

Rede Neural LSTM Para Classificação de Fake News

Douglas Felipe Alves Lima¹

¹Instituto Metr pole Digital – Universidade Federal do Rio Grande do Norte (UFRN)

Resumo. *Nesse relat rio apresentamos um sistema de classifica  o de fake news. O modelo treinado   baseado em Redes Neurais LSTM(Long Short Term Memory) aplicado a base WEL_FAKE [Verma et al. 2021]. O modelo treinado tem 97% de acur cia*

1. Introdu  o

Nesse trabalho propomos um classificador para identificar Fake News, treinado na base de dados WEL_FAKE. O classificador em quest o trata-se de uma rede neural com x camadas, sendo a primeira uma camada de Embedding que   treinada na base de dados junto com o restante da rede, em seguida duas delas camadas LSTM (Long Short Term Memory) que s o camadas de rede recorrentes, e por fim uma camada de rede densa que transforma o resultado das camadas posteriores em uma resposta final (sim ou n o)

A base de dados   formada por 71.537 exemplos sendo desses 36509 noticias verdadeiras e o restante noticias falsas como podemos ver na figura 1

2. Fundamenta  o Te rica

O modelo implementado funciona baseado em duas tecnologias principais: Word Embeddings, rede LSTM.

Word Embedding   uma t cnica em Processamento de Linguagem Natural (PLN) que tenta traduzir palavras de um vocabul rio em vetores de n meros reais. Os vetores gerados possuem a propriedade de que, duas palavras com significado parecido formar o vetores pr ximos. Nesse trabalho utilizamos embedding com dimens o 128, ou seja cada palavra nos textos das noticias viraram um vetor de tamanho 128.

Redes LSTM s o uma vers o melhorada de redes recorrentes tradicionais, que na fase de back propagation passam pelo problema do *vanishing gradient* (desaparecimento do gradiente) ou seja, a relevancia do gradiente do erro vai diminuindo at  ficar irrelevante nas camadas mais iniciais da rede. Redes LSTM resolvem o problema do *vanishing gradient* deixando com que em alguns casos o gradiente flua sem ser atualizado.

3. Tratamento de dados

Foram realizado poucos tratamentos de dados, primeiro titulo e texto das noticias foram concatenados, em seguidas esses texto foram "tokenizados", isto  , as palavras do texto foram transformadas em n meros de acordo com a frequencia da palavra no dataset inteiro, sendo 1 a palavra mais frequente no dataset inteiro, e 286000 sendo a palavra menos utilizada, e tamb m o tamanho do vocabul rio no dataset.

Apos a tokeniza  o foi realizado uma redu  o na dimensionalidade dos dados, todas as sequencias de titulo texto, foram reduzidas a 600 palavras (no caso de noticias com

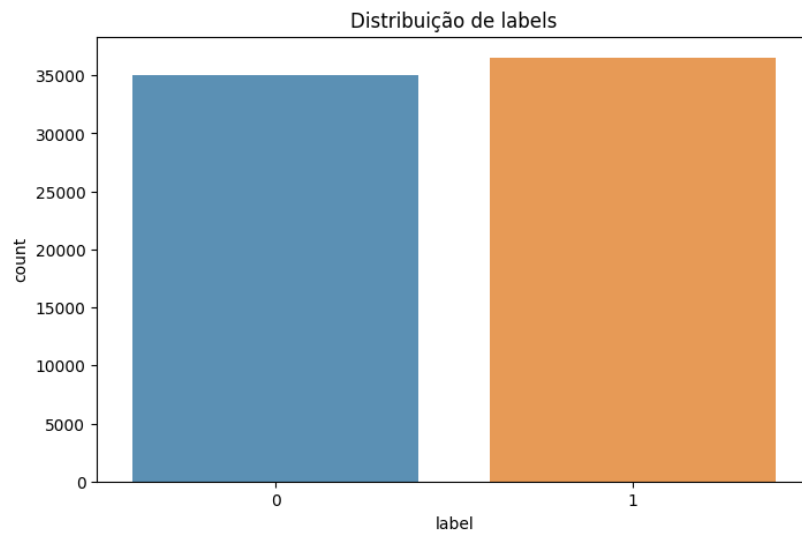


Figura 1. Distribuição das classes no dataset

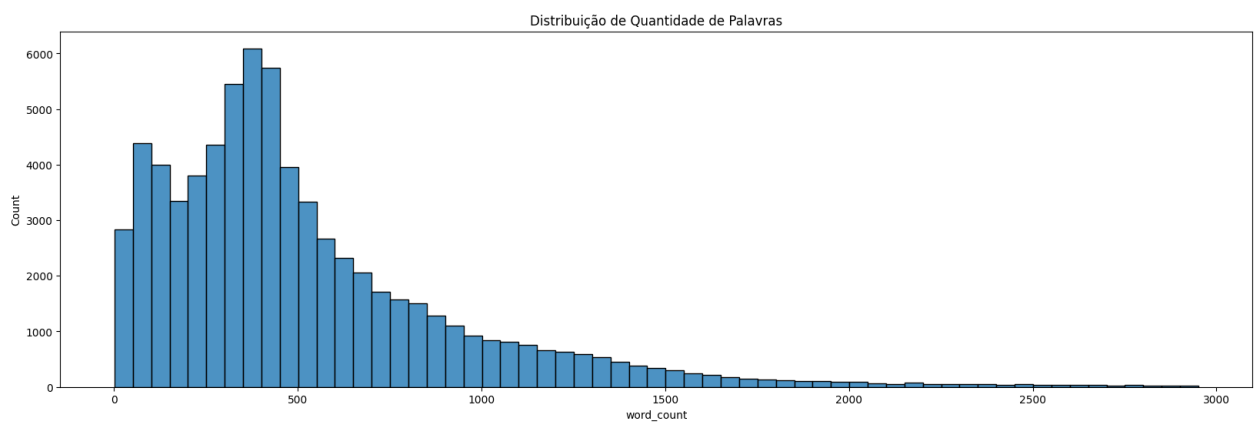


Figura 2. Distribuição da quantidade de palavras no dataset

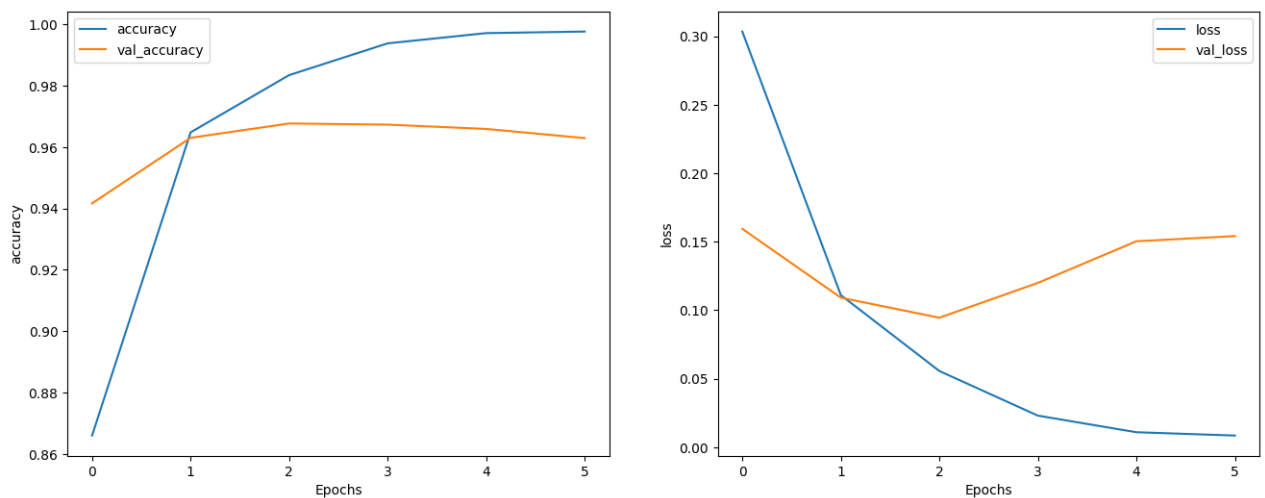


Figura 3. Gráfico de acurácia por época e erro por época

menos de 600 palavras zeros foram adicionados ao final do vetor) com o objetivo de reduzir o custo de memória. Como vemos no histograma da figura 2 a grande maioria dos exemplos tem menos de 600 palavras.

Como veremos na seção de Resultados essa decisão não comprometeu o desempenho do modelo.

