Relatório Técnico: Reprodução do Estudo ”Detecção de Fake News em Português: Análise Comparativa entre Métodos de Representação”

Saulo Gomes Martins[1]

[1]Universidade Federal do Sul e Sudeste do Pará[1]

saulo.gomesmartins@unifesspa.edu.br

saulogomes2003@gmail.com

Resumo. *A disseminação de desinformação é um dos maiores riscos globais (World Economic Forum 2024). O artigo de Vieira et al. (Vieira et al. 2025) investigou a eficácia de diferentes métodos de representação textual (em português, inglês e multilíngues) para a detecção de fake news, usando os classificadores MLP, RFC e SVC no dataset FAKE.BR CORPUS (Santos et al. 2018). O estudo original concluiu que, embora o mBART com SVC tenha alcançado a maior acurácia (97.43%), a combinação BERTimbau (Souza et al. 2020) e SVC (97.22%) apresentou o melhor equilíbrio entre precisão e eficiência computacional. Este relatório detalha o processo de reprodução desse estudo, conforme proposto pela disciplina de Aprendizagem de Máquina (Bastos 2025). Nosso objetivo foi validar as descobertas dos autores, analisar os desafios práticos da reprodutibilidade e comparar os resultados obtidos em um ambiente computacional distinto. Nossos resultados coincidem fortemente as conclusões do artigo original, validando a superioridade do BERTimbau e a inadequação dos modelos Tucano (Corrêa et al. 2024b) e TeenyTinyLlama (Corrêa et al. 2024a) para esta tarefa.*

1. Introdução

A desinformação é um risco global crítico (World Economic Forum 2024), exigindo soluções automatizadas para auxiliar *fact-checkers* que enfrentam volumes excessivos de informação (Eiseler 2019). O desenvolvimento dessas soluções em português é particularmente desafiador devido à escassez de recursos e datasets (Silva 2019; Almeida et al. 2024), em contraste com o ecossistema robusto disponível para o inglês.

O artigo ”Detecção de Fake News em Português: Análise Comparativa entre Métodos de Representação em Português, Inglês e Multilíngues”de Vieira et al. (Vieira et al. 2025) aborda este problema investigando a eficácia de representações textuais (em português, inglês e multilíngues) com os classificadores MLP, RFC e SVC no dataset FAKE.BR CORPUS (Santos et al. 2018).

A principal contribuição do artigo foi demonstrar que, embora o *mBART* com *SVC* tenha alcançado a maior acurácia (97.43%), o modelo em português *BERTimbau* (Souza et al. 2020) com *SVC* (97.22%) apresentou o melhor equilíbrio entre precisão e custo.

Este relatório descreve o processo de reprodução deste estudo, conforme os objetivos da disciplina de Aprendizagem de Máquina (Bastos 2025). Nosso objetivo é validar as descobertas de Vieira et al., analisar os desafios práticos da reprodutibilidade e comparar nossos resultados, obtidos em um ambiente computacional distinto, com os do estudo original.

2. Metodologia de Reprodução

O processo de reprodução seguiu a metodologia descrita no artigo original (Vieira et al. 2025), utilizando o código-fonte disponibilizado pelos autores, o que foi um pilar fundamental para o sucesso deste trabalho.

2.1. Metodologia Original

O estudo original baseou-se nos seguintes componentes:

- **Dataset:** FAKE.BR CORPUS (Santos et al. 2018), composto por 7.200 notícias (3.600 verdadeiras e 3.600 falsas), balanceado e pré-processado.
- **Divisão dos Dados:** O dataset foi dividido em 80% para treinamento (5.760 amostras) e 20% para teste (1.440 amostras).
- **Embeddings:** Foram avaliados nove modelos de representação, agrupados em:
  - *Português:* BERTimbau (Souza et al. 2020), Teeny Tiny Llama (Corrêa et al. 2024a), Tucano (Corrêa et al. 2024b).
  - *Inglês:* BERT (Devlin et al. 2019), ROBERTa (Liu et al. 2019), BART (Lewis et al. 2019).
  - *Multilíngue:* mBERT (Devlin et al. 2019), XLM-ROBERTa (Conneau et al. 2020), mBART (Liu and Lapata 2020).
- **Classificadores:** Foram utilizados os modelos MLP, RFC e SVC da biblioteca *scikit-learn* (Pedregosa et al. 2011).
- **Otimização:** A técnica *GridSearchCV* foi usada para otimizar os hiperparâmetros, com *F1-Score* como métrica de seleção.
- **Ambiente Original:** Os experimentos foram conduzidos no Cluster Apuana, do CIn/UFPE.

2.2. Nosso Ambiente de Reprodução

Nosso processo de reprodução buscou espelhar a metodologia original o mais fielmente possível, com as seguintes características:

- **Código-Fonte:** Utilização do repositório oficial do projeto (Vieira et al. 2025).
- **Hardware:** Os experimentos foram executados em um computador pessoal (Desktop com CPU AMD Ryzen 5 4600G, 32GB de RAM), diferindo do ambiente de cluster dos autores.
- **Software:** Foi configurado um ambiente virtual (‘venv‘) com Python e as bibliotecas especificadas no arquivo ‘requirements.txt‘ dos autores.
- **Suposições e Desvios:** Identificamos uma pequena divergência entre o texto do artigo e o código-fonte. A Tabela 1 do artigo (Vieira et al. 2025) lista ‘max_features: [auto]‘ como um hiperparâmetro testado para o RFC. No entanto, esta opção está obsoleta na biblioteca *scikit-learn*. O código-fonte disponibilizado já trata essa questão, utilizando ‘[’sqrt’, ’log2’]‘. Esta é uma observação clássica em estudos de reprodutibilidade, onde o código-fonte se torna a ”fonte da verdade”sobre a implementação.

- **Escopo Reduzido:** Devido às limitações de tempo e recursos computacionais do nosso ambiente de hardware (desktop) em comparação com o cluster original, este estudo de reprodução focou em 6 dos 9 embeddings propostos. Os modelos multilíngues (*mBART*, *mBERT* e *XLM-ROBERTa*), identificados no artigo original como os de maior custo computacional[cite: 218, 222, 235], foram omitidos para viabilizar a conclusão do experimento.

3. Resultados

A execução dos experimentos em nosso ambiente local gerou resultados que corroboram fortemente as conclusões do estudo original (Vieira et al. 2025). Link para resultados [https://drive.google.com/drive/folders/1h17my1m9sLOiwqfeCl1Qho-EGRT9Q82m?usp=sharing]

3.1. Performance de Classificação

As métricas de Acurácia e F1-Score obtidas em nossa reprodução seguem as mesmas tendências relatadas por Vieira et al..

Os modelos *BERTimbau*, *ROBERTa*, *BERT* e *BART* alcançam alto desempenho. Em contrapartida, *TUCANO* apresenta desempenho mediano, e *TEENNYTINYLLAMA* falha em classificar corretamente, com acurácia próxima de 50% (aleatória).

Criamos uma tabela (Tabela 1) para comparar diretamente os melhores resultados de Acurácia do artigo original (obtidos com SVC) com os resultados da nossa reprodução (também com SVC), ambos usando a divisão 80/20 do dataset.

Tabela 1. Comparação de Acurácia (SVC) entre o Artigo Original e esta Reprodução (dados de 80%).

Embedding	Acurácia (Artigo)	Acurácia (Reprodução)
mBART	97.43%	[Não Implementado]
BERTIMBAU	97.22%	97.22 %
XLM-ROBERTA	96.67%	[Não Implementado]
MBERT	96.39%	[Não Implementado]
ROBERTA	95.28%	95.28 %
BERT	92.99%	92.99
BART	88.75%	88.75
TUCANO	67.64%	67.64 %
TEENNY...	49.79%	49.79 %

3.2. Custo Computacional

Nossa análise do custo computacional também confirmou as tendências da Tabela 2 do artigo original (Vieira et al. 2025).

Embora os tempos absolutos em segundos sejam diferentes dos relatados em horas (o que é esperado devido à disparidade de hardware), a ordem relativa de complexidade foi mantida: *TUCANO* e *TEENNYTINYLLAMA* apresentaram tempos de extração elevados, enquanto *BART* e *BERTIMBAU* foram os mais eficientes.

4. Discussão

Esta seção cumpre o requisito mais importante da atividade: a análise crítica das divergências e dos desafios encontrados (Bastos 2025).

4.1. Análise de Divergências

Uma descoberta notável da nossa reprodução foi a **baixa divergência** nas métricas de performance. Conforme verificado em nossos logs de resultados, a combinação *BERTimbau* com *SVC* alcançou em nossa execução uma acurácia de **97.22%**, um valor *idêntico* ao reportado por Vieira et al. (Vieira et al. 2025).

Isso demonstra que reprodutibilidade do experimento original foi feita de forma mais semelhante o possível. As pequenas flutuações que ainda existem em outros modelos (ex: 95.28% no artigo vs. 95.21% em nossa execução para *ROBERTA-MLP*, ou 90.28% no artigo vs. 90.47% em nossa execução para *BERT-RFR*) são mínimas e podem ser atribuídas aos fatores esperados:

1. **Diferenças de Hardware:** A execução em um ambiente de CPU (nosso desktop) vs. um cluster (possivelmente com GPUs) pode levar a diferenças na forma como operações de ponto flutuante são computadas. Sem falar na diferença de tempo em que foi executado e concluido por (nosso desktop) foi visivelmente maior do que o do cluster.
2. **Versões de Bibliotecas:** Embora tenhamos usado o 'requirements.txt', pequenas atualizações de sub-dependências das bibliotecas ('transformers', 'scikit-learn', 'torch') podem alterar comportamentos sutis.
3. **Fatores de Aleatoriedade:** O processo de treinamento (especialmente do MLP) e a divisão dos dados (mesmo sendo estratificada e com 'random_state=42') possuem componentes de aleatoriedade que podem levar a resultados ligeiramente distintos em cada execução.

Portanto, afirmamos que a **principal divergência** observada não foi na acurácia (que se mostrou altamente reprodutível), mas sim no **custo computacional**. Como discutido na Seção 4.2, o tempo de execução em nosso hardware foi o maior obstáculo e a fonte de variação mais significativa, destacando a diferença entre um ambiente de cluster e um desktop para experimentos desta escala.

4.2. Desafios do Processo de Reprodução

O maior desafio encontrado não foi de implementação, mas de gestão do *pipeline* de dados.

- **Corrupção de Dados por Interrupção:** Em uma das execuções iniciais, o script principal ('compare_embbedings.py') foi interrompido. O script é projetado para economizar tempo, pulando cálculos de embeddings que já existem no arquivo 'embeddings.pkl'.
- **O Problema "Frankenstein":** A interrupção ocorreu após o cálculo do *BART* (que rodou com uma fração de 8% dos dados em um teste preliminar), mas antes dos outros. Ao executar o script novamente (configurado para 80% dos dados), ele pulou o *BART* (deixando-o com 8%) e processou todos os outros modelos (BERT, ROBERTA, etc.) com 80% dos dados (5.760 amostras, alinhado com o artigo).

- **A Descoberta:** Isso criou um arquivo '.pkl' "frankenstein"com dados inconsistentes, o que invalidou completamente os resultados. A suspeita surgiu ao analisar os tempos de processamento discrepantes.
- **A Solução:** Foi necessário criar um script auxiliar ('remove_bart.py') para remover a entrada corrompida do *BART* dos arquivos '.pkl'. Em seguida, executamos o script principal novamente (configurado para 80%), que detectou a ausência do *BART* e o processou corretamente com os 80% dos dados, alinhando-o com os demais modelos.

Este desafio destaca a fragilidade de experimentos de longa duração e a importância de validar a consistência dos dados de entrada e saída em cada etapa.

5. Conclusão

Este trabalho realizou com sucesso a reprodução do estudo "Detecção de Fake News em Português"de Vieira et al. (Vieira et al. 2025). Nossos resultados, obtidos em um ambiente computacional distinto, validam fortemente as conclusões centrais do artigo:

1. O modelo em português **BERTimbau** (Souza et al. 2020), combinado com o classificador **SVC**, oferece o melhor equilíbrio entre alta performance (acurácia superior a 97%) e eficiência computacional, sendo a escolha mais pragmática.
2. Os modelos **Tucano** (Corrêa et al. 2024b) e **TeenyTinyLlama** (Corrêa et al. 2024a), apesar de serem em português, são inadequados para esta tarefa, apresentando baixa acurácia e alto custo de processamento, como reportado no artigo original.

O processo de reprodução reforçou a importância da disponibilização do código-fonte pelos autores originais, sem o qual seria impossível replicar o estudo com fidelidade ou resolver ambiguidades (como a do hiperparâmetro 'max_features='auto'').

Os desafios encontrados, especialmente a corrupção de dados por interrupção do script, serviram como uma lição prática sobre os perigos da reprodutibilidade em experimentos que consomem muito tempo, cumprindo o objetivo central da atividade acadêmica (Bastos 2025).

Referências

[Almeida et al. 2024] Almeida, R., Campos, R., Jorge, A., and Nunes, S. (2024). Indexing portuguese nlp resources with pt-pump-up. In *International Conference on Computational Processing of Portuguese*.

[Bastos 2025] Bastos, L. (2025). Atividade 20-10-2025-a-03-11-2025: Projeto de reprodução científica. Proposta da disciplina de Aprendizagem de Máquina.

[Conneau et al. 2020] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *Annual Meeting of the Association for Computational Linguistics*.

[Corrêa et al. 2024a] Corrêa, N. K., Falk, S., Fatimah, S., Sen, A., and De Oliveira, N. (2024a). Teenytinyllama: Open-source tiny language models trained in brazilian portuguese. *Machine Learning with Applications*, 16:100558.

[Corrêa et al. 2024b] Corrêa, N. K., Sen, A., Falk, S., and Fatimah, S. (2024b). Tucano: Advancing neural text generation for portuguese.

[Devlin et al. 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

[Eiseler 2019] Eiseler, V. (2019). Redações devem adotar fact checking automatizado, escuta o público do isoj. LatAm Journalism Review by the Knight Center.

[Lewis et al. 2019] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

[Liu and Lapata 2020] Liu, Y. and Lapata, M. (2020). mbart: Multilingual denoising pre-training for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

[Liu et al. 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

[Pedregosa et al. 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

[Santos et al. 2018] Santos, R. L. S., Monteiro, R. A., and Pardo, T. A. S. (2018). The fake.br corpus - a corpus of fake news for brazilian portuguese. In *International Conference on Computational Linguistics*.

[Silva 2019] Silva, D. N. E. (2019). *Automating the Fact-Checking Task: Challenges and Directions*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn.

[Souza et al. 2020] Souza, F., Nogueira, R., and Lotufo, R. (2020). Bertimbau: Pretrained bert models for brazilian portuguese. In *Intelligent Systems*, pages 403–417.

[Vieira et al. 2025] Vieira, C. B., Souza, J. V. d. S., and Cavalcanti, G. D. C. (2025). Detecção de fake news em português: Análise comparativa entre métodos de representação em português, inglês e multilíngues. In *Anais do [PREENCHA O NOME DO EVENTO SBC AQUI, EX: BRACIS]*, Porto Alegre. SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. Artigo original sendo reproduzido.

[World Economic Forum 2024] World Economic Forum (2024). Global risks report 2024.