Classificação de *fake news* utilizando Aprendizado de Máquina

Murilo Luis C. Neves[1], **Vitor Padovani**[1], **Fernando Silva Grande**[1]

[1]Departamento de Informática – Universidade Estadual de Maringá

Abstract. *This meta-paper describes the style to be used in articles and short papers for SBC conferences. For papers in English, you should add just an abstract while for the papers in Portuguese, we also ask for an abstract in Portuguese ("resumo"). In both cases, abstracts should not have more than 10 lines and must be in the first page of the paper.*

Resumo. *Este meta-artigo descreve o estilo a ser usado na confecção de artigos e resumos de artigos para publicação nos anais das conferências organizadas pela SBC. É solicitada a escrita de resumo e abstract apenas para os artigos escritos em português. Artigos em inglês deverão apresentar apenas abstract. Nos dois casos, o autor deve tomar cuidado para que o resumo (e o abstract) não ultrapassem 10 linhas cada, sendo que ambos devem estar na primeira página do artigo.*

1. Introdução

Falar um pouco sobre fake news e aprendizado de máquina...

2. Fundamentação Teórica

2.0.1. Random Forest

2.0.2. Logistic Regression

2.0.3. Gradient Boosting

2.0.4. Redes Neurais

3. Materiais e métodos

Nesta seção, são descritos os materiais utilizados para o trabalho (seção 3.1) e os métodos como foram trabalhados (seção 3.2).

3.1. Materiais

Os materiais utilizados para esse trabalho incluem o *dataset* WELFake, e, como ferramentas de codificação principais, as bibliotecas *Scikit-Learn*, *Spacy* e *Torch*.

3.1.1. Base de dados

O *dataset* utilizado é o WELFake, devido ao grande volume de dados presentes no mesmo e um relativo balanceamento entre as classes real e falso. Ele possui 72134 entradas, onde

35028 são reais e 37106 são falsas. No entanto, como é mostrado na seção 3.2.1, há duplicatas, o que torna o número de entradas válidas no *dataset* menor.

O *dataset* possui como colunas o identificador de cada texto, título, texto e *label*, onde a *label* 0 indica notícia falsa e 1 indica notícia real. Um ponto importante de ser notado é que, como o *dataset* não possui *features* pré-extraídas (apenas possui o texto em si), elas foram posteriormente extraídas pelos discentes (seção 3.2.1).

Uma análise inicial indica a seguinte proporção de duplicatas:

Duplicatas em títulos	9786
Duplicatas em textos	9415
Duplicatas em linhas inteiras	8456

Table 1. Quantidade de duplicatas encontradas na base de dados

Para este trabalho, considerou-se que apenas duplicatas tanto em título quanto em textos deveriam ser removidas. Alguns gráficos que mostram a distribuição dos tamanhos dos textos e títulos seguem.

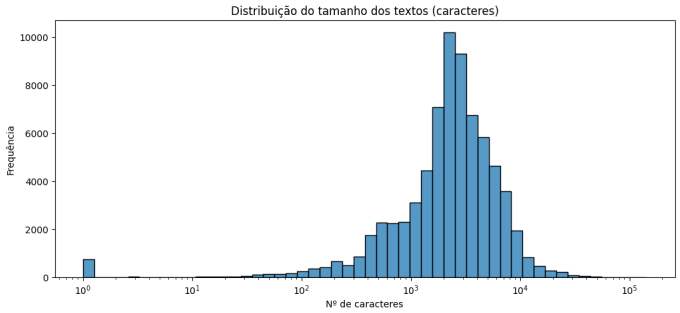

Figure 1. Distribuição dos tamanhos dos textos (em caracteres)

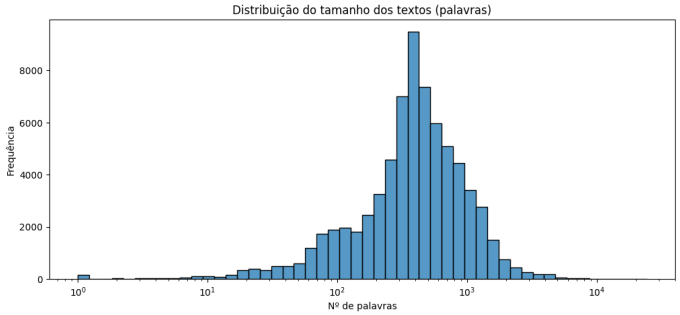

Figure 2. Distribuição dos tamanhos dos textos (em palavras)

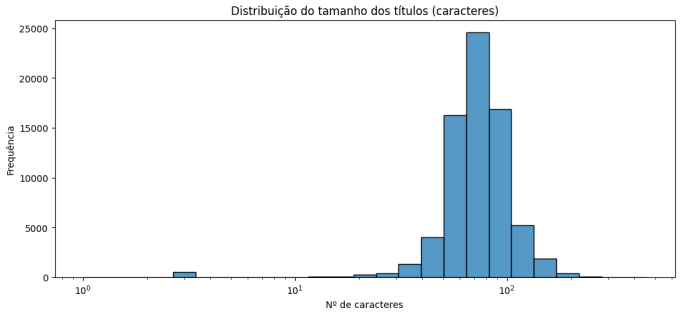

Figure 3. Distribuição dos tamanhos dos títulos (em caracteres)

3.1.2. Scikit-Learn

3.1.3. Spacy

3.1.4. Torch

3.2. Métodos

Nesta seção são descritos os métodos utilizados. Por curiosidade, foram explorados dois métodos de se trabalhar com os dados, nomeados 1 e 2, além de haver um pré-processamento necessário (seção 3.2.1).

3.2.1. Pré-processamento

Nesta etapa, um primeiro passo importante é a retirada das duplicatas, a fim de evitar possíveis *data leaks* para os modelos. Além disso, como haviam entradas onde uma das colunas (ora texto, ora título) era nula, foi considerado como 'texto' das entradas a concatenação de título e texto, separados por um espaço.

Após isso, para que fosse possível a execução do método 1, foi-se necessário definir *features* a serem extraídas dos textos para que, a partir delas, fosse criado um novo *dataset* apenas com as *features extraídas*. Para tanto, foram definidas as *features* dispostas na tabela a seguir.

Polaridade
Subjetividade
Tamanho médio de palavra
Tamanho médio de sentença
Número de exclamações
Proporção de *stopwords*
Diversidade léxica
Quantidade de números presentes no texto
Quantidade de entidades nomeadas
Tamanho da maior entidade nomeada
Proporção de pronomes
Número de links
Proporção de adjetivos
Proporção de advérbios
Proporção de verbos de dúvida/incerteza
Proporção de citações
Pontuação não terminal por sentença
Proporção de *hedges*
Proporção de conjunções coordenativas

Table 2. *Features* extraídas dos textos

Além disso, para o método 2, foram utilizadas como *features* uma composição entre um vetor da frequência TF-IDF (com limite de 5000) e um vetor com os *embeddings* gerados pelo processamento do texto por meio do BERT (MiniLM-L12-V2), que possui 768 dimensões. O tamanho total do vetor de entrada é de 5768 dimensões. No entanto, como não há maneira significativa e intuitiva de se visualizar esse vetor, não foram realizadas visualizações nessa abordagem.

3.2.2. Método 1: Técnicas clássicas

Nesse método, foi-se trabalhado com os indicadores extraídos dos textos conjuntamente a alguns algoritmos classificadores da biblioteca *scikit-learn*. Em um primeiro momento, foram definidos os algoritmos a serem testados e os parâmetros para *GridSearch*.

Algoritmo	Parâmetro	Valores testados
Regressão Logística	C	0.1, 1, 10
Regressão Logística	Penalidade	l1, l2
Regressão Logística	*Solver*	liblinear
Random Forest	Num. estimadores	50, 100
Random Forest	*Max. depth*	5, 10, None
Gradient boosting	Num. estimadores	50, 100
Gradient boosting	Taxa de aprendizado	0.1, 0.05
Gradient boosting	*Max. depth*	3, 5

Table 3. Parâmetros explorados durante o *GridSearch*

Após isso, os modelos foram ranqueados e escolhido o melhor deles. Para este, foi montada a matriz de confusão e feita uma análise de quais *features* foram mais relevantes bem como uma curva de aprendizado para observar possível *overfitting*.

3.2.3. Método 2: Redes neurais

Para esse método, foi-se utilizada uma rede neural com duas camas escondidas, a primeira com 512 dimensões e a segunda com 128. O diagrama que ilustra essa rede está a seguir. A taxa de aprendizado foi de 0.0001.

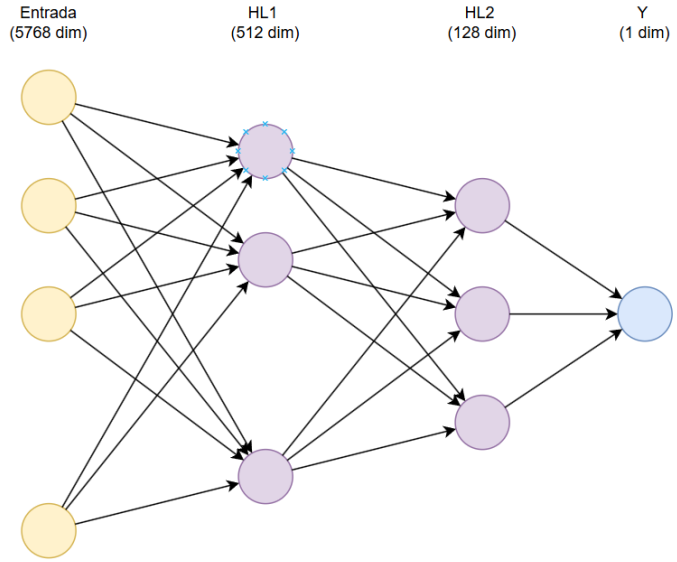

Figure 4. Rede neural utilizada

O treino foi configurado com um limite de épocas como sendo 10, mas com um contador de paciência de 5, i.e, após 5 épocas sem melhora no F1-Score, o treino para para evitar *overfitting*.

4. Resultados

4.1. Método 1

4.2. Método 2

O treino no método 2 foi interrompido na nona época devido ao contador de paciência ter se esgotado, e o melhor resultado obtido foi também na nona época. Testes subsequentes com mais épocas e mais flexibilidade no contador não melhoraram significativamente o resultado.

Acurácia	Precisão	Recall	F1	AUC
0.9711	0.9686	0.9676	0.9681	0.9962

Table 4. Resultados obtidos com o método 2

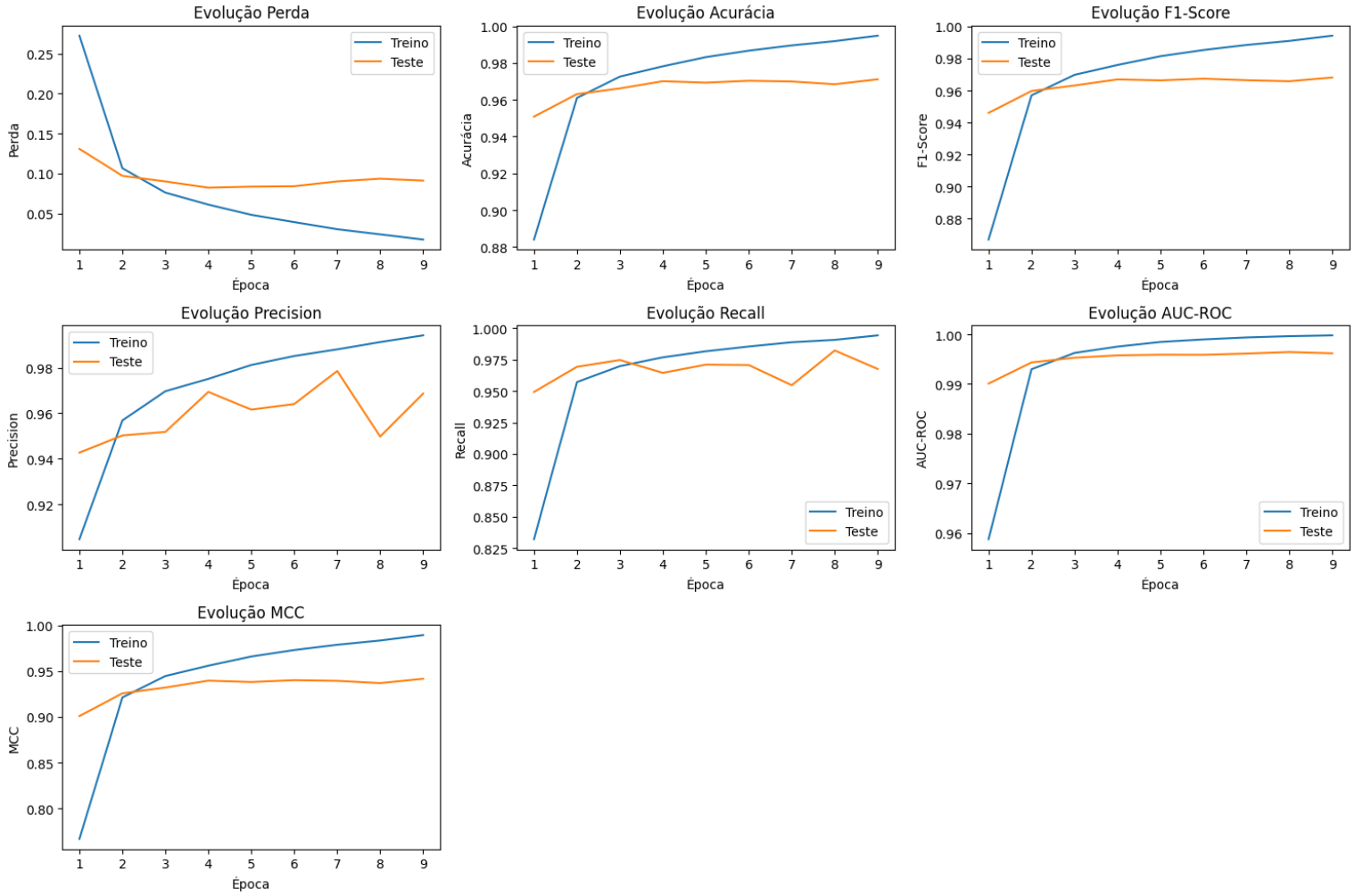

Figure 5. Resultados obtidos com o método 2 durante o treinamento

5. Conclusões

References