UNIVERSIDADE FEDERAL DO CEARÁ

INSTITUTO UNIVERSIDADE VIRTUAL

BACHARELADO EM SISTEMAS E MÍDIAS DIGITAIS

JOSUÉ DOS SANTOS MARQUES

ANÁLISE DE SENTIMENTOS UTILIZANDO UMA BASE DE TREINAMENTO LIMITADA

FORTALEZA - CEARÁ

2022

JOSUÉ DOS SANTOS MARQUES

ANÁLISE DE SENTIMENTOS UTILIZANDO UMA BASE DE TREINAMENTO LIMITADA

Trabalho de Conclusão de Curso apresentado ao curso de Sistemas e Mídias Digitais da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Sistemas e Mídias Digitais.

Orientadora: Profª. Dra. Ticiana Linhares Coelho da Silva

FORTALEZA - CEARÁ

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M318a Marques, Josué dos Santos.
Análise de sentimentos utilizando uma base de treinamento limitada / Josué dos Santos Marques. – 2022.
38 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto UFC Virtual, Curso de Sistemas e Mídias Digitais, Fortaleza, 2022.
Orientação: Profa. Dra. Ticiana Linhares Coelho da Silva.

1. Análise de sentimentos. 2. Classificação de textos. I. Título.

CDD 302.23

JOSUÉ DOS SANTOS MARQUES

ANÁLISE DE SENTIMENTOS UTILIZANDO UMA BASE DE TREINAMENTO LIMITADA

Monografia apresentada ao Curso de Sistemas e Mídias Digitais da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Sistemas e Mídias Digitais.

Aprovada em: 07/02/2022.

BANCA EXAMINADORA

_____

Prof. Dra. Ticiana Linhares Coelho da Silva (Orientadora)

Universidade Federal do Ceará (UFC)

_____

Prof. Dr. Alysson Diniz dos Santos

Universidade Federal do Ceará (UFC)

_____

Prof. Dr. Windson Viana de Carvalho

Universidade Federal do Ceará (UFC)

AGRADECIMENTOS

Primeiramente, a Deus pela força de vontade e a coragem de transpassar as barreiras.

À minha esposa, Ana Karine pela presença e companhia em todos os momentos.

Aos padres: Pe. José Carlos, Pe. Regis Teles e Pe. Jânio José que me apoiaram em suas orações.

À todos os meus amigos que me motivaram.

À minha orientadora Dra. Ticiana Linhares.

Não fui Eu que ordenei a você? Seja forte e corajoso! Não se apavore nem desanime, pois o Senhor, o seu Deus, estará com você por onde você andar.
(Josué 1: 9)

RESUMO

A análise de sentimentos é um algoritmo de classificação que necessita de dados rotulados para o treinamento da sua rede neural. Este trabalho levantou a questão se era possível a criação de uma rede neural a partir de uma pequena base de dados, visto a dificuldade de uma rotulagem manual de grandes massas de dados. A partir da observação desta base classificada manualmente foi levantado critérios e executado rotulações automáticas em mais dados a fim de obter uma base de treinamento com a quantidade de dados classificada necessária. Finalizando com o treinamento de um modelo e aferindo suas métricas baseadas na base de teste, obtendo um satisfatório resultado de 77% de acurácia.

Palavras-chave: Análise de Sentimentos, Classificação de textos.

ABSTRACT

Sentiment analysis is a classification algorithm that needs labeled data to train your neural network. This work raised the question whether it was possible to create a neural network from a small database, given the difficulty of manually labeling large masses of data. From the observation of the manually classified base, criteria were raised and automatic labeling was performed on more data in order to obtain a training base with the necessary amount of classified data. Finishing with the training of a model and gauging its metrics based on the test base, obtaining a satisfactory result of 77% of accuracy.

Keywords: Sentiment Analysis, Text Classification.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

PLN	Processamento de Linguagem Natural
SPS	Sem ou Pouco Sentimento
CS	Com Sentimento
TF-IDF	*Term Frequency – Inverse Document Frequency*
RNN	Redes Neurais Recorrentes
LSTM	*Long Short-Term Memory*
HTTP	*Hypertext Transfer Protocol*
HTML	Linguagem de Marcação de Hipertexto
NILC	Núcleo Institucional de Linguística Computacional
USP	Universidade de São Paulo
TP	*True Positive* - Verdadeiro Positivo
FP	*False Positive* - Falso Positivo
FN	*False Negative* - Falso Negativo
TN	*True Negative* - Verdadeiro Negativo
URL	*Uniform Resource Locator*
CSV	Comma-Separated-Values
NLTK	Natural Language Toolkit
BiLSTM	LSTM Bidirecional

SUMÁRIO

1. INTRODUÇÃO

A evolução das tecnologias da informação vem promovendo diversas mudanças na sociedade em geral. Entre elas está a disponibilização de uma quantidade cada vez maior de informações, resultado principalmente do aumento da capacidade de processamento e armazenamento. Este fenômeno torna-se cada vez mais evidente e vem sendo observado por diversos estudiosos da área (SILVA, 2012).

Essa situação propicia muitas oportunidades de uso dessa informação para a tomada de decisão, porém, lança muitos desafios em como armazenar, recuperar e transformar essa informação em conhecimento (BOVO, 2011).

Profissionais que trabalham com a ciência de dados desenvolveram ferramentas baseadas nestes desafios apontados por Bovo. Uma dessas ferramentas é a análise de sentimentos.

O principal objetivo da análise de sentimentos é definir técnicas automáticas capazes de extrair informações subjetivas de textos em linguagem natural, como opiniões e sentimentos, a fim de criar conhecimento estruturado que possa ser utilizado por um sistema de apoio ou tomador de decisão. A identificação de sentimentos em textos é uma das áreas de pesquisa mais destacadas em Processamento de Linguagem Natural (PLN) desde o início dos anos 2000, quando se tornou uma área de pesquisa muito ativa (Liu, 2012).

Uma das principais abordagens para o problema de extração de sentimentos em textos é embasada nos conceitos de aprendizagem de máquina partindo da definição de características que permitam distinguir entre sentenças com diferentes sentimentos e o treinamento de um modelo com sentenças previamente rotuladas. Gerando um modelo que seja capaz de identificar o sentimento em sentenças até então desconhecidas. Composta por técnicas supervisionadas, pelo fato de exigir uma etapa de treinamento de um modelo com amostras previamente classificadas (BENÉVOLO; RIBEIRO; ARAÚJO, 2015).

Este trabalho levantou a questão se era possível criar uma rede neural a partir de uma pequena base de dados, visto a dificuldade de preparar amostras previamente classificadas que necessitem de uma rotulagem manual.

Diante desta questão, este trabalho utilizou o caso de uso de notícias jornalísticas, onde rotulou uma base de notícias como: Sem ou Pouco Sentimento (S.P.S.) ou Com Sentimento (C.S.). Para analisar quanto o jornalista consegue

remover suas emoções ao redigir notícias. Com o objetivo de contribuir para a análise de sentimentos, verificando se é possível treinar um modelo com uma base de dados reduzida. Este mesmo estudo poderia ser aplicado em qualquer base ou contexto que tenha dados categorizados.

2. REFERENCIAL TEÓRICO

2.1. Análise de Sentimentos

De acordo com Pak e Paroubek (2010) existem três níveis principais de classificação em análise de sentimento: em nível de documento, em nível de frase e em nível de aspecto.

- **Em nível de documento:** a tarefa consiste em analisar um documento inteiro para então classificá-lo em um sentimento positivo ou negativo.
- **Em nível de frase:** visa classificar o sentimento expresso em cada frase. O primeiro passo é identificar se a sentença é subjetiva ou objetiva. Se a sentença é subjetiva, essa tarefa irá determinar se a frase exprime opiniões positivas ou negativas.
- **Em nível de aspectos:** este nível baseia-se na ideia de que uma opinião é constituída por um sentimento (positivo ou negativo) e um alvo (de opinião). Por exemplo, mesmo que a frase “Embora o serviço não seja tão bom, eu ainda amo este restaurante” tenha claramente um tom positivo, não se pode dizer que esta frase é inteiramente positiva. De fato, a frase é positiva a respeito do restaurante, mas é negativa a respeito dos serviços. Portanto, o objetivo deste nível é descobrir os sentimentos relacionados às entidades e seus aspectos.

Para o trabalho, é conveniente o nível de documento, para a análise da notícia como um todo.

2.2. Processamento de Linguagem Natural (PLN)

O PLN é a subárea da Inteligência Artificial que estuda a capacidade e as limitações de uma máquina em entender a linguagem dos seres humanos. O objetivo do processamento de linguagem natural é fornecer aos computadores a capacidade de entender e compor textos. “Entender” um texto significa reconhecer o

contexto, fazer análise sintática, semântica, léxica e morfológica, criar resumos, extrair informação, interpretar os sentidos, analisar sentimentos e até aprender conceitos com os textos processados (ROSA, 1997).

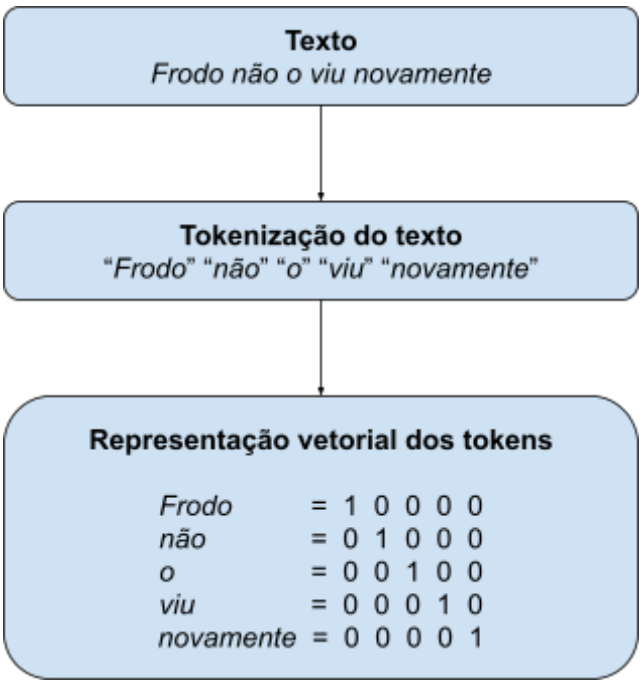
Um texto pode ser fragmentado em diferentes partes: palavras ou caracteres. Esses fragmentos são chamados de *tokens* e a divisão do texto em tais *tokens* é chamada de *tokenização*. Existem diferentes formas de representação das palavras ou *tokens,* algumas delas são *Bag of Words* e *Word Embeddings*, que serão abordadas a seguir.

2.3. Bag of Words

Após aplicada a *tokenização*, são associados vetores numéricos aos *tokens* gerados. Estes vetores são as *Bag of Words*, ou mala de palavras, que são uma matriz contendo uma variação entre 0 e 1 para representar a significância de uma palavra em uma determinada sentença. Uma alternativa de construção dessa matriz é ao invés de usar 0 e 1, usar a frequência em que a palavra ou *token* aparece no(s) documento(s) ou usar a frequência invertida (TF-IDF).

Esta abordagem trata cada documento como um conjunto de palavras individuais. Considerando cada palavra como uma palavra-chave em potencial. Exemplificados abaixo no fluxograma da Figura 01.

Figura 01 - Fluxograma com os passos da vetorização de um texto



Fonte: Elaborado pelo autor (2022).

2.4. Stop Words

As “palavras de parada” (*stop words*) são palavras que não adicionam sentido ao texto, pois são usadas para coesão e dar contexto, mas não fazem sentido específico quando olhadas individualmente. Por exemplo “as”, “e”, “de”, “com”, “foi”, “que”.

Portanto, o processo necessário em análises que usam PLN é eliminar as palavras vazias (*stop words*), que são extremamente comuns no idioma, mas que servem apenas para fazer a ligação entre frases ou palavras. As *stop words* não trazem um significado adicional para o texto.

2.5. Word Embeddings

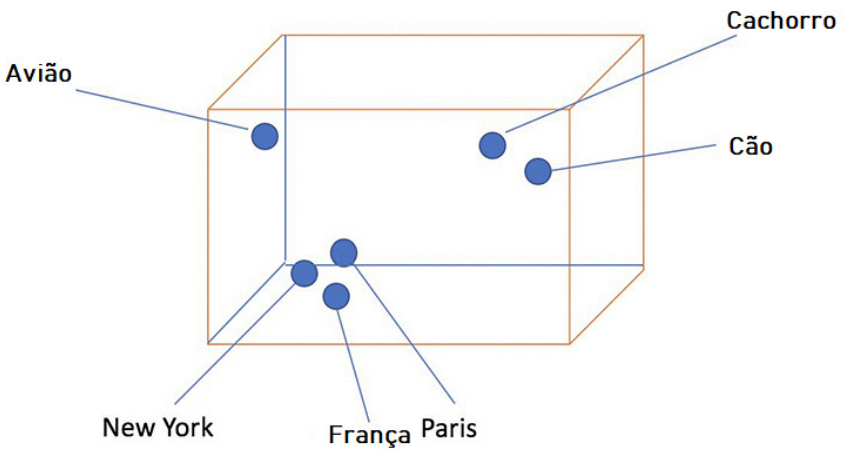
Representações de palavras são importantes para muitas tarefas de PLN. Os *embeddings* de palavras são basicamente uma forma de representação de palavras que liga a compreensão humana da linguagem à de uma máquina. Os *embeddings* são representações distribuídas de texto em um espaço n-dimensional (GUPTA, 2019).

A representação em vetor de palavras treinadas em conjuntos de dados de pesquisa pode ajudar a incorporar relações complexas entre as respostas que estão sendo analisadas e o contexto específico no qual a resposta foi feita.

O *Word Embedding* é um conjunto de técnicas que mapeiam em um espaço estatístico, a semântica e a sintática de uma linguagem natural. Dessa forma, palavras de um conjunto de texto são mapeadas para vetores. O espaço destes vetores é denominado *embedding space (GOLDBERG, 2017)*.

Palavras como “Paris” e “França” são mapeadas em vetores próximos devido ao seu grau de similaridade, assim como sinônimos como “cão” e “cachorro” (CARVALHO, 2018). Exemplificado abaixo na Figura 02.

Figura 02 - Representação do *Embedding Space*



Fonte: Elaborado pelo autor (2022).

2.6. Aprendizagem Profunda

Do inglês *Deep Learning,* ou Aprendizagem Profunda, é o uso de redes neurais com várias camadas e inúmeros parâmetros a serem treinados para alguma tarefa, como análise de sentimento, reconhecimento de imagens, entre outras. Trata de redes neurais artificiais que imitam a complexidade e o funcionamento dos neurônios. São utilizadas para soluções complexas e precisas.

É o aprendizado de máquina que treina computadores para realizar tarefas humanas, de forma natural a partir de redes neurais artificiais. Ao invés de usar algoritmos com equações pré definidas, o aprendizado profundo ajusta parâmetros básicos, para que através do reconhecimento de padrões em várias camadas de processamento, o computador seja capaz de aprender (DALALANA, 2020).

As redes neurais mais frequentemente utilizadas são as RNNs e LSTM.

- **RNN, Redes Neurais Recorrentes ou *Recurrent Neural Networks*.** Estas redes são compostas por um módulo reciclável, que processa as palavras da sequência de entrada uma por uma – ou seja, de forma recorrente. As RNN são eficazes com o processamento de informação sequencial (CECCON, 2020).
- **LSTM ou *Long Short-Term Memory*,** é uma variante da rede neural recorrente. Em uma arquitetura que “lembra” valores em intervalos arbitrários. A LSTM é bem adequada para classificar, processar e prever séries temporais com intervalos de tempo de duração desconhecida (DEEP LEARNING BOOK, 2019).

3. TRABALHOS RELACIONADOS

Este capítulo apresenta alguns trabalhos relacionados a esta monografia, que foram selecionados visto a semelhança temática e a linguagem de programação assumida.

3.1. Análise de sentimento em redes sociais utilizando combinação de classificadores

Erikson Júlio (2017) propôs um método para estimar sentimentos em redes sociais na língua portuguesa, tendo como foco o Twitter. O método utiliza de uma abordagem baseada em algoritmos de aprendizado de máquina que é denominado *Comitê*, no qual combina a predição de um conjunto de seis algoritmos e define o valor predito como o mais votado entre eles, considerando que o voto entre eles tem peso.

Chegou a conclusão que o *Comitê* é uma proposta interessante para ser utilizada em um conjunto de dados que contém poucos registros. Porém o algoritmo *Naive Bayes* pode ser indicado para bases maiores, apresentando os melhores modelos de classificação para essa perspectiva e também tem um tempo de execução menor que os outros algoritmos utilizados.

Este trabalho visa a análise de matérias jornalísticas que possuem uma quantidade maior de texto do que postagem no Twitter, porém os estudos de Erikson abrem oportunidades de métodos para diferentes quantidades de registros de banco de dados.

3.2. Mineração de textos: análise de sentimento utilizando tweets referentes às eleições presidenciais 2014

O trabalho de Zarathon (2014) utilizou a técnica de mineração e classificação de dados para extrair o sentimento dos usuários do Twitter em relação aos candidatos à presidência do Brasil no ano de 2014. Classificando-os como positivo, negativo, neutro ou ambíguo.

Utilizou o classificador *Naive Bayes*, que é utilizado para classificar dados baseados em um modelo computacional, é um dos mais utilizados no mundo para o aprendizado de máquinas. A classificação utilizando as técnicas de *Bayes* consiste em fazer um registro de treinamento que mais tarde seja aplicado a um conjunto de

dados para que sejam classificados baseados no que o modelo gerado na fase de treinamento aprendeu.

Zarathon teve um resultado bastante satisfatório na análise de uma grande massa de dados, caracterizando o público da rede social analisada.

Semelhante ao trabalho anterior quanto a fonte de dados e ao tamanho dos textos.

3.3. Identificando emoções em manchetes de notícias escritas em português do Brasil utilizando *Naïve Bayes*

André Davys (2016) tinha o objetivo de construir um modelo capaz de identificar uma das seis emoções básicas (alegria, tristeza, surpresa, medo, raiva e desgosto) em manchetes de notícias escritas em português utilizando o classificador *Naïve Bayes*.

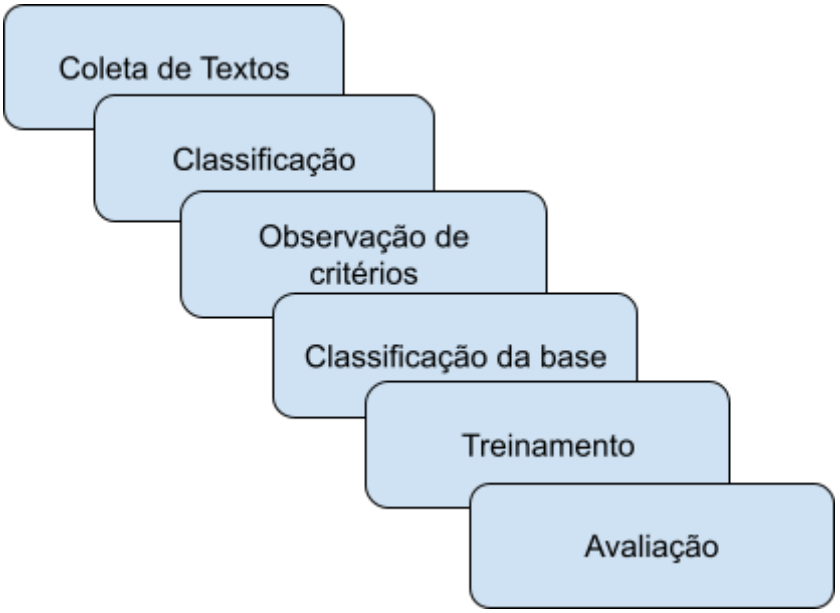
Foram aplicadas técnicas de processamento de linguagem natural e aprendizado de máquina no desenvolvimento do método e na realização dos experimentos. André Davys concluiu que é possível identificar emoções de forma automatizada.

Existem diversos pontos em comum entre este trabalho e o de André Davys, porém este será focado estritamente no âmbito do estado do Ceará de onde foram capturadas as notícias da base como será visto a seguir.

4. METODOLOGIA

Como pode ser observado na figura 03, Inicialmente foi desenvolvido um projeto para realizar a coleta dos textos. Em seguida foi realizada a rotulação manual de alguns textos. Analisando estas notícias criou-se critérios para a classificação do restante da base. Com toda a base rotulada foi preparado o modelo e aferido suas métricas em comparação com a base que foi rotulada manualmente.

Figura 03 - Metodologia



Fonte: Elaborado pelo autor (2022).

4.1. Coleta de textos de notícias

Para realizar a coleta da massa de dados foi desenvolvido um projeto [1]em *Python*, escolhido esta linguagem pela quantidade de material de suporte disponível e por possuir diversas bibliotecas que auxiliam no processamento textual.

Uma das bibliotecas escolhidas para agilizar o desenvolvimento foi o *Scrapy*, que recolhe informações na internet a partir de raspadores através do protocolo padrão da *web* HTTP.

Nestes raspadores de notícias foi implantado um número mínimo de validações para a proposta do trabalho. Os raspadores buscaram pela categoria política nas sessões de assunto dos sites selecionados e recolheram as notícias relacionadas à esta categoria. Apesar de se tratar de uma etapa de coleta, os textos sofreram um processo de limpeza, em que foram removidos os *hyperlinks* e *tags* HTML.

Os portais de notícia: G1, Diário do Nordeste e O Povo foram escolhidos seguindo o critério de importância, credibilidade e disponibilidade para *scraping* utilizando o *framework Scrapy*. Apesar de existirem outras mídias importantes, foi selecionado apenas um escopo reduzido para estudo.

[1] Projeto GitHub. Disponível em: <github.com/JosueSantos> Acesso em: 10 de Fev, 2022

4.2. Anotação dos textos em categorias

Foram rotuladas manualmente uma certa quantidade de notícias, devido ao tempo de produção do projeto. Observando as características dos rótulos disponíveis, como descrito na Tabela 01:

Tabela 01: Características das categorias

Com Sentimento (C.S.)	São indicados no texto as emoções do jornalista.
Sem ou pouco Sentimento (S.P.S)	Notícias que apenas informam o fato sem acréscimos. Sem a utilização exagerada de adjetivos.

Fonte: Elaborado pelo autor (2022).

A partir da análise da base de teste, foram desenvolvidos alguns critérios para criar a categorização para a base de treinamento. A fim de que toda a base de dados possua um rótulo para o treinamento do modelo.

4.3. Pré-processamento dos textos

Essa fase visou formatar os dados para que sejam utilizados no classificador, é necessário deixar os dados mais limpos e expressivos. Ao final dessa fase os dados tornaram-se mais simples e distantes da linguagem compreensível para as pessoas.

Utilizando os textos das notícias obtidas na coleta, foi aplicado técnicas de PLN. O objetivo deste pré-processamento foi realizar a limpeza das palavras que possuem pouca importância para a classificação do texto e facilitou a obtenção das palavras de característica do texto.

4.4. Representação dos textos em formato de *embeddings*

Para o melhor desempenho do classificador foi necessário uma robusta base de dados, para a representação das palavras da linguagem humana para a linguagem de máquina. Visto isto, utilizou-se textos em formato de *embeddings*

pré-treinados em português do NILC [2](Núcleo Institucional de Linguística Computacional) da USP.

4.5. Treinamento do modelo para categorização das notícias

O conjunto de dados foi dividido em dois conjuntos: um para a realização do treinamento do classificador, chamado como conjunto de treinamento, e o outro utilizado para avaliar o classificador, chamado conjunto de teste.

O modelo foi treinado com as notícias do conjunto de treinamento juntamente com seus respectivos rótulos para se realizar o aprendizado de máquina, deste modo, torna-se possível para o modelo do classificador associar as palavras do texto a classe e construir seu aprendizado. Este aprendizado será utilizado na fase de teste para o classificador predizer as classes das notícias.

Após essa etapa, aplicou-se o modelo sobre o conjunto de testes. Os resultados obtidos nesta aplicação foram metrificados com base na próxima seção deste trabalho.

4.6. Avaliação da qualidade do modelo baseado em métricas de precisão e *recall*

As métricas de avaliação que foram utilizadas são baseadas nos seguintes conceitos:

- **TP** (*true positive* ou verdadeiro positivo): Representa a quantidade de instâncias que foram classificadas corretamente. Por exemplo, se o classificador classifica uma notícia como "SPS" e a classe desta notícia era realmente "SPS", temos um verdadeiro positivo para a classe “SPS”.
- **FP** (*false positive* ou falso positivo): Representa a quantidade de instâncias que não pertencem a uma classe, mas foram classificadas como pertencente a ela. Por exemplo, se o classificador classifica uma notícia como “SPS”, mas a mesma é "CS", temos um falso positivo para a classe “SPS”.
- **FN** (*false negative* ou falso negativo): Representa a quantidade de instâncias que pertencem a uma classe, mas foram classificadas como

[2] NILC. Disponível em: <http://www.nilc.icmc.usp.br/nilc/index.php> Acesso em: 10 de Fev, 2022

não pertencentes a ela. Por exemplo, se o classificador classifica a notícia como “CS”, mas ela era “SPS”, temos um falso negativo para a classe “SPS”.

- **TN** (*true negative* ou verdadeiro negativo): representa a quantidade de instâncias que não pertencem a uma classe e foram classificadas como não pertencentes a ela. Por exemplo, o classificador classifica uma notícia como “CS” e a notícia era “CS”, temos um verdadeiro negativo para todas as classes que não são “CS”, no caso deste trabalho só temos duas classes.

Foram avaliados os resultados do classificador quanto a acurácia, cobertura e precisão.

4.6.1. Acurácia

A acurácia é a métrica que calcula a taxa de acerto geral do classificador, ou seja, determina a taxa que o classificador obteve ao identificar corretamente a classe (OLIVEIRA, 2016). A acurácia é dada pela equação:

$$\frac{. \ T\,PA \ .}{QN}$$

Em que **T PA** representa a quantidade de verdadeiros positivos de todas as classes e **QN** é a quantidade de notícias presente no conjunto de testes.

4.6.2. Cobertura (*Recall*)

A cobertura (*recall*) é uma métrica para a avaliação de modelos que representa a porcentagem que o modelo de predição obteve para identificar corretamente os elementos de uma determinada classe (OLIVEIRA, 2016). A cobertura do classificador é dada pela equação:

$$\frac{. \quad TP \quad .}{TP + FN}$$

Em que é calculado os verdadeiros positivos e falsos negativos para cada classe.

4.6.3. Precisão

A precisão é uma métrica para a avaliação de modelos que representa a proporção da quantidade de instâncias classificadas corretamente em uma determinada classe pela quantidade de instâncias que foram classificadas à esta classe. Em outras palavras, diz para as instâncias classificadas em uma classe quantas estão corretas (OLIVEIRA, 2016). Dada pela equação:

$$\frac{TP}{TP + FP}$$

Em que é calculado os verdadeiros e falsos positivos para cada classe.

5. DESENVOLVIMENTO / RESULTADOS

5.1. Projeto

O projeto foi desenvolvido em Python e disponibilizado no *github.com*[3]. Possui uma arquitetura modularizada, onde cada módulo responsabiliza-se por um grupo de tarefas.

- ***Extractor* -** Responsável pela captura dos dados, onde se realiza os *crawlers* do *Scrapy*;
- ***Files* -** Onde são armazenados os arquivos extraídos e possui um utilitário com diversas funções de acesso a estes arquivos;
- ***Classifier* -** Responsável pela criação do modelo *Naive Bayes* e do *SentiStrength,* armazenando toda a lógica necessária para o tal.
- **Raiz -** Na raiz do projeto localizam-se os *scripts* de acesso, para que com apenas uma execução toda a funcionalidade seja atendida;

5.2. Coleta de textos

Dentre os *scripts* de acesso, foi executado diariamente durante alguns meses o *scriptUpdateURLs.py* que executava os *cralwers* necessários para a captura das URLs das notícias recentes que ainda não existiam na base de dados.

Foram consultados os três principais sites de notícias do Ceará: G1, O Povo e o Diário do Nordeste. A busca localizava a URL das notícias que tinham política como tema principal.

---

[3] Projeto GitHub. Disponível em: <github.com/JosueSantos> Acesso em: 10 de Fev, 2022

Após a localização das notícias mais recentes, era executado o *scriptExtractor.py* que realizava a coleta dos textos. Este *script também* remove *tags* HTML e *links*, além de descartar as notícias sem texto, no caso de notícias em vídeo, por exemplo. Salvando os dados em arquivos CSV.

Foram capturadas mais de 18 mil notícias datadas entre 2012 e 2021. A diversidade de datas é importante para não haver viés em determinados políticos.

Do site O Povo foram capturadas um total de 8.504 notícias, do G1 foram 7.101 e do Diário do Nordeste, por conta de barreiras de conteúdo pago, foram capturadas 2.570 notícias. Resultando um total de 18.175 como pode-se identificar na Tabela 02.

Tabela 02 - Quantidade de notícias capturadas

Origem	**Quantidade de Notícias**
O Povo	8.504
G1	7.101
Diário do Nordeste	2.570
Total	**18.175**

Fonte: Elaborado pelo autor (2022).

5.3. Classificação

Um dos obstáculos deste trabalho foi a classificação manual da base de dados. Visto que para o treinamento do modelo é necessário um rótulo classificando como CS ou SPS, para cada notícia.

Para obter essa rotulagem foi realizada uma análise detalhada do texto obedecendo alguns critérios. Para ser considerada CS a notícia deveria demonstrar a emoção do jornalista através de palavras, porém a citação de relatos de terceiros não será considerada, pois não são as palavras propriamente ditas do redator. Será considerada SPS a notícia que não demonstrar demasiada quantidade de emoção nas palavras do autor.

Levando em conta o tempo de execução deste trabalho, não foi possível a construção de um grande volume de dados rotulados manualmente. Desta maneira,

os dados classificados manualmente foram nomeados como **base de teste**[4], e a base de treinamento foi rotulada de maneira automática.

5.3.1. Classificação da Base de Teste

A partir da leitura das notícias já existentes na base de dados, foi realizada a análise e rotulagem manual alcançando um total de 300 notícias. Com 232 rotuladas como SPS e 68 rotuladas como CS. A base de teste está desbalanceada, mas isso já é esperado uma vez que a maioria das notícias tendem a ser SPS. Por este trabalho ser manual, seria inviável aumentar a base de notícias para além de 300 para a finalização do trabalho.

Esta denominou-se como base de teste, onde seus rótulos foram utilizados para a medição e avaliação a partir de comparações com as classificações obtidas pelo modelo.

5.3.2. Classificação da Base de Treinamento

- **Palavras de Citação**

Um dos fatores que foi considerado para classificar uma notícia como SPS foi a presença do relato da pessoa ou órgão envolvido na matéria. E geralmente, uma palavra de citação introduz ou finaliza esses relatos, por exemplo: "disse o presidente".

Observando esta característica, foi destacado as palavras encontradas que fazem esse papel de citação, em busca de um padrão entre essas palavras e a classificação.

Analisando as notícias rotuladas manualmente foi descoberto um padrão com a quantidade de palavras de citação em relação a quantidade de frases da notícia. Sendo que, se houvesse uma citação em pelo menos 60% das frases da notícia, esta teria maior chance de ser considerada SPS.

As palavras de citação mapeadas estão presentes na Tabela 03.

[4] Base de Teste. Disponível em: <github.com/JosueSantos/ground_truth.csv> Acesso em: 10 de Fev, 2022

Tabela 03 - Palavras de citação mapeadas

conforme	segundo	disse	declarou
acordo	afirma	argumenta	enumera
diz	explica	apontou	pesquisa
relatou	relatório	informou	afirmou
aponta	ressaltou	escreveu	anunciou
destacou	ressalta	frisou	mencionou
comenta	informaram	decidiu	determinou
concluiu	ibope	tse	eleitoral
agenda			

Fonte: Elaborado pelo autor (2022).

Desta forma, foi criado o primeiro critério para a classificação automática: possuir no corpo do texto uma quantidade de palavras de citação que superasse em 60% a quantidade de frases contidas no texto.

- **SentiStrength**

Com base na premissa que uma notícia SPS possui um texto com a ausência de fortes emoções, a próxima etapa para a classificação automática foi o processamento pela ferramenta *SentiStrength*[5] de Thelwall (2010).

Que avalia os sentimentos presentes nas palavras, baseado em um dicionário léxico que atribui às palavras com emoções positivas valores entre 1 e 5 e a palavras com emoções negativas valores entre -1 e -5. Analisando a sentença que recebe, a divide em tokens e para cada palavra que transmite uma emoção é atribuída uma pontuação determinada. Após pontuar todas as palavras, a ferramenta retorna uma pontuação geral dos sentimentos negativos, positivos e neutros.

[5] SentiStrength. Disponível em: <http://sentistrength.wlv.ac.uk/> Acesso em: 10 de Fev, 2022

Foi realizado um acréscimo ao dicionário léxico da ferramenta *SentiStrength*, com a mesclagem de sua base com a do *OpLexicon*[6] de Marlo Souza e Renata Vieira (2012), também conhecido como *Sentiment Lexicon*, que consiste de uma lista de palavras rotuladas como positivas e negativas. Este é um método léxico criado a partir de textos coletados em *reviews* de produtos em sites de compra.

A base de teste foi processada com o *SentiStrength* com o dicionário reforçado pelo *OpLexicon* e os valores encontrados foram comparados com os seus rótulos. Descobrindo que a maior frequência de rótulos de SPS estavam nas notícias com sentimentos positivos até +2 (+1 e +2) e em sentimentos negativos até -2 (-1 e -2). E notícias que ultrapassem os limites, tanto superiores a +2, quanto inferiores a -2, poderiam ser consideradas CS.

A partir destes critérios o *scriptBuildData.py* foi desenvolvido. Este *script* coleta todos os textos presentes nos arquivos CSVs, processando-os com a ferramenta *SentiStrength* para a captura dos pesos positivos e negativos. Realiza a contagem das palavras de citação e a contagem de frases e envia estes dados para a função *scoreClassifier* que realiza a lógica de classificação baseada nos critérios definidos.

A execução deste *script* gerou um novo arquivo CSV com os textos das notícias e rótulos chamado de **Base.csv**[7]. Após a criação deste arquivo foi dividido em base de teste e base de treinamento. A base de teste sendo as notícias rotuladas manualmente e a base de treinamento sendo rotulada a partir dos critérios.

Assim, às 17.875 notícias da base de treinamento foram rotuladas baseando-se no primeiro critério, e as notícias que não possuíam os 60% de palavras de citação por frase foram classificadas pelo *SentiStrength*, obtendo assim 9.536 notícias CS e 8.339 SPS.

Foi realizada a conferência da base de teste, comparando-se os rótulos manuais com os rótulos obtidos pelos critérios, para a observação de métricas. Obtendo uma acurácia de 57,33%, a cobertura do rótulo SPS em 57,33% e 57,35%

---

[6] OpLexicon. Disponível em: <https://www.inf.pucrs.br/linatural/wordpress/recursos-e-ferramentas/oplexicon/> Acesso em: 10 de Fev, 2022

[7] Base.csv. Disponível em: <github.com/JosueSantos/base.csv> Acesso em: 10 de Fev, 2022

nas CS. E a precisão das SPS obteve 82,10% enquanto as CS apenas 28,26%. Apresentando estas métricas na Tabela 04.

Tabela 04 - Métricas da classificação baseada nas palavras de citação e no *SentiStrength*

Acurácia	**57,33%**
Recall (Cobertura) SPS	57,33%
Precisão SPS	82,10%
Recall (Cobertura) CS	57,35%
Precisão CS	28,26%

Fonte: Elaborado pelo autor (2022).

- **Naive Bayes**

Em busca de uma base de treinamento mais assertiva, a mesma foi processada pelo classificador *Naive Bayes*, que utiliza um modelo probabilístico baseado nos rótulos existentes para classificar os dados. É baseado no "Teorema de Bayes", o qual foi criado por Thomas Bayes (1701 - 1761).

A Teoria da Decisão Bayesiana realiza uma simulação matemática do senso comum que classifica com a classe mais provável. Utilizando probabilidades para fazer previsões com base no conhecimento prévio das condições que podem estar relacionadas. Em outras palavras, ele usa probabilidades condicionais de cada recurso lexical ocorrendo em texto em seu respectivo rótulo nos dados de treinamento para chegar ao resultado.

Neste trabalho foi utilizado o classificador *Naive Bayes* presente na *NLTK*[8], que é uma biblioteca escrita para a construção de softwares que visam trabalhar com linguagem humana (*PLN*). Ela é desenvolvida na linguagem de programação Python.

Nesta etapa foi utilizado o *scriptClassifier.py*, que implementa de maneira Orientada a Objetos todas as funções necessárias para utilizar o *Naive Bayes*. Encapsulando em *ClassifierModel* o modelo, *ClassifierController* as funções de interação com o modelo e *ClassifierService* a camada externa para a utilização do modelo pelos scripts de outros módulos.

[8] Naive Bayes NLTK. Disponível em: <https://www.nltk.org/_modules/nltk/classify/naivebayes.html> Acesso em: 10 de Fev, 2022

Todas as notícias passaram por uma limpeza para evitar ruídos. Foram removidos os *stopwords*, com exceção da palavra “Não” pelo seu impacto no sentido do texto. Também foram removidas as pontuações presentes no texto.

Após, foi realizado o *stemming*, que é o processo de reduzir palavras flexionadas ou derivadas a sua base. Neste processo a tarefa é transformar palavras próximas em uma só, como ficar apenas com o radical dos verbos ou transformar todos os substantivos para o singular. Um exemplo da aplicação da técnica de *stemização* é reduzir as palavras “estudar, estudou, estudo e estudando” ao termo “estud”, que representa a base de todas as variações citadas.

O modelo foi treinado utilizando as notícias presentes na base de treinamento e seus respectivos rótulos, no caso a classificação obtida pelo *SentiStrength* anteriormente. Para a utilização posterior, o modelo foi salvo utilizando a biblioteca *pickle,* que permite a serialização de objetos, ou seja, os transforma em sequências de *bytes*. O método *saveModel* e *getModelSaved* no *ClassifierModel* realiza o registro e leitura do modelo quando necessário.

Com o modelo *Naive Bayes* treinado, executou-se o *scriptPopulateAnalytics.py* que avaliará novamente toda a base de dados, a partir do novo modelo para a captura desta nova opção de rótulo.

Este script retorna um novo arquivo CSV, chamado ***Base_Analytics.csv***[9] que possui o texto da notícia, sua classificação manual (para as que possuem), seu rótulo *SentiStrength* e seu rótulo obtido pelo *Naive Bayes*.

Observando a base de treinamento (17.875 notícias que **não** foram rotuladas manualmente) o Naive Bayes obteve 10.102 notícias como SPS e 7.773 CS.

Novamente foi medido as métricas, comparando-se os rótulos manuais das 300 notícias com os seus respectivos rótulos obtidos pelo modelo Naive Bayes. Obtendo 73,33% de acurácia, cobertura SPS de 84,91% e CS de 33,82%. Apesar da cobertura CS ter diminuído, a precisão aumentou, com 81,40% nas SPS e 39,66% nas CS. Como pode-se identificar na Tabela 05.

[9] Base_Analytics.csv. Disponível em: <github.com/JosueSantos/base_analytics.csv> Acesso em: 10 de Fev, 2022

Tabela 05 - Métricas das classificações *Naive Bayes* e *SentiStrength*

	Naive Bayes	***SentiStrength***
Acurácia	**73,33%**	**57,33%**
Recall (Cobertura) SPS	84,91%	57,33%
Precisão SPS	81,40%	82,10%
Recall (Cobertura) CS	33,82%	57,35%
Precisão CS	39,66%	28,26%

Fonte: Elaborado pelo autor (2022).

Encontrando então a base de treinamento com duas opções de rótulo automático, a primeira com a cobertura em 57% para ambos os rótulos e outra processada pelo *Naive Bayes* que teve uma melhoria de 27,59% nas SPS e uma piora de 23,53% nos rótulos CS.

Porém, observando a acurácia, a primeira opção processada pelo *SentiStrength* possui 57,33% enquanto a rotulada pelo *Naive Bayes* alcançou 73,33%. Houve uma melhora significativa no resultado da acurácia, pois na base classificada manualmente temos uma grande quantidade de rótulos SPS em relação aos CS (232 SPS e 68 CS). Sendo necessário ainda a redução dos falsos positivos encontrados.

Redução esta, que foi obtida pelo relacionamento semântico das palavras disponibilizado pelo *Word Embedding*, que será visto à frente.

5.4. Pré-Processamento

Para esta etapa foi necessário um nível de processamento de máquina muito superior ao disponível para o desenvolvimento. Com isso foi utilizado o Google Colab[10], que é uma ferramenta *web* do Google que permite que qualquer pessoa escreva e execute código *Python* pelo navegador e é especialmente adequado para *machine learning*, análise de dados e educação. Mais tecnicamente, o Colab é um serviço de *notebooks* hospedados do *Jupyter* que não requer nenhuma

[10] Google Colab. Disponível em: <https://colab.research.google.com/> Acesso em: 10 de Fev, 2022

configuração para usar e oferece acesso gratuito a recursos de computação (COLAB). O projeto [11]desenvolvido foi disponibilizado no github.

O arquivo *Base_Analytics*.csv foi enviado ao Colab para a continuidade do desenvolvimento.

Os textos das notícias passaram novamente pela limpeza semelhante ao que ocorreu com o Naive Bayes, com exceção do *stemming*, pois a variação das palavras não interfere com a utilização de *Word Embedding*.

Foi utilizada a função *word tokenize* da biblioteca NLTK para a quebra dos textos em *tokens*, onde cada palavra é separada em um *array*.

Nesta etapa, foi carregado o *Word Embedding* pré-treinado de 50 dimensões da Língua Portuguesa, criado pela *NILC-Embeddings*[12]. E para cada *token* capturou-se o seu identificador dentro dos vocabulários do *embedding*.

Com a base de dados preparada, foram separados pelo tipo de origem do rótulo. A base de teste com os rótulos manuais e a base de treinamento com os rótulos classificados pelo *Naive Bayes*.

- Base de Treinamento *Naive Bayes*;
- Base de Teste;

5.5. Treinamento

Para a construção do modelo proposto neste trabalho foi utilizado a biblioteca *Keras* que é uma *API* de redes neurais em *Python*. Ela provê uma estrutura que permite compilar redes neurais combinando camadas de diferentes dimensões e funções de ativação, tornando o ciclo de desenvolvimento de novos modelos de aprendizado de máquina muito mais rápido.

Inicialmente carregando um modelo sequencial, que permite inserir camadas de uma rede neural em série, onde o *output* da primeira camada serve como *input* da segunda, e assim por diante.

A primeira camada adicionada foi um *Embedding*, que ajusta os pesos da rede neural e gera os vetores de espaço multidimensional fixo para representar a entrada. Passando como parâmetro o *Word Embedding* Pré-treinado.

---

[11] Jupyter Notebook. Disponível em: <github.com/JosueSantos/notebook.ipynb> Acesso em: 10 de Fev, 2022

[12] NILC. Disponível em: <http://nilc.icmc.usp.br/nilc/index.php/repositorio-de-word-embeddings-do-nilc> Acesso em: 10 de Fev, 2022

A segunda camada adicionada foi o *SpatialDropout1D*, que realiza um descarte programático de algumas amostras para aliviar quando as características estiverem fortemente correlacionadas.

A terceira e mais importante camada, foi a *LSTM* Bidirectional, ou BiLSTM, que é um modelo de processamento de sequência que consiste em duas *LSTM*s: uma levando a entrada para a frente e a outra para trás. Os BiLSTMs aumentam efetivamente a quantidade de informação disponível para a rede, melhorando o contexto disponível para o algoritmo.

A quarta camada foi um *Dropout*, semelhante ao *SpatialDropout1D*, utilizada para evitar que determinadas partes da rede neural tenham muita responsabilidade e consequentemente, possam ficar muito sensíveis a pequenas alterações.

E por fim a camada *Dense*, que tem como objetivo calcular uma função de ativação em conjunto com os dados de entrada e pesos.

Na Figura 04 abaixo, o resumo de todas as camadas criadas no modelo.

Figura 04 - Resumo do modelo fornecido pelo *Keras*

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 2396, 50)          46480300

 spatial_dropout1d (SpatialD  (None, 2396, 50)         0
 ropout1D)

 bidirectional (Bidirectiona  (None, 128)              58880
 l)

 dropout (Dropout)           (None, 128)               0

 dense (Dense)               (None, 2)                 258

=================================================================
Total params: 46,539,438
Trainable params: 46,539,438
Non-trainable params: 0
_________________________________________________________________
```

Fonte: Elaborado pelo autor (2022).

Com o modelo carregado foi executado o método "*fit*" que treina o modelo em um determinado número de "*epochs*" (iterações em um conjunto de dados), neste caso definido como 10 *epochs* e com determinado "*batch_size*" (tamanho do lote

que define o número de amostras que serão propagadas pela rede), definido como 1.000 amostras para a otimização de tempo de processamento. Os dados de entrada devem ser dividido em "*x_train*" e "*y_train*" (domínio e imagem de uma parte dos dados do conjunto) para treino, assim como "*x_test*" e "*y_test*" (domínio e imagem da outra parte dos dados) para validação.

Para obter os dados de entrada foi utilizado o método *train_test_split* da biblioteca *Sklearn*, que permite aos usuários dividir seus dados em conjuntos de treinamento e teste. Com a divisão 80-20, a base de dados de treinamento rotulada pelo *Naive Bayes* foi particionada aleatoriamente em *x_train* e *x_test*, com seus respectivos rótulos em *y_train* e *y_test*.

Com o modelo treinado, a base de teste foi submetida a classificação pelo modelo BiLSTM para a metrificação, com a comparação dos rótulos manuais com os respectivos rótulos obtidos pelo BiLSTM obteve-se 74,67% de acurácia e precisão no rótulo SPS de 84,48%, valores bem semelhantes aos obtidos pelo modelo *Naive Bayes* que eram 73,33% e 81,91%, respectivamente. Porém houve melhoria deste novo modelo com as métricas do rótulo CS, com cobertura de 41,18% e precisão de 43,75%.

A classificação realizada pelo *Naive Bayes* retornou um bom valor de acurácia, que foi mantido no modelo BiLSTM, porém ainda existia um alto valor de falsos positivos. Utilizando as relações das palavras obtidas através do *Word Embedding* o novo modelo conseguiu realizar esta redução.

Desta forma, a porcentagem em que o modelo de predição obteve para identificar corretamente os elementos em seus respectivos rótulos foi melhorada, como pode ser visto na Tabela 06.

Tabela 06 - Métricas das classificações BiLSTM, *Naive Bayes* e *SentiStrength*

	BiLSTM	***Naive Bayes***	***SentiStrength***
Acurácia	**74,67%**	**73,33%**	**57,33%**
Recall (Cobertura) SPS	84,48%	84,91%	57,33%
Precisão SPS	83,05%	81,40%	82,10%
Recall (Cobertura) CS	41,18%	33,82%	57,35%
Precisão CS	43,75%	39,66%	28,26%

Fonte: elaborado pelo autor (2022).

Em caráter experimental, também foi utilizado os rótulos do *SentiStrength* para o treinamento do modelo BiLSTM para verificar se haveria alguma melhoria nos seus valores. Foi recriado todo o processo utilizado anteriormente, alterando-se apenas os rótulos.

Com a base de teste submetida novamente a classificação do modelo, e analisados os respectivos rótulos manuais e do modelo, foram obtidos valores melhores do que os anteriores. Com acurácia de 77,33%, precisão do rótulo SPS de 84,91% e 51,47% no rótulo CS.

Houve uma melhora de 2,6% comparado ao BiLSTM baseado no *Naive Bayes* e 20% se comparado a classificação das palavras de citação com *SentiStrength*. Na Tabela 07 é exibido a comparação entre as métricas obtidas.

Tabela 07 - Métricas das classificações *BiLSTM* baseadas no *SentiStrength* e no *Naive Bayes*

	BiLSTM Senti	**BiLSTM Naive**	***Naive Bayes***	***SentiStrength***
Acurácia	**77,33%**	**74,67%**	**73,33%**	**57,33%**
Recall (Cobertura) SPS	84,91%	84,48%	84,91%	57,33%
Precisão SPS	85,65%	83,05%	81,40%	82,10%
Recall (Cobertura) CS	51,47%	41,18%	33,82%	57,35%
Precisão CS	50,00%	43,75%	39,66%	28,26%

Fonte: Elaborado pelo autor (2022).

5.6. Avaliação

A cada nova etapa de melhoria a quantidade de classificações como SPS aumentavam enquanto as CS diminuíam. Como nota-se na Tabela 08 abaixo, levando em consideração que o modelo BiLSTM *Senti* obteve seus rótulos a partir do *SentiStrength*.

Tabela 08 - Quantidade de rótulos por modelo para a base completa de 18.175 notícias

	SPS	***CS***
BiLSTM Senti	10.079	8.096
BiLSTM Naive	10.847	7.328
Naive Bayes	10.344	7.831
SentiStrength	8.501	9.674

Fonte: Elaborado pelo autor (2022).

Com modelo BiLSTM partindo dos rótulos do *Naive Bayes* não houve grandes melhorias visto que já recebia os dados com uma maior quantidade de rótulos SPS. Quando partiu dos dados classificados pelo *SentiStrength* recebeu os dados com uma divisão melhor e conseguiu um resultado satisfatório, agora com falsos positivos reduzidos, ou seja, foi reduzido a quantidade de vezes em que rotula-se erroneamente.

Com este modelo estabelecido, avaliou-se novamente toda a base de dados para observar a classificação geral da base de dados analisada. Das 18.175 notícias, 55% delas foram categorizadas como SPS e 45% CS.

5.7. Ameaças à Validade

Foi observado alguns fatores que poderiam ser melhorados neste trabalho. Podendo ser revisado para trabalhos futuros. Um deles foi o tamanho das notícias. Textos menores tinham maior probabilidade de serem considerados SPS, visto o critério da presença de palavras de citação. Desta forma, sites que possuam uma média de frases por notícia maior podem receber mais rótulos como CS.

Uma solução seria um outro critério de classificação para notícias grandes, obtida a partir de uma observação mais atenta a este tipo de texto.

Outro fator observado foi a qualidade da limpeza do texto obtido, pontuações importantes para a avaliação, como por exemplo as aspas, foram apagadas no processo de captura das notícias. Cabe a este ponto uma melhoria nos métodos de raspagem ou a busca por outra fonte de dados.

Um cuidado importante que vale a pena mencionar é que o aprendizado de máquina supervisionado requer que os dados sejam rotulados, o que pode significar que qualquer subjetividade e preconceito nos dados ou rótulos sejam refletidos no modelo. Desta forma pode-se declarar que o modelo gerado reflete uma parte da opinião de quem gerou os dados iniciais.

6. CONCLUSÃO E TRABALHOS FUTUROS

Levantou-se a questão da possibilidade de criar uma rede neural a partir de uma pequena base de dados e rótulos pré-definidos. Foram utilizadas técnicas de mineração e classificação de texto. O trabalho coletou, separou, tratou e armazenou os dados.

A base de dados passou por alguns métodos em busca da melhoria da acurácia do modelo. Onde a cada nova etapa de aprimoramento, a quantidade de notícias rotuladas como SPS tendia a aumentar, enquanto a CS tendia a diminuir. O resultado foi bastante satisfatório. Foram analisadas as notícias sobre política de três grandes jornais do Ceará. Obtendo um resultado de 55% das notícias coletadas poderem ser consideradas Sem ou com Pouco Sentimento (S.P.S.).

Foi observado uma interferência quanto ao tamanho das notícias. Aquelas com poucos parágrafos foram rotuladas seguindo o mesmo critério adotado para as notícias grandes. Observando esta característica pode-se encontrar um novo padrão para refinar ainda mais os rótulos.

Sendo que, o ideal seria que toda a base de dados fosse rotulada por humanos para o treinamento do modelo, não sendo possível obter esta meta, é satisfatório para trabalhos futuros uma maior quantidade de notícias rotuladas manualmente. Com uma base de teste mais expressiva daria mais confiabilidade ao resultado.

REFERÊNCIAS

SILVA, Thales N.. **Uma Arquitetura Para Descoberta De Conhecimento A Partir De Bases Textuais**. 2012. Disponível em 10 de Fevereiro de 2022: <https://repositorio.ufsc.br/bitstream/handle/123456789/61448/TCC-Thales-final.pdf>

BOVO, Alessandro Botelho. **Um Modelo de descoberta de conhecimento inerente à evolução temporal dos relacionamentos entre elementos textuais**. 2011. Disponível em 10 de Fevereiro de 2022: <https://repositorio.ufsc.br/xmlui/bitstream/handle/123456789/95137/290620.pdf?sequence=1&isAllowed=y>

LIU, B. **Sentiment analysis and opinion mining. Synthesis lectures on human language technologies**, Morgan & Claypool Publishers. 2012.

BENEVENUTO, Fabrício; RIBEIRO, Filipe; ARAÚJO, Matheus. **Métodos para análise de sentimentos em mídias sociais.** 2015. Disponível em 10 de Fevereiro de 2022: <https://homepages.dcc.ufmg.br/~fabricio/download/webmedia-short-course.pdf>

PAK, A.; PAROUBEK, P. **Twitter as a corpus for sentiment analysis and opinion mining.** 2010.

ROSA, G. LUIS JOÃO. **O Significado da Palavra para o Processamento de Linguagem Natural.** 1997.

GUPTA, Shashank . **Word Embeddings in NLP and its Applications**, 2019. Disponível em 10 de Fevereiro de 2022: <https://hackernoon.com/word-embeddings-in-nlp-and-its-applications-fab15eaf7430>

GOLDBERG, Yoav. **Neural network methods for natural language processing. Synthesis lectures on human language technologies**, 2017

DE CARVALHO, MATHEUS HERMÍNIO. **Estudo Comparativo dos Métodos de Word Embedding na Análise de Sentimentos**. 2018. Disponível em 10 de Fevereiro de 2022: <https://www.cin.ufpe.br/~tg/2018-2/TG_CC/tg_mhc.pdf>

DALALANA, Anna Carolina. **Descubra como o Deep Learning está influenciando nossos hábitos e transformando a indústria**. 2020. Disponível em 10 de Fevereiro de 2022: <https://www.voitto.com.br/blog/artigo/deep-learning>

CECCON, Denny, **Transformers em PLN (parte 1 de 2): mecanismo de atenção** 2020. Disponível em 10 de Fevereiro de 2022: <https://iaexpert.academy/2020/02/26/transformers-em-pln-parte-1-de-2-mecanismo-de-atencao/>

DEEP LEARNING BOOK, **Capítulo 51 – Arquitetura de Redes Neurais Long Short Term Memory (LSTM).** 2019. Disponível em 10 de Fevereiro de 2022: <https://www.deeplearningbook.com.br/arquitetura-de-redes-neurais-long-short-term-memory/>

AGUIAR, Erikson Júlio. **Análise De Sentimento Em Redes Sociais Utilizando Combinação De Classificadores.** 2017. Disponível em 15 de Janeiro de 2021 em: <http://200.201.11.152/bitstream/handle/123456789/240/Tcc-analise-sentimento-vfinal.pdf?sequence=1&isAllowed=y>

VIANA, Zarathon Lopes. **Mineração De Textos: Análise De Sentimento Utilizando Tweets Referentes Às Eleições Presidenciais 2014.** 2014. Disponível em 15 de Janeiro de 2021 em: <http://www.repositoriobib.ufc.br/000017/000017d1.pdf>

OLIVEIRA, André Davys C. M. **Identificando Emoções Em Manchetes De Notícias Escritas Em Português Do Brasil Utilizando Naïve Bayes.** 2016. Disponível em 15 de Janeiro de 2021 em: <http://www.repositoriobib.ufc.br/000032/0000323e.pdf>

THELWALL, M., Buckley, K., Paltoglou, G., Cai, D., and Kappