Relatório da Implementação de Análise de comentários usando Redes Neurais em Python 3.6

João Victor Oliveira Couto & Leandro Pereira Sampaio

[1]Engenharia de Computação – Universidade Estadual de Feira de Santana (UEFS)
Caixa Postal 252 e 294 – 44.036-900 – Feira de Santana – BA – Brazil

[2]Departamento de Ciências Exatas – Universidade de Feira de Santana (UEFS)
Feira de Santana, BA – Brazil.

jictyvoo.ecomp@gmail.com, leandrosampaio827@gmail.com

Abstract. *This is a report about a system developed by two Computer Engineer Students of Universidade Estudal de Feira de Santana. Will be presented here, the steps for the developing, and the decisions taken in all project development.*

Resumo. *Este é um relatório sobre um sistema desenvolvido por dois Estudantes de Engenharia de Computação da Universidade Estudal de Feira de Santana. Serão apresentados aqui, as etapas para o desenvolvimento e as decisões tomadas em todo o desenvolvimento do projeto.*

1. Introdução

O mundo passa hoje pela era da informática, com computadores, celulares, relogios, todos conectados uns com os outros através da rede mundial de computadores. Usando essa conexão com a rede mundial de computadores, uma grande quantidade de lojas acabam por criar sites próprios para vender seus pedidos de forma online.

Os sites de venda de produtos, geralmente, possuem um mecanismo de *review* para seus produtos e/ou vendedores. Dessa forma, é desejável que essa loja faça uma análise desses comentários dos consumidores para poder melhorar. Porém, com o advento da tecnologia, uma quantidade maior de pessoas tem acesso à internet a cada dia, levando a mais pessoas comprarem em lojas virtuais, e assim terem muitos *reviews* para serem analisados manualmente.

Tendo em vista essa realidade, faz-se ncessário algum tipo de sistema que venha a identificar o que os *reviews* tendem a dizer, se é algo positivo ou negativo. Esse relatório visa descrever um desses sistemas, o qual utiliza para realizar sua tarefa uma Rede Neural Artificial.

2. Métodologia

O sistema foi desenvolvido utilizando a linguagem de programação Python, de forma colaborativa, através do sistema GitHub, utilizando a IDE PyCharm Professional como ambiente para edição e teste do sistema desenvolvido.

2.1. Projeto da Rede Neural

A rede neural artificial foi desenvolvido utilizando a biblioteca pyBrain o qual utiliza como linguagem de programação associada o Python.

Essa rede foi desenvolvida utilizando inicialmente 1 camada escondida, 1 saída, e possuindo o número de neurônios de entrada como sendo a quantidade de palavras únicas (palavras distintas que aparecem em todas as frases da base), sendo assim foi utilizado 2075 entradas na rede.

Para criação dessa rede, utilizou o método *buildNetwork* existente na biblioteca do pyBrain. Vale lembrar que a rede foi criada utilizando um *bias*. Além disso, os pesos iniciais da rede foram escolhidos aleatoriamente pela própria biblioteca.

Na rede neural foi utilizado a função de ativação linear (função mais básica porque não altera a saída do neurônio) na camada de entrada e saída. No entanto para a camada oculta foi aplicado a função sigmóide, onde é frequentemente utilizada como uma função de transferência, porque introduz a não-linearidade e o resultado dessa função fica entre o intervalo [0,1].

Para o treinamento da rede neural foi utilizado o algoritmo backpropagation, nele a rede opera em uma sequência de dois passos. Primeiro, um padrão é apresentado à camada de entrada da rede. A atividade resultante flui através das camadas da rede, até que a resposta seja produzida pela camada de saída. No segundo passo, a saída obtida é comparada à saída desejada para essa ocorrência particular. Caso não esteja correto, o erro é calculado e propagado a partir da camada de saída até a camada de entrada, e os pesos das conexões das unidades das camadas internas vão sendo modificados conforme o erro é retropropagado.

Depois que a rede estiver treinada e o erro estiver em um nível satisfatório, ela poderá ser utilizada como uma ferramenta para classificação de novos dados. Para isto, a rede deverá ser utilizada apenas no modo progressivo (feed-forward). Ou seja, novas entradas são apresentadas à camada de entrada, são processadas nas camadas intermediárias e os resultados são apresentados na camada de saída, como no treinamento, mas sem a retropropagação do erro.

Na configuração da rede foram definidas algumas taxas:
- Taxa de Aprendizado: 0,01
- Taxa Momentum: 0,06

Foi observado que a quantidade de neurônios, interfere consideravelmente o tempo de treinamento da rede. Outros parâmetros interferem no treinamento, como a taxa de aprendizado e o momentum. A taxa de aprendizado está relacionada com a mudança dos pesos sinápticos, dessa forma uma taxa de aprendizado muito pequena implica numa trajetória suave e pequenas mudanças nos pesos a cada iteração. No entanto, requer-se um tempo de treinamento muito longo. Já o momentum leva em consideração o efeito de mudanças anteriores de pesos na direção do movimento atual no espaço de pesos.

Sendo assim, o aumento da taxa de aprendizado independente do valor definido para o momentum torna a rede instável. Por outro lado, diminuindo-se a taxa de aprendizado tem-se uma rede mais confiável.

2.2. Obteção da Base de Dados

Para obtenção da base de dados, optou-se por utilizar a loja online da Amazon Brasil, mais especificamente obtendo a informação da seção de livros da mesma.

Após a definição da fonte dos dados a serem obtidos, estudou-se a código-fonte das páginas de produto para que pudesse ser realizado o *scrape*, ou seja, a busca de dados nas mesmas.

Para realizar o *Web Scrapping* utilizou a biblioteca *Beautiful Soup 4* a qual permite a análise de arquivos no formato html. Usando essa bilbioteca foram criados métodos para obtenção dos comentários e seus respectivos valores de avaliação, bem como para detectar se aquele produto possuia mais uma página com comentários.

A base foi obtida a partir de 19 produtos da Amazon, rendendo dessa forma o total de 3248 cometários para a base. Os comentários obtidos na Amazon possuíam como classificação números de estrelas, tendo 5 no total. Visando facilitar a evolução da rede neural, definiu-se que todos os comentários acima de 3.5 estrelas foram considerados positivos, enquanto que os que estavam abaixo dessa linha, foram considerados negativos.

Os comentários obtidos nessa base eram compostos por cerca de 70% de sua totalidade como comentários positivos, o que pode vir a causar uma superadaptação da rede mais posteriormente.

2.2.1. Execução do *Bag of Words*

Para a execução efetiva da rede neural se faz necessário utilizar a técnica denominada *Bag of Words* que consiste em tratar o texto que será inserido na rede neural, garantindo seu funcionamento e maior eficiência.

Para o desenvolvimento do *Bag of Words* utilizou-se uma biblioteca chamada **nltk**, a qual disponibiliza uma série de funções para a criação do tratamento necessário das frases.

Inicialmente, foi criada uma função que remove todas as acentuações existentes nas frases para facilitar a execução posterior do tratamento das mesmas. Essa remoção foi realizada utilizando a função **normalize** presente na biblioteca padrão do Python. Acompanhada dela, foi utilizada a função RSLPStemmer da biblioteca NLTK para remover a raiz das palavras, com o intuito de diminuir a quantidade de dados a serem tratados devido ao fato de palavras sem a raiz possuírem o mesmo significado porém em diferentes conjugações e tempos verbais.

Logo em seguida, foi utilizado o NLTK para retornar uma lista de *Stop Words* e removê-las dos textos, as *Stop Words* são palavras que não são de grande utilidade no texto, tais como verbos de ligação, pronomes de ligação, artigos, dentre outros.

Visando aumentar o desempenho da rede neural, optou-se por remover as palavras que apareciam menos de 4 vezes em todas as frases da base. Ou seja, palavras que apareciam esporadicamente, de forma que apareciam em menos de 5% das frases contidas na base.

Levando em consideração que em Python a implementação de HashTables é na-

tiva, e sua denominação é dicionário, ao final do processo, foi gerado um vetor de dicionários contendo todas as palavras única que são essenciais à rede, como foi dito anteriormente, e apresentando as frequências que essas mesmas palavras apareciam em cada frase. Vale lembrar, que esse vetor gerado também possui o significado de cada frase, isto é se a mesma é positiva ou negativa.

2.3. Separação da base para teste e treino

Visando garantir uma maior eficiência no processo de treino, bem como uma boa validação dos testes realizados na rede, a base foi dividida em 80% dos comentários para o treinamento, e 20% para o teste da mesma. Dessa forma obteve-se no final 2598 comentários para o treinamento da rede e 650 comentários para o teste da rede neural.

3. Resultados e Discussões

Ao longo do desenvolvimento da rede neural, foram realizados cerca de 6 testes para cada tipo de treinamento utilizado, sendo um treinamento sem base de validação, e outro utilizando a base de validação. Os gráficos referentes ao treinamento sem validação podem ser visualizados nas figuras 1, 2, 3, 4, 5. Enquanto que os gráficos referentes ao treinamento da rede utilizando um conjunto de validação pode ser visualizado nas figuras 6, 7, 8, 9, 10.

3.1. Sem Base de Validação

No primeiro treinamento foram utilizados na rede neural, 30 neurônios na camada oculta e uma taxa de aprendizagem de 0,01. Inicialmente, para realizar os testes na rede neural de forma mais rápida, foi separada uma base contendo apenas 500 comentários, que nos testes finais foi substituida pela base anteriormente mencionada com 3248 comentários. Nos primeiros testes utilizando a base de 500 comentários foi verificado a taxa de 52% de sucesso enquanto que para a segunda base, foi analisado uma taxa de 55% de sucesso.

Figure 1. Primeiro Treinamento

No segundo treinamento foram utilizados na rede neural, 100 neurônios na camada oculta e uma taxa de aprendizagem de 0,05. Nos primeiros comentários de testes

foi verificado a taxa de 64% de sucesso e para o segundo de comentários foi analisado uma taxa de 58% de sucesso.

Figure 2. Segundo Treinamento

Para o terceiro treinamento foram utilizados na rede neural, 300 neurônios na camada oculta e uma taxa de aprendizagem de 0,08. Nos primeiros comentários de testes foi verificado a taxa de 68% de sucesso e para o segundo de comentários foi analisado uma taxa de 71% de sucesso.

Figure 3. Terceiro Treinamento

No quarto treinamento foram utilizados na rede neural, 500 neurônios na camada oculta e uma taxa de aprendizagem de 0,1. Nos primeiros comentários de testes foi verificado a taxa de 73% de sucesso e para o segundo de comentários foi analisado uma taxa de 62% de sucesso.

Figure 4. Quarto Treinamento

No último treinamento foram utilizados na rede neural, 1000 neurônios na camada oculta e uma taxa de aprendizagem de 0,01. Nos primeiros comentários de testes foi verificado a taxa de 74% de sucesso e para o segundo de comentários foi analisado uma taxa de 67% de sucesso.

Figure 5. Quinto Treinamento

Vale salientar, que pelo erro está em um valor muito baixo as linhas do gráfico aparentam está sempre no mesmo valor 0.

3.2. Com Base de Validação

Visando estabelecer uma relação semelhante ao treinamento da rede sem a base de validação, bem como aumentar a velocidade do treinamento, não foi definido uma taxa de erro da validação para que o treinamento parasse. Logo, seguiu-se o critério de realizar o treinamento da base 35 vezes.

Ao realizar os treinamentos da rede utilizando parte da base de treino para realizar a validação do treinamento, inicialmente notou-se um período de 2559.6095 segundos para a realização do mesmo.

Os testes foram realizados seguindo os mesmos parâmetros que os usados nos treinamentos sem validação, tendo como única mudança a quantidade de neurônios utilizadas na rede.

O tamanho da base de validação utilizada foi 25% da base de treino existente, dessa forma obteve-se 645 *reviews* para realizar a validação do treinamento da rede neural.

O primeiro treinamento foi realizado com a utilização de 380 neurônios na camada oculta. O resultado dessa base de treino foi de 75.846154% de acertos nos comentários selecionados. O gráfico contendo o resultado desse treinamento pode ser visualizado na Figura 6.

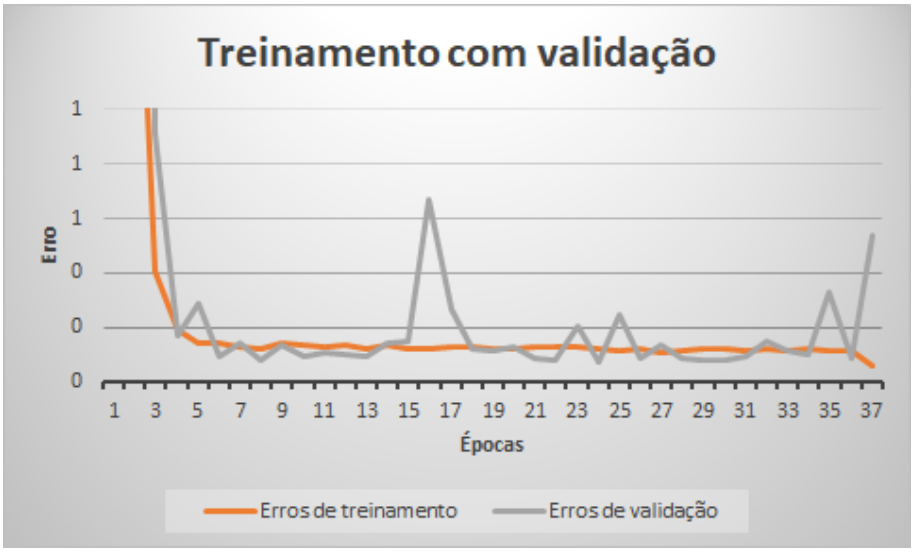

Figure 6. Primeiro Treinamento com Validação

Enquanto que o treinamento foi realizado com a utilização de 380 neurônios na camada oculta, o segundo treinamento foi realizado utilizando 400 neurônios, com o intuito de melhorar a taxa de sucesso da rede. O resultado dessa base de treino foi de 78.769231% de acertos nos comentários selecionados. Esse aumento na taxa de acerto, demonstrou que o acréscimo dee neurônios aumentou significativamente a eficiência da rede. O gráfico contendo o resultado desse treinamento pode ser visualizado na Figura 7.

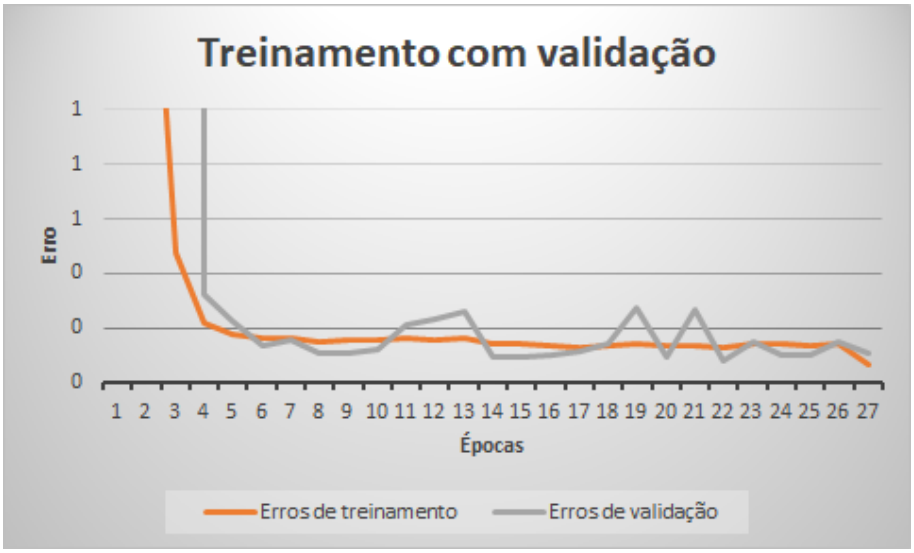

Figure 7. Segundo Treinamento com Validação

O terceiro treinamento foi realizado em uma rede contendo 320 neurônios na camada oculta. Tendo o resultado do treinamento o valor de 80.153846% de acertos nos comentários selecionados. A partir desse momento passou-se a ponderar a possibilidade de aumentar a eficiência da rede, ao diminuir o número de neurônios na mesma. O gráfico contendo o resultado desse treinamento pode ser visualizado na Figura 8.

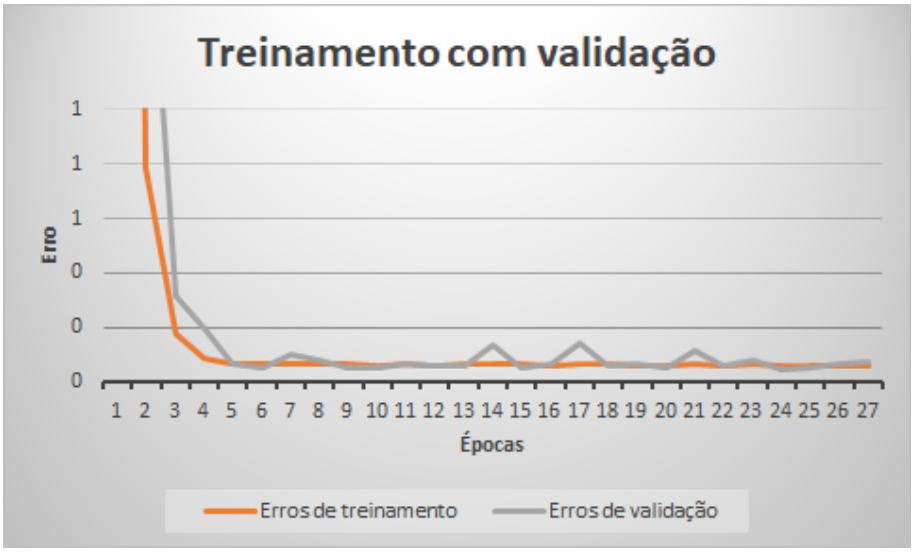

Figure 8. Terceiro Treinamento com Validação

Buscando ainda aumentar a eficiência da rede, reduziu-se o número de neurônios na camada oculta de forma mais significativa, tendo dessa forma 100 neurônios nessa rede. A porcentagem de acertos do treino dessa rede foi de 82.153846%. O gráfico contendo o resultado desse treinamento pode ser visualizado na Figura 9.

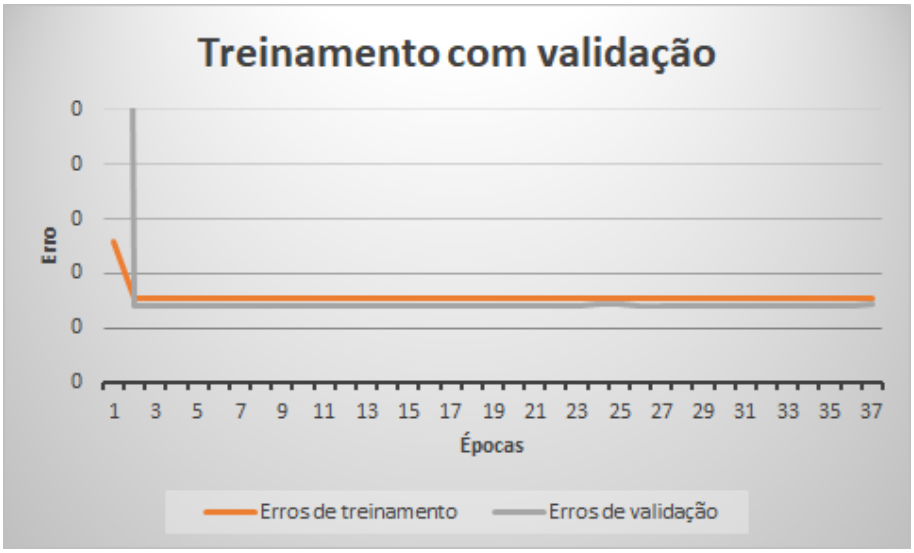

Figure 9. Quarto Treinamento com Validação

Utilizando como último teste da rede neural, a criação de uma rede utilizando apenas 1 neurônio, verificou-se que a porcentagem de acertos do treino dessa rede foi de 82.153846%. Esse valor tende a causar uma certa confusão, devido ao fato desse mesmo valor ser obtido utilizando uma rede com 100 neurônios. O resultado desse teste pode resultar em um problema proveniente da superadaptação da rede, pois ao utilizar apenas um neurônio, é difícil dicernir o que realmente a rede aprendeu ou não. O gráfico contendo o resultado desse treinamento pode ser visualizado na Figura 10.

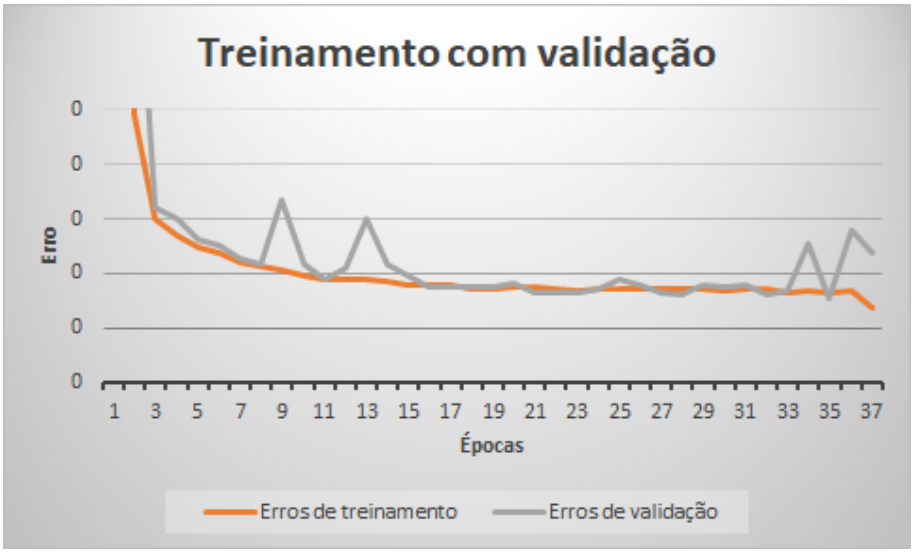

Figure 10. Quinto Treinamento com Validação

Vale ressaltar que durante todos esses treinamentos com validação, o número de neurônios variou, porém a quantidade de vezes que o treinamento foi executado na rede manteve-se constante, sendo esse valor como 35 vezes (buscando uma maior velocidade na obtenção dos resultados). A partir da análise dos últimos treinamentos executados é seguro deduzir que para aumentar a eficiência da rede, pode-se aumentar a quantidade de

neurônios nessa mesma rede, de modo que também seja ampliado a quantidade de vezes que a rede é treinada de forma proporcional.

4. Conclusão

Foi apresentado nesse relatório a descrição das etapas de desenvolvimento do sistema de Análise de Sentimentos em Comnetários. Observou-se que é necessário um tratamento prévio dos dados antes que os mesmos sejam enviados à rede neural.

É fato influenciador também a taxa de aprendizado em que a rede irá operar, de forma que o mesmo não seja nem muito alto, nem muito baixo, para que essa mesma rede consiga alcançar resultados mais desejáveis.

Pode-se deduzir também, baseado nos dados obtidos, que ao realizar o treinamento da rede através da separação da base apenas em teste e treinamento pode não vir a ser tão proveitoso, visto que a rede pode adaptar-se muito à base de treinamento. Sendo assim, realizar o treinamento da rede utilizando parte da base como conjunto de validação, pode vir a trazer mais benefícios no treinamento da mesma.

References

http://conteudo.icmc.usp.br/pessoas/andre/research/neural/MLP.htm
https://www.maxwell.vrac.puc-rio.br/7587/7587_5.PDF
http://www.inicepg.univap.br/cd/INIC_2005/inic/IC1%20anais/IC1-17.pdf