Detecção de Sentimento em Avaliações de Produtos Escritas em Língua Portuguesa

Ihan Belmonte Bender[1]

[1]CDTEC – Universidade Federal de Pelotas (UFPEL)
Pelotas – RS – Brazil

ibbender@inf.ufpel.br

Abstract. *This brief paper presents a work designed to detect sentiment automatically in Portuguese language written texts, specifically in online shopping product reviews. To achieve this, a methodology based on previous work is used, using recurrent neural networks to indirectly learn to detect sentiment. The initial goal was to go beyond simply detecting negative or positive sentiment, using the model to predict the exact score (1 to 5) the users gave to the products based only on their review text. Unfortunately, this task was not complete perfectly, even though the model presented really good results on differentiating positive and negative texts.*

Resumo. *Este pequeno artigo apresenta um trabalho realizado para detectar sentimento de maneira automatizada em textos escritos em português, mais especificamente em avaliações de produtos comprados em lojas online. É utilizada uma técnica inspirada em outra já existente, utilizando redes neurais recorrentes para aprender detecção de sentimento de maneira indireta. O objetivo inicial do trabalho era ir além da simples detecção, tentando prever a nota exata dada pelo usuário (de 1 a 5). Porém, essa previsão não foi possível, embora o modelo tenha apresentado bons resultados em simplesmente diferenciar textos positivos de negativos.*

1. Introdução e Motivação

A detecção automática de sentimento em textos tem experimentado um crescente interesse nos últimos anos [Tang et al. 2009]. Diversos trabalhos na área foram realizados utilizando técnicas de aprendizado de máquina principalmente. O uso de tais técnicas exige que haja um conjunto de dados para treinar os diferentes modelos e outro para testá-los. Até hoje a maior parcela dos estudos tem focado em utilizar dados escritos em idiomas como o inglês, deixando outros idiomas, como a língua portuguesa, com menos avanços.

Mesmo com pouco foco em algoritmos de detecção de sentimento em textos escritos em português, existem algumas técnicas que tiveram bons resultados em outras linguagens e que com um algumas adaptações podem ter bons resultados em novos domínios. Isso se baseia principalmente na ideia de que os modelos utilizados nestes trabalhos não são construídos em torno da linguagem, mas sim em torno da estrutura de texto que grande parte dos idiomas ocidentais segue.

Neste trabalho é tido como objetivo utilizar uma das metodologias de detecção de sentimento automático que teve grande sucesso quando aplicado ao inglês, mas desta vez para a língua portuguesa. O assunto dos textos (análise de produtos de lojas online)

é mantido, mas existem algumas variações apresentadas neste em relação ao original, acrescentando algumas dificuldades.

Ao contrário do estudo original, onde se focou em prever apenas se uma análise era positiva ou negativa, neste trabalho teremos como objetivo prever, através da mensagem de avaliação, a nota data pelo consumidor. Esta modificação adiciona uma camada de dificuldade ao problema pois não basta saber se o texto é negativo ou positivo, mas sim quão negativo ou positivo ele é.

2. Dataset

O conjunto de dados escolhido para o desenvolvimento deste trabalho consiste em avaliações de produtos vendidos em lojas *online* da *Olist Store*. Os dados estão disponíveis no site *Kaggle*. O dataset obtido tinha diversas entradas sobre pedidos de produtos armazenados em diferentes arquivos *csv*, mas apenas o arquivo inerente às avaliações de produtos foram utilizadas. Neste, as colunas mais importantes para a realização deste trabalho eram: mensagem de avaliação e nota do consumidor. Outras colunas como identificador do pedido e identificador do produto foram ignoradas, já que não são necessárias durante a aplicação da metodologia.

Da maneira com que o dataset é fornecido é possível realizar sem dificuldades uma rápida avaliação e criação de muitos pares rotulados, tanto para treinamento quanto para teste dos modelos de redes neurais. É possível utilizar cada uma das mensagens como entrada da rede, enquanto as notas dadas pelos clientes consistem nos rótulos. Dessa maneira, mais do que simplesmente detectar o sentimento das análises, é tarefa da rede aprender a atribuir uma nota de um a cinco para o texto escrito.

No total haviam cerca de cem mil avaliações únicas no arquivo, porém grande parte delas tinham a coluna com a mensagem de avaliação em branco, impossibilitando o uso. Após remover linhas com este campo em branco foram totalizados cerca de trinta e sete mil exemplos rotulados. Foi possível observar nos exemplos restantes uma grande concentração de exemplos com nota cinco, sendo mais da metade dos dados assim. Em segundo lugar estavam as avaliações com nota um, enquanto avaliações com notas intermediárias surgiram mais raramente. A distribuição completa pode ser observada na figura 1

Após esse processamento dos dados foi possível dividir o restante em duas partes: dataset de treinamento e dataset de teste. O dataset de treinamento tem tamanho total de trinta mil exemplos, mantendo a mesma proporção de notas do que o conjunto de dados processados. Também mantendo esta proporção, mas com cerca de sete mil exemplos rotulados, foi construído o conjunto de dados de teste. Durante o treinamento dos modelos apenas o conjunto de treino pode ser utilizado pelos modelos, enquanto o teste é utilizado apenas para a avaliação ao final.

3. Processamento do Texto

Para processar o texto em si não foram utilizadas técnicas específicas de normalização. Acreditou-se que a rede neural poderia perder informações importantes caso os textos originais fossem modificados. Outro motivo para a não remoção foi o objetivo de verificar o desempenho nas redes a se adaptarem a erros na linguagem proposta. Porém, para que

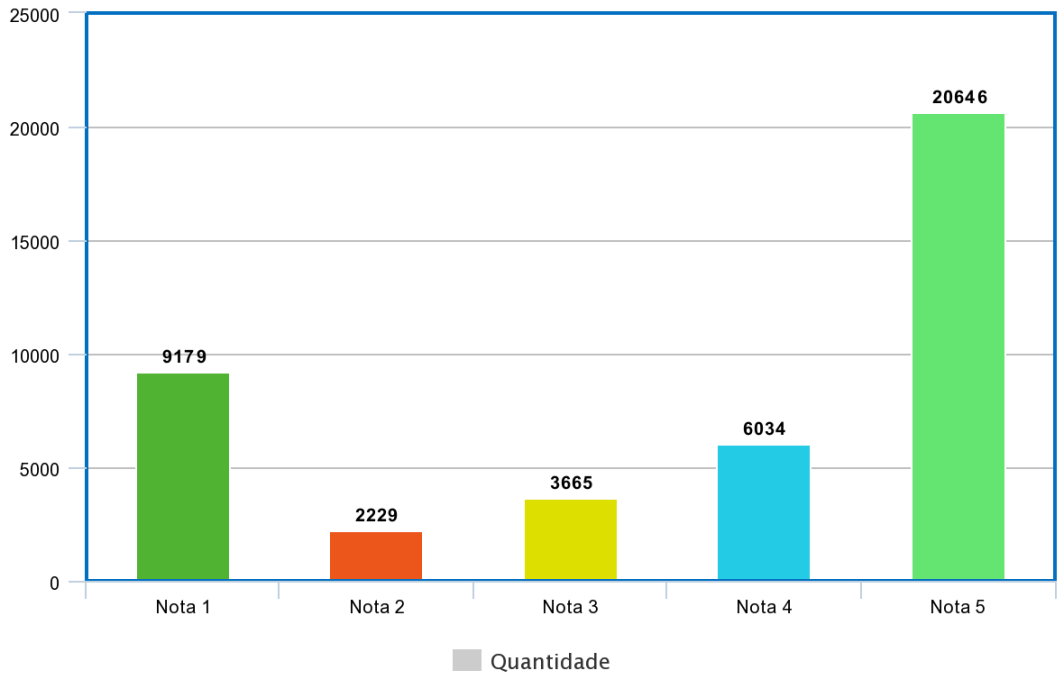

Figure 1. Distribuição de Exemplos Rotulados por Nota Atribuída

o texto pudesse ser consumido pela rede neural recorrente utilizada, foi necessário criar uma codificação de cada caractere em vetores numéricos.

Cada caractere dos textos de entrada para a mLSTM foi representado como um vetor de cento e vinte e oito (128) posições, sendo a de índice igual ao valor do caractere na tabela ASCII o único diferente igual a um, sendo os demais iguais a zero. Assim sendo, o caractere a por exemplo seria representado por um vetor de 128 posições com o valor zero, e o valor 1 na posição 97 (código ASCII da letra a).

4. Metodologia

A técnica utilizada para realizar a detecção de sentimento se baseou no trabalho desenvolvido em [Radford et al. 2017], onde uma rede neural recorrente é treinada para realizar a previsão do próximo caractere, dado um pedaço de uma análise de produto. Um exemplo de treinamento consiste em apresentar para a rede a sequência ”Adorei o prod” e esperar que o caractere previsto seja ”u”, já que a análise original consiste na frase ”Adorei o produto!”.

Assim como no trabalho original, o tipo de rede neural recorrente utilizada foi a multiplicative LSTM (mLSTM) [Krause et al. 2016], uma variação da LSTM original proposta em [Hochreiter and Schmidhuber 1997]. Esta rede foi utilizada apenas para o treinamento inicial de previsão de caracteres. O treinamento consistiu em fazer com que a rede previsse cada um dos caracteres (a partir do segundo) de cada avaliação por dez vezes.

Após o treino da rede neural recorrente ser finalizado foi criada uma rede neural artificial *multilayer perceptron* (MLP) de quatro camadas, sendo destas duas escondidas,

uma de entrada e uma de saída. Para cada texto de entrada, após a mLSTM processa-lo totalmente, dois estados escondidos da rede recorrente são concatenados criando um vetor de 4096 posições que serve de entrada para a MLP. Esta foi treinada passando por dez épocas tendo de prever o *score* dado pelo usuário baseado apenas nos estados escondidos da mLSTM. Os estados escolhidos foram o *hidden state* e o *cell state*, que representam o texto processado de alguma maneira criada durante o treinamento da rede recorrente.

Para tentar evitar o *overfitting* (quando a rede se adapta bem ao treino, mas não aos testes), foi utilizada a técnica de *dropout* que consiste em remover com probabilidade p cada um dos neurônios artificiais da rede durante o treinamento [Srivastava et al. 2014]. O valor de p escolhido foi de 0,5, fazendo com que em média metade dos neurônios fosse desativado por camada. A camada de saída não utilizou *dropout* e a função de ativação de todas as camadas foi a ReLU.

Para a otimização tanto da mLSTM quanto da MLP foi utilizada a técnica Adam [Kingma and Ba 2014], com a implementação realizada pela biblioteca Pytorch. A taxa de aprendizado também foi a mesma nos dois modelos, de 0,0001. A função de erro utilizada também para ambos foi a CrossEntropy, também implementada na biblioteca Pytorch.

5. Resultados e Conclusão

Ao final do treinamento de todos os modelos, o conjunto de aplicações foi exposto ao teste final. Este consistiu em fazer com que o modelo avaliasse todos os exemplos do conjunto de dados de teste e teve como objetivo identificar quantos exemplos seriam classificados corretamente.

Durante a realização foram obtidos resultados interessantes, e foi possível notar com clareza que o modelo tendenciava a sempre apostar em uma nota 5 (máxima) ou 1 (mínima), ignorando as intermediárias. Isso provavelmente se deve à pequena parcela de notas intermediárias nos dados de treinamento. Com exemplos não balanceados e tentando reduzir a função de erro, o modelo alcançava melhores resultados ignorando notas intermediárias. Assim, além de verificar quantos acertos perfeitos o modelo teve, será realizada uma medição que considerará as notas 1, 2 e 3 como negativas, enquanto as 4 e 5 serão positivas.

Na primeira avaliação o modelo teve uma acurácia de aproximadamente 65,16%. Nas avaliações com nota 1, o modelo acertou em 92,03% dos casos, enquanto nas avaliações nota 5 o resultado foi de 92,62%. Os resultados mostram que o motivo da baixa acurácia foi realmente a não adaptação fina do modelo para previsão de *scores* intermediários. Já na segunda avaliação, quando agrupamos mensagens com nota menor ou igual a três como negativas, o modelo detectou o sentimento correto em 84,06% dos casos negativos e 87,43% nos positivos.

Assim, foi possível verificar que mesmo com menos poder computacional do que o utilizado no trabalho original, e mesmo utilizando a técnica em uma linguagem diferente, foi possível criar um agente com bons resultados em detectar sentimento em textos de análise de produto. Infelizmente, quando foi necessário realizar a previsão da nota dada pelo usuário os resultados não foram tão bons, exigindo reajustes na metodologia como o balanceamento da base de dados de treinamento, por exemplo.

References

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization.

Krause, B., Lu, L., Murray, I., and Renals, S. (2016). Multiplicative LSTM for sequence modelling. *CoRR*, abs/1609.07959.

Radford, A., Jozefowicz, R., and Sutskever, I. (2017). Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Tang, H., Tan, S., and Cheng, X. (2009). A survey on sentiment detection of reviews. *Expert Syst. Appl.*, 36(7):10760–10773.