4. A construção do modelo

O modelo construído tratasse de uma rede profunda sequencial em quatro camadas principais:

- Camada de Embedding (com dimensionalidade 128)
- Camada de LSTM 1 (com 100 neurônios de saída)
- Camada de LSTM 2 (com 100 neurônios de saída)
- Camada Densa com 64 neurônio de saída
- Camada Densa com 1 neurônio de saída

Essa configuração foi escolhida por performar melhor durante a calibração dos hiperparâmetros. O modelo foi treinado por durante 5 épocas de treinamento.

Durante o treinamento, entre as camadas densas, foi aplicado uma camada de Dropout (que só é ativa durante o treinamento) que faz com que em 50% dos valores da entrada sejam zerados, evitando o overfitting, além disso ele também aumenta os valores que sobram de forma a não mudar a soma das entradas.

Na figura 3 podemos acompanhar o treinamento do modelo, podemos ver na sequência azul a acurácia de treinamento e o erro de treinamento, e em amarelo podemos ver a acurácia de validação e o erro de validação. Como podemos ver não houve overfitting nem underfitting.

5. Resultados

Agora vamos apresentar os resultados obtidos na base de testes para o modelo treinado. O modelo apresentou as seguintes métricas:

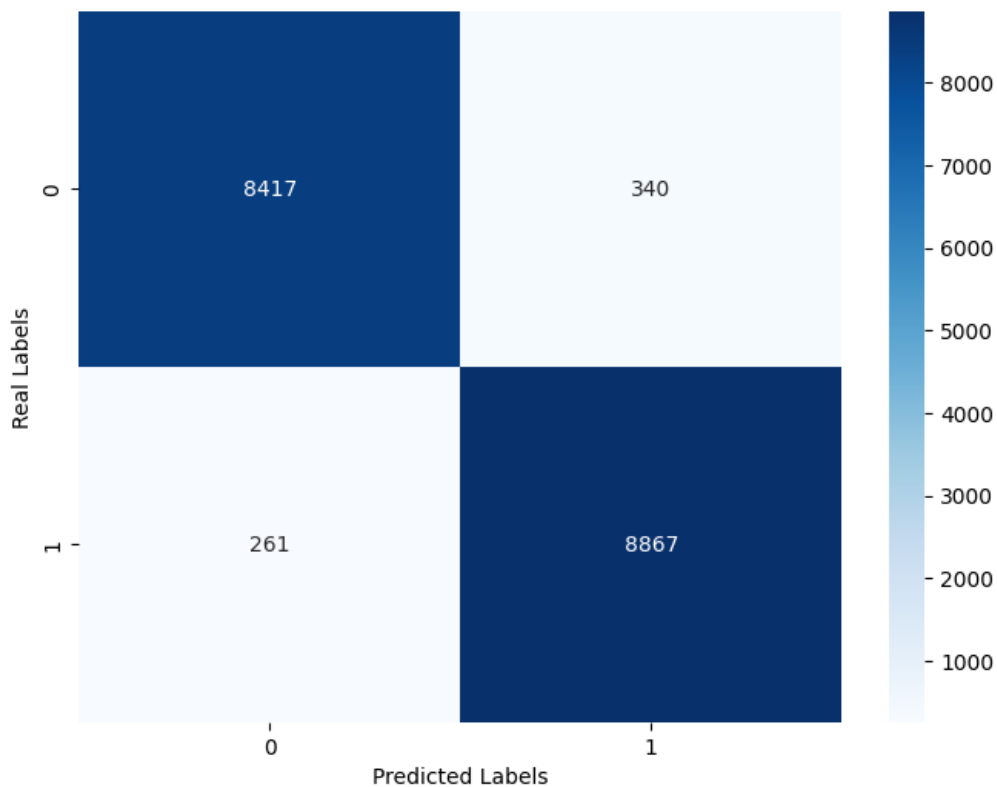


Figura 4. Gráfico de acurácia por época e erro por época

- Acurácia: 0,966396
- Precision: 0,961173
- Recall: 0,969923
- F1-Score: 0,965528

Na figura 4 podemos ver a matriz de confusão do modelo, como podemos ver o modelo tende a não cometer falsos positivos nem falsos negativos, o tornando um modelo bastante robusto.

Referências

Verma, P. K., Agrawal, P., Amorim, I., and Prodan, R. (2021). Welfake: Word embedding over linguistic features for fake news detection. *IEEE Transactions on Computational Social Systems*, 8(4):881–893.