mais leve do BERT) ([DistilBERT, a distilled version of BERT: smaller, faster, cheaper and ...](#)) ([DistilBERT, a distilled version of BERT: smaller, faster, cheaper and ...](#)), que reduz o custo computacional mantendo grande parte da performance original. Paralelamente, para idiomas além do inglês, esforços de pré-treinamento específico foram realizados – no caso do português, destaca-se o **BERTimbau**, um BERT treinado em larga escala em corpora em português brasileiro ([Fábio Souza - dblp](#)). Esses modelos fornecem **representações de sentença** de alta qualidade, fundamentais para a tarefa de análise de sentimento em nosso contexto.

Este projeto, intitulado “**NLP-Sentinel**”, insere-se na disciplina de Projeto em Deep Learning (CIn-UFPE) e tem como objetivo principal **comparar diferentes técnicas avançadas de Deep Learning** para classificação de sentimentos em avaliações reais de produtos multissetoriais (multi-domínio) em português. O foco está em avaliar: (1) modelos *supervisionados* fine-tunados, (2) abordagens híbridas que utilizam *embeddings* semânticos com classificadores simples, e (3) o uso de **LLMs de última geração** (e.g. GPT-4 da OpenAI) em regime *zero-shot*, sem qualquer treinamento nos dados em questão. Busca-se não apenas alcançar alta **acurácia** na distinção de avaliações positivas e negativas, mas também extrair **insights** sobre *trade-*

offs de desempenho, custo computacional e viabilidade de cada abordagem em cenários práticos.

Adicionalmente, foi desenvolvida uma **aplicação web** interativa para demonstração em tempo real do melhor modelo resultante (BERT fine-tunado), permitindo ao usuário final inserir textos de avaliações e obter instantaneamente a classificação de sentimento acompanhada de grau de confiança. Essa aplicação visa tornar palpável o resultado do projeto, evidenciando a utilidade prática do modelo treinado.

Nas seções a seguir, apresentamos a **Metodologia** empregada, incluindo descrição do conjunto de dados e pré-processamento (Seção II), detalhamos os **Experimentos** conduzidos e modelos testados (Seção III), reportamos os **Resultados** quantitativos obtidos (Seção IV) e descrevemos o desenvolvimento da **Aplicação Web** (Seção V). Em seguida, discutimos os resultados com análises qualitativas, incluindo abordagens descartadas e lições aprendidas (Seção VI), e concluímos o trabalho destacando os principais achados e recomendações para trabalhos futuros (Seção VII).

Metodologia

A. Conjunto de Dados e Preparação

Para treinar e avaliar os modelos, foi utilizado o *dataset* público **Olist Store** ([v2_06_abr_11_l2_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto\(2\).py](#)), que consiste em dados de um e-commerce brasileiro abrangendo múltiplas categorias de produtos (multidomínio). Este conjunto inclui mais de **100 mil avaliações de clientes** feitas entre 2016 e 2018, contendo para cada pedido uma nota de **1 a 5 estrelas** (*review score*) e um texto livre de avaliação do produto/serviço pelo consumidor. Os dados passaram por um processo de anonimização – nomes próprios foram substituídos por identificadores fictícios (por exemplo, alusões a personagens de *Game of Thrones*) – garantindo privacidade e até permitindo alguma criatividade na análise textual.

Para efeito de **análise de sentimento**, as avaliações foram categorizadas de forma **binária** em *positivas* vs. *negativas*. Definimos como **avaliações positivas** aquelas com nota alta (4 ou 5 estrelas) e **negativas** as de nota baixa (1 ou 2 estrelas). Avaliações neutras ou ambíguas (caso das notas 3 estrelas) foram excluídas do treinamento, de modo a polarizar bem as classes e facilitar o aprendizado de padrões claros de satisfação vs. insatisfação. Após essa filtragem, obteve-se um conjunto balanceado de aproximadamente 40 mil avaliações positivas e 20 mil negativas (havia significativamente menos avaliações de 1-2 estrelas no dado bruto).

O texto das avaliações passou por **pré-processamento** básico para limpeza e normalização. Foram removidos caracteres estranhos ou não textuais (p.ex. emojis ou símbolos sem significado claro), corrigidos eventuais erros ortográficos simples e padronizadas certas expressões. Mantiveram-se, contudo, aspectos importantes da linguagem natural em português, como acentuação, pontuação e caixa alta/baixa, uma vez que o modelo base escolhido (BERTimbau cased) diferencia maiúsculas e minúsculas e lida internamente com pontuação através de tokenização subword. Também não foram removidas *stopwords* nem realizada lematização, pois modelos Transformers tendem a capturar bem o contexto sem necessidade dessas etapas manuais de PLN. Em vez disso, confiou-se na tokenização do próprio modelo pré-treinado para segmentar o texto em subtokens apropriados.

Para mitigar o desbalanceamento entre classes durante o treinamento supervisionado, adotou-se a estratégia de **ponderação de classes na função de perda**. A classe minoritária (negativa) recebeu um peso maior proporcional à razão de desequilíbrio, fazendo com que erros em avaliações negativas fossem mais penalizados que erros em positivas. Essa técnica permitiu um treinamento mais justo sem duplicar dados (oversampling) nem descartar exemplos (undersampling). Além disso, os dados foram divididos em conjuntos de treino (80%), validação (10%) e teste (10%) de forma estratificada, garantindo proporções semelhantes de classes em cada parte. A avaliação final dos modelos supervisionados foi realizada no conjunto de teste reservado, completamente separado dos dados de treinamento, para obter métricas de generalização confiáveis.

B. Modelos e Abordagens

Foram exploradas diversas abordagens de modelagem, desde métodos tradicionais de aprendizado de máquina até os mais recentes modelos de linguagem, conforme resumido a seguir:

- **SVM + Bag of Words (BoW):** Como *baseline* inicial, considerou-se um classificador linear SVM usando vetores de palavras (BoW) gerados a partir dos textos. No entanto, esta abordagem apresentou desempenho fraco em experimentos preliminares (acurácia abaixo de 80%) e foi rapidamente descartada em favor de representações mais ricas. *Insights:* BoW ignora contexto e ordem das palavras, limitando sua eficácia em texto curto e informal como reviews.
- **Redes Neurais Recorrentes (RNN/LSTM):** Tentou-se uma arquitetura recorrente (LSTM) para processar as sequências de texto. Embora RNNs captem a ordem, notou-se grande custo de treinamento e dificuldade em competir com modelos pré-treinados; a acurácia estagnou na faixa de 85-88% mesmo após vários ajustes. Assim, as redes LSTM/RNN puras foram **abandonadas** nesta fase devido ao

baixo desempenho e alto custo computacional, reforçando a necessidade de técnicas mais avançadas.

- **SVM + *Embeddings* Universais:** Implementou-se então um pipeline híbrido de **representação vetorial + classificação**. As sentenças foram embutidas usando modelos pré-treinados de sentença, e em seguida alimentou-se um classificador simples. Duas variantes de *embeddings* semânticos foram testadas: (i) **Universal Sentence Encoder (USE)** – um modelo do Google que gera vetores de 512 dimensões para sentenças em diversos idiomas – e (ii) **Sentence-BERT (SBERT)** ([Reimers, N. and Gurevych, I. \(2019\) Sentence-BERT Sentence ...](#)) – que produz *embeddings* de sentenças usando uma rede Siamese BERT fine-tunada para similaridade semântica. Ambos fornecem representações capazes de capturar o sentido global da avaliação. Com os vetores gerados (fixos, sem precisar treinamento pesado), treinou-se um **SVM linear** para distinguir as classes. Essa abordagem mostrou-se **simples e eficiente**, obtendo desempenho surpreendentemente alto dado sua baixa complexidade: acurácia em torno de 92-93%, conforme será detalhado. Em particular, **SVM + SBERT** destacou-se, rivalizando com alguns Transformers mais complexos. Essa técnica híbrida é atrativa para cenários com restrição de recursos, evitando o custo de treinar uma rede profunda completa.
- **Transformers Pré-treinados (fine-tuning):** O cerne do projeto concentrou-se em modelos transformers de última geração pré-treinados em grandes corpora, posteriormente **fine-tunados** nos dados de avaliações da Olist para classificação binária. Três modelos foram experimentados:
 - **BERT** base em português (**BERTimbau**) ([Fábio Souza - dblp](#)): Modelo de 12 camadas e ~110M parâmetros, pré-treinado em corpora extensos do português brasileiro (e.g. BrWaC). Espera-se que por ter aprendido características específicas do idioma, ofereça vantagem na nossa tarefa. Foi carregado o checkpoint público `neuralmind/bert-base-portuguese-cased` via Hugging Face.
 - **RoBERTa** (XLM-RoBERTa base) ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)) ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)): Variante robusta e **multilíngue** do BERT, pré-treinada em 100 idiomas (incluindo português). Usa melhorias de treinamento (mais dados, sem next-sentence prediction, etc.) apresentadas por Liu *et al.*. Utilizamos o modelo `xlm-roberta-base` (~125M parâmetros). Apesar de não

ser específico de português, sua abrangência multilíngue poderia capturar nuances não vistas em corpora monolíngues.

- **DistilBERT** multilingue ([DistilBERT, a distilled version of BERT: smaller, faster, cheaper and ...](#)) ([DistilBERT, a distilled version of BERT: smaller, faster, cheaper and ...](#)): Uma versão enxuta do BERT (cerca de 66M parâmetros) obtida via *distillation* do modelo base multilíngue. O objetivo foi testar se um modelo bem menor conseguiria desempenho próximo, trazendo ganhos em tempo de inferência. Usamos `distilbert-base-multilingual-cased`.

Cada modelo foi fine-tunado adicionando-se uma camada densa de classificação binária no topo do transformer. O treinamento ocorreu em ambiente com GPU, usando **TensorFlow** e a API *Transformers* da Hugging Face

([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)). Configurações consistentes foram adotadas: taxa de aprendizado de $\sim 3e-5$ com otimizador Adam com decaimento de peso, *batch size* de 32 e até 3 epochs (com *early stopping* baseado na acurácia de validação). A **função de perda** (cross-entropy) incorporou os pesos de classe mencionados para compensar o desequilíbrio. Durante o fine-tuning, monitorou-se a *accuracy* e o *F1-score* em validação a cada época, e escolheu-se o modelo final de cada arquitetura com melhor *F1* validatório.

- **LLMs Zero-Shot**: Por fim, investigou-se a capacidade de modelos de linguagem gigantes (**Large Language Models**) realizarem a tarefa **sem qualquer treinamento específico**, apenas interpretando a instrução/prompt adequadamente – abordagem conhecida como *zero-shot classification*. Avaliamos duas opções via APIs:
 - **OpenAI GPT-4** ([\(PDF\) GPT-4 Technical Report - ResearchGate](#))
 - modelo generativo de última geração (estimado >170B parâmetros) com compreensão de múltiplos idiomas. Foi utilizada a *GPT-4 API* com um *prompt* do tipo: “*Classifique o sentimento (Positivo/Negativo) do seguinte comentário: ...*”, seguido do texto da avaliação.
 - **Google Gemini 1.5** - modelo multimodal da Google DeepMind, também de grande porte, cujo acesso prévio foi obtido via programa experimental da API Google AI ([Introducing Gemini 1.5, Google’s next-generation AI model](#)) ([Introducing Gemini 1.5, Google’s next-generation AI model](#)). O *prompt* similar foi usado para obter a classificação.

Os LLMs foram testados em um subconjunto limitado de avaliações (10 exemplos equilibrados) por questões de custo e tempo. Ainda que essa avaliação seja restrita, permite observar se tais modelos

“generalistas” já dominam a tarefa de sentimento em português sem ajuste. Ressalta-se que, por serem usados como caixas-pretas via API, não há controle sobre seu processo; eles geram respostas textuais que foram interpretadas e convertidas em rótulos de classe.

Em resumo, a **Figura 1** ilustra a diversidade de abordagens exploradas no projeto – desde modelos leves e métodos clássicos até arquiteturas complexas e serviços de IA generativa –, compondo um panorama comparativo abrangente.

Experimentos e Resultados

A. Avaliação Quantitativa dos Modelos Supervisionados

Os modelos treinados supervisionados (SVM com *embeddings* e Transformers fine-tunados) foram avaliados no conjunto de **teste** reservado, utilizando métricas padrão de classificação binária: **Acurácia**, **Precisão**, **Recall** e **F1-score**. Dado o balanceamento aproximado das classes no teste, a acurácia é informativa, mas o *F1-score* (média harmônica de precisão e recall) foi adotado como métrica principal por refletir melhor o desempenho em cada classe sem viés de prevalência.

A **Tabela I** resume os resultados obtidos pelos principais modelos. Observa-se que todos os métodos baseados em *Transformers* superaram 93% de acurácia, indicando a eficácia dessas representações profundas de linguagem. O melhor desempenho foi alcançado pelo **BERTimbau fine-tunado**, com **94,26%** de acurácia e **F1 = 94,37%**, ligeiramente superior ao RoBERTa (93,94% F1) e DistilBERT (93,37% F1). A margem entre BERT e RoBERTa foi pequena (~0,4 ponto percentual), sugerindo que ambos conseguiram aprender de forma semelhante os padrões do conjunto de dados – possivelmente o BERT monolíngue teve vantagem marginal por estar totalmente ajustado à distribuição do português coloquial presente nas avaliações.

Notavelmente, as abordagens híbridas com **SVM + *embeddings*** semânticos obtiveram resultados próximos aos dos Transformers completos. Com *Sentence-BERT* como gerador de embeddings, o SVM atingiu **92,88%** de acurácia (F1 = 93,01%), apenas ~1,3 ponto de F1 abaixo do BERT completo. Já usando o Universal Sentence Encoder, chegou a 92,59% (F1 = 92,72%). Essa pequena diferença mostra que embeddings de alta qualidade podem suprir boa parte da necessidade de um modelo supervisionado complexo. Em termos práticos, a inferência via SVM+embeddings é muito mais rápida e leve, o que faz desse método uma opção interessante quando recursos computacionais ou tempo são limitantes, com um sacrifício mínimo de acurácia.

Tabela I - Desempenho de Modelos Supervisionados (Teste)

Modelo	Acurácia (%)	F1-Score (%)
BERT (BERTimbau fine-tuning)	94,26	94,37
RoBERTa (XLM-R fine-tuning)	93,83	93,94
DistilBERT (multilingual)	93,23	93,37
SVM + Sentence-BERT	92,88	93,01
SVM + USE	92,59	92,72

Para aprofundar a avaliação, foram geradas **matrizes de confusão** para cada modelo (não incluídas aqui por brevidade). Essas matrizes confirmaram que o modelo BERT errou muito pouco em ambas as classes, com leve predominância de falsos negativos (algumas poucas avaliações negativas classificadas como positivas). Isso se explica pelo desbalanceamento original: apesar da correção via pesos, ainda havia mais dados de avaliações positivas, tornando o modelo ligeiramente mais “otimista”. Mesmo assim, tanto precisões quanto *recalls* por classe ficaram acima de 93% no BERT, indicando alta confiabilidade em identificar corretamente tanto elogios quanto reclamações. Os outros transformers tiveram padrões similares de erro. Já os modelos SVM apresentaram um número um pouco maior de falsos negativos, refletido no *recall* da classe negativa ligeiramente inferior – ou seja, tendiam a confundir algumas avaliações negativas como positivas –, possivelmente por limites do espaço vetorial fixo dos embeddings em capturar certas sutilezas do texto. Contudo, no geral os resultados de todos os modelos modernos mostraram-se excelentes para a tarefa.

B. Avaliação de Modelos Generativos Zero-Shot

Para efeito comparativo, testamos também **LLMs de propósito geral** (GPT-4 e Google Gemini) na tarefa de classificar sentimento sem nenhum treinamento nos dados da Olist. Foi fornecido a esses modelos apenas o *prompt* descritivo da tarefa junto a exemplos de avaliações, e eles retornaram um rótulo previsto. A **Tabela II** apresenta o desempenho *zero-shot* destes modelos em um pequeno conjunto de 10 avaliações de teste.

Tabela II - Desempenho de LLMs em Classificação Zero-shot

Modelo	Acurácia (%)	Acertos	Erros
OpenAI GPT-4	100,0	10	0
Google Gemini 1.5	90,0	9	1

O resultado impressionante foi que o **GPT-4** acertou **100%** das classificações nos 10 casos apresentados, não cometendo nenhum erro de polaridade. O **Gemini 1.5** também teve ótimo desempenho, acertando 9 de 10 (90%). Embora o tamanho da amostra seja pequeno, esses números dão uma indicação clara da **capacidade desses modelos de ponta em realizar análise de sentimento apenas pela compreensão contextual**, mesmo sem treino supervisionado específico. O GPT-4, em particular, demonstrou entendimento perfeito das nuances de linguagem nas avaliações testadas – por exemplo, reconhecendo ironias sutis ou linguagem sarcástica que poderiam confundir modelos menos sofisticados. Isso reflete os avanços recentes em modelos de grande porte, capazes de realizar *in-context learning*, isto é, adaptar-se a uma tarefa a partir apenas de instruções e exemplos fornecidos na entrada ([v2_06_abr_11_l2_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)) ([v2_06_abr_11_l2_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)).

Entretanto, é importante relativizar esses resultados: LLMs como GPT-4 alcançam tal desempenho *one-shot* a um **custo computacional** muito elevado e com dependência de acesso à API proprietária, o que pode ser inviável em larga escala. No cenário de uso prático em milhares de avaliações diárias, invocar um GPT-4 para cada texto teria latência e custo financeiro significativos. Além disso, modelos generativos não garantem 100% de acerto em geral – nossa amostra foi pequena, e um teste mais amplo poderia revelar algumas falhas, sobretudo em textos fora do comum ou que envolvam gírias muito regionais. De todo modo, a experiência *zero-shot* confirma que a tarefa de sentimento é efetivamente resolvida pelos conhecimentos linguísticos que esses LLMs já possuem, sem precisar de ajuste fino.



Desenvolvimento da Aplicação Web

Com o intuito de disponibilizar o melhor modelo treinado de forma acessível e interativa, foi desenvolvida uma **aplicação web** chamada *NLP-Sentinel Sentiment Analyzer*. A aplicação foi implementada utilizando a biblioteca **Gradio** para criação de interfaces web amigáveis com Python, e o modelo de linguagem **BERTimbau fine-tunado** foi integrado para fazer as previsões de sentimento em tempo real.

([image](#)) *Figura 1. Interface da aplicação web de análise de sentimento desenvolvida, mostrando um exemplo de avaliação negativa inserida pelo usuário e a saída do modelo BERT indicando “NEGATIVO” com alta confiança (99,4%).*

A **Figura 1** ilustra a interface da aplicação. A estética escura e limpa apresenta no topo o título “*Análise de Sentimento em Avaliações de Produtos*” e, logo abaixo, um campo de texto para o usuário digitar (ou colar) uma avaliação de produto em português. Ao lado do campo de entrada, há uma caixa de saída intitulada “*Resultados da Análise*”, onde o sistema exibirá a predição do sentimento após a análise. O usuário interage através de dois botões: “**Analisar Sentimento**”, que aciona o modelo para classificar o texto inserido, e “**Limpar**”, que apaga os campos para permitir uma nova consulta.

A aplicação fornece também convenientes **exemplos de avaliações** logo abaixo dos botões – uma lista de botões pré-preenchidos com frases típicas de avaliações positivas e negativas. Isso permite que o usuário rapidamente teste o sistema clicando em exemplos como “*Produto excelente! Entrega rápida e qualidade ótima. Recomendo!*” (claramente positivo) ou “*Produto muito ruim. Não funciona!*” (negativo), sem precisar digitar manualmente. Esses exemplos cobrem uma gama de cenários de satisfação/insatisfação para demonstrar o comportamento do modelo.

Ao submeter uma avaliação para análise, o backend da aplicação realiza o seguinte fluxo: o texto é **tokenizado** usando o tokenizador do BERT português (*neuralmind/bert-base-portuguese-cased*) – convertendo as palavras em *subword tokens* numéricos –, os tokens são alimentados no modelo BERT fine-tunado (*layers2024/bert-sentiment* hospedado no Hugging Face Hub), e o modelo produz uma **distribuição de probabilidades** sobre as classes (positiva vs. negativa) via sua camada *softmax*. A classe com maior probabilidade é selecionada como predição. Em seguida, calcula-se a **confiança** associada (probabilidade normalizada da classe escolhida). Esses resultados são então exibidos na interface: o sistema mostra o rótulo predito (**POSITIVO** ou **NEGATIVO**), juntamente com um indicador textual de confiança. Implementou-se uma lógica de *threshold*: se a confiança estiver acima de 90%, o resultado é marcado como “Alta confiança” com um símbolo ; caso contrário, aparece como “Possivelmente [positivo/negativo]” com um símbolo de alerta , sinalizando que a classificação não foi tão segura.

Por exemplo, na Figura 1, o usuário digitou a frase “*horrível. ruim demais, nunca mais eu compro*”, claramente uma avaliação negativa. Ao clicar em “**Analisar Sentimento**”, o modelo retorna “NEGATIVO” com **99,4%** de confiança, indicando altíssima certeza, e a interface marca como “Alta

confiança”. Essa transparência é importante em aplicações de IA para que o usuário saiba distinguir resultados certos de casos limítrofes.

A seção “*Sobre o Projeto*” exibida na figura é um painel expansível que apresenta informações técnicas para usuários interessados: descreve que o modelo usado é o **BERT fine-tunado para análise de sentimentos** em português (com link para o repositório no Hugging Face ([app.py](#))), menciona o dataset Olist e credita o autor do projeto.

Em termos de desempenho, a aplicação se mostrou bastante **responsiva**. Graças ao uso do DistilBERT no front (caso implementado) ou otimizações do BERT via tensorflow-serving, cada inferência leva em torno de poucos **centésimos de segundo**, tornando a experiência praticamente em tempo real. O modelo carrega inicialmente ~400MB em memória, mas após carregado, as predições são rápidas. A opção de hospedar a aplicação em um **Hugging Face Space** também foi explorada, aproveitando a infraestrutura gratuita para demonstrações online, conforme mencionado no README do projeto.

Em suma, a aplicação web desenvolvida comprova a viabilidade de levar o modelo de **estado-da-arte** para um cenário de uso real, onde qualquer interessado pode testar a classificação de sentimentos em texto livre. Esse produto final integra os conhecimentos adquiridos durante o projeto e serve tanto como ferramenta prática quanto como demonstração visual dos resultados alcançados.

Discussão

Os resultados obtidos neste trabalho fornecem uma série de **insights importantes** tanto do ponto de vista de desempenho dos modelos quanto de considerações práticas para implantação de soluções de PLN em português:

Desempenho de Modelos Avançados: Confirmou-se que os modelos transformers pré-treinados, quando fine-tunados com um conjunto robusto de dados anotados, **alcançam desempenho superior** para análise de sentimentos. O BERTimbau atingiu *F1-score* acima de 94%, estabelecendo um novo patamar para a tarefa nos dados da Olist. Esse resultado se alinha ao que a literatura reporta para BERT em outras línguas – modelos bem pré-treinados capturam intrinsecamente as relações semânticas e sintáticas do texto ([1810.04805] [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#)), necessitando de relativamente poucos ajustes para tarefas específicas. Notou-se que mesmo um modelo multilingue genérico (XLM-RoBERTa) teve performance quase equivalente

(93,9% F1), o que indica que o conhecimento das estruturas linguísticas do português já está amplamente presente nesses modelos de base ampla. O DistilBERT, apesar de menor, conseguiu chegar próximo (93,3% F1), o que é notável dada sua leveza – na prática, essa pequena perda de ~1 ponto percentual poderia ser aceitável em troca de velocidade 60% maior, dependendo do caso de uso ([DistilBERT, a distilled version of BERT: smaller, faster, cheaper and ...](#)).

Abordagens Híbridas e Simplicidade: Um dos achados mais relevantes foi o **forte desempenho de métodos híbridos simples**. A combinação de *embeddings* de sentença de alta qualidade (como SBERT) com um classificador linear provou ser **altamente competitiva**, atingindo >93% de F1. Essa solução não requer treinamento pesado (apenas treinar um SVM, o que leva segundos) e é extremamente econômica em recursos – os embeddings SBERT podem ser pré-calculados ou obtidos sob demanda via um modelo menor que BERT. Esse resultado reforça que, para muitas aplicações práticas, pode não ser estritamente necessário treinar um modelo complexo end-to-end se dispomos de bons embeddings pré-treinados ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)). Em cenários com restrição de memória ou ausência de GPU, um modelo SVM+embeddings poderia rodar em CPU comum com latência baixa, viabilizando uso em larga escala ou em dispositivos com menos poder computacional.

Modelos Tradicionais vs. Modernos: Em contraste, as abordagens tradicionais puramente baseadas em Bag-of-Words ou mesmo redes LSTM não atingiram o patamar desejado. Esses métodos foram **descartados** ao longo do projeto justamente por apresentarem **menor desempenho e maior custo de treinamento** (no caso de LSTM) ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)) ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto \(2\).py](#)). Essa decisão ilustra um ponto metodológico importante: **simplificar e focar no que funciona**. Ao perceber que BoW e LSTM não ofereceriam resultados melhores que opções modernas, o esforço foi redirecionado para técnicas mais promissoras (Transformers e embeddings). Houve, portanto, um **aprendizado incremental** – testar hipóteses mais simples inicialmente, mas saber **pivotar** para abordagens mais poderosas quando as limitações ficam claras. Esta análise criteriosa do *retorno técnico* de cada técnica evitou desperdiçar tempo em otimizações infrutíferas e permitiu concentrar o projeto no estado da arte.

LLMs e Zero-shot: A incursão por modelos generativos de larga escala como GPT-4 trouxe reflexões sobre **até onde esses modelos generalistas**

já conseguem ir em tarefas específicas. O GPT-4 mostrou-se extremamente proficiente em detectar o sentimento, mesmo sem treino, graças ao seu amplo conhecimento linguístico adquirido durante o pré-treinamento ([v2_06_abr_11_l2_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto\(2\).py](#)). Isso sinaliza que, para aplicações futuras, se serviços como GPT-4 ou Google Gemini estiverem facilmente disponíveis e com custo acessível, pode-se considerar utilizar diretamente tais APIs para análise de sentimento, **eliminando a necessidade de construir um modelo próprio** – especialmente em cenários onde a precisão máxima é crítica e o volume de dados não é massivo. No entanto, questões de **custo, privacidade e dependência externa** surgem: enviar textos confidenciais de clientes para uma API de terceiro nem sempre é aceitável, e pagar por milhares de requisições diárias pode ser oneroso. Portanto, há um **trade-off** entre a comodidade/poder dos LLMs e a autonomia fornecida por um modelo custom (como BERT) rodando localmente. Neste projeto optamos por treinar um modelo próprio pelas razões acima, mas os resultados dos LLMs confirmam que eles são uma referência de desempenho máximo – saber que nosso BERT alcançou ~94% em dados reais enquanto o GPT-4 fez 100% em exemplos limitados mostra que ficamos **próximos do teto** que um sistema pode atingir nessa tarefa, dado que mesmo humanos podem discordar em alguns casos ambíguos.

Importância do Pré-processamento e Dados: Destaca-se também o **impacto do pré-processamento e do balanceamento de classes** no sucesso do projeto. O cuidado em limpar textos (remover ruídos, normalizar) e em tratar do desequilíbrio (via pesos de classe) teve influência direta no desempenho dos modelos ([v2_06_abr_11_l2_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto\(2\).py](#)). Em experimentos iniciais sem esses ajustes, observou-se métricas inferiores. Por exemplo, antes de aplicar peso de classe, o BERT tendia a ter ~5 pontos percentuais a menos de *recall* em avaliações negativas. Com o balanceamento, esse gap foi praticamente eliminado. Isso enfatiza que uma **boa curadoria dos dados** pode às vezes melhorar tanto o resultado quanto trocar de modelo. Da mesma forma, a decisão de excluir as avaliações neutras (3 estrelas) do escopo evitou confundir o modelo com exemplos de difícil julgamento, concentrando o aprendizado nos polos claros. Um possível trabalho futuro seria incorporar também a classe neutra e verificar se modelos como BERT conseguem identificar essa categoria intermediária – espera-se que sim, mas com performance menor, pois “neutralidade” é mais sutil que positividade ou negatividade explícitas.

Deploy e Manutenibilidade: No desenvolvimento do produto final, aprendeu-se que aspectos como **facilidade de deploy, versionamento e acessibilidade do modelo** também são essenciais. A opção por **publicar o**

modelo fine-tunado no Hugging Face Hub se mostrou vantajosa para organizar o trabalho e permitir reuso ([v2_06_abr_11_12_optimization_pos_deep_learning_projeto_nlp_sentinel_projeto\(2\).py](#)). Isso tornou simples integrar o modelo à aplicação web (bastando carregá-lo do hub) e possibilita que outros possam baixá-lo ou testá-lo, fomentando colaboração. Além disso, a construção da aplicação em Gradio evidenciou que hoje existe um ecossistema maduro para levar modelos de NLP do laboratório para produção com pouco esforço adicional – diferentemente de alguns anos atrás, quando criar uma interface custom era trabalhoso. Em suma, as decisões de engenharia adotadas visaram **simplicidade e eficiência**, alinhadas aos objetivos do projeto.

Em geral, os experimentos confirmaram que **não há uma solução única que seja ótima em todos os critérios**. O BERT fine-tunado foi o campeão em acurácia, mas requer GPU para treinar e alguma otimização para servir. O SVM+SBERT foi levemente inferior em acurácia, porém trivial de treinar e leve de rodar. O GPT-4 acertou tudo, mas depende de fatores externos. Assim, a escolha do modelo ideal deve considerar o **contexto de aplicação**: se a prioridade for máxima precisão e houver infraestrutura disponível, um Transformer fine-tunado é indicado; se for necessário algo rápido, simples e que rode em qualquer lugar, um modelo baseado em embeddings pré-calculados pode ser preferível; se a escala for pequena e for aceitável usar um serviço externo, um LLM via API pode valer a pena. Essa compreensão mais abrangente é um dos principais aprendizados do projeto.

Conclusão

Este trabalho apresentou uma investigação aprofundada de técnicas de Deep Learning aplicadas à classificação de sentimentos em avaliações de produtos em língua portuguesa. Por meio do projeto **NLP-Sentinel**, experimentamos desde abordagens tradicionais até modelos de linguagem de última geração, obtendo **alto desempenho** na tarefa e desenvolvendo uma solução completa – do treinamento à disponibilização via aplicação web.

Em termos quantitativos, alcançamos **94% de acurácia** na identificação de polaridade positiva/negativa, usando um modelo BERT pré-treinado em português e posteriormente ajustado ao nosso conjunto de dados específico. Este resultado demonstra o poder dos modelos **Transformer** modernos em capturar nuances da linguagem natural e fornece evidências de que é possível aplicar tais modelos com sucesso em **PLN para o português**, historicamente um idioma menos atendido por soluções prontas do que o inglês. A título de comparação, mesmo modelos muito mais simples, como um SVM alimentado por embeddings de sentença, aproximaram-se desse

desempenho, o que reforça a robustez das representações semânticas utilizadas.

Qualitativamente, discutimos diversas considerações importantes para projetos de aprendizado de máquina no mundo real: a necessidade de **balancear desempenho e recursos**, a importância de **pré-processamento** e tratamento adequado dos dados, e os **trade-offs** entre manter um modelo próprio ou usar um serviço de terceiros. Cada técnica avaliada trouxe lições – por exemplo, modelos *híbridos* mostraram-se válidos vs. modelos end-to-end, e modelos gigantes como GPT-4 evidenciaram um teto de performance que os modelos menores buscam alcançar.

Como trabalhos futuros, várias direções podem ser exploradas. Uma extensão natural seria **incluir a classificação de sentimento “neutra”** nas previsões, desafiando o modelo a identificar textos ambíguos ou mistos. Outra possibilidade é experimentar técnicas de *data augmentation* para gerar mais exemplos de treinamento, especialmente de classe minoritária, visando ainda mais robustez. Do ponto de vista de modelos, poderíamos avaliar arquiteturas transformers mais novas ou específicas, como o **RoBERTa-large** ou modelos recentemente lançados para português, e também testar métodos de compressão do modelo (quantização, poda) para reduzir latência sem perder muita acurácia – algo valioso para deploy mobile, por exemplo. Além disso, integrar um módulo de **explicação de decisões** (via LIME ou SHAP) na aplicação web seria interessante para tornar o sistema mais interpretável, destacando quais palavras de uma avaliação influenciaram na classificação de sentimento.

Em resumo, o projeto atingiu seus objetivos ao comparar e identificar as melhores técnicas de análise de sentimento para o cenário proposto e produzir um artefato utilizável. Os resultados evidenciam que **altas taxas de acerto (>90%) são atingíveis** mesmo em língua portuguesa e em domínio de e-commerce, desde que se utilizem modelos de linguagem robustos e se tenha atenção aos detalhes do processo de aprendizado. Espera-se que este estudo e a solução desenvolvida sirvam de base para aplicações reais de monitoramento de opinião de consumidores, bem como inspiração para trabalhos acadêmicos futuros que busquem aliar **desempenho de ponta e aplicabilidade prática** em processamento de linguagem natural.

Agradecimentos

Agradeço ao professor **Cléber Zanchettin** pela orientação durante a disciplina e pelo incentivo na realização deste projeto. Também expresso gratidão aos colegas do **CIn-UFPE** pelas discussões frutíferas que ajudaram

a moldar as ideias aqui apresentadas. A comunidade open-source (Hugging Face, Gradio, etc.) merece reconhecimento por fornecer ferramentas essenciais para o desenvolvimento rápido deste trabalho. Por fim, agradeço à **Olist** e ao Kaggle por disponibilizarem o conjunto de dados público que viabilizou os experimentos.

Referências

- [1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, **“BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,”** *Proc. of NAACL-HLT*, pp. 4171–4186, 2019. DOI: 10.48550/arXiv.1810.04805.
- [2] F. Souza, R. Nogueira, and R. Lotufo, **“BERTimbau: Pretrained BERT Models for Brazilian Portuguese,”** *Proc. 9th Brazilian Conf. on Intelligent Systems (BRACIS)*, Springer, pp. 403–417, 2020.
- [3] Y. Liu *et al.*, **“RoBERTa: A Robustly Optimized BERT Pretraining Approach,”** arXiv preprint arXiv:1907.11692, 2019.
- [4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, **“DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,”** *Proc. of NeurIPS Workshop on Energy Efficient ML (EMC²)*, 2019. DOI: 10.48550/arXiv.1910.01108.
- [5] N. Reimers and I. Gurevych, **“Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,”** *Proc. of EMNLP/IJCNLP*, pp. 3980–3990, 2019. DOI: 10.48550/arXiv.1908.10084.
- [6] D. Cer *et al.*, **“Universal Sentence Encoder,”** *Proc. of EMNLP Workshop on RepEval*, 2018. DOI: 10.48550/arXiv.1803.11175.
- [7] OpenAI, **“GPT-4 Technical Report,”** arXiv preprint arXiv:2303.08774, 2023.
- [8] S. Pichai and D. Hassabis, **“Our next-generation model: Gemini 1.5,”** *Google AI Blog*, Feb. 2024. [Online]. Available: <https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/>
- [9] Olist, **“Brazilian E-commerce Public Dataset by Olist,”** *Kaggle*, 2018. [Online]. Available: <https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce/data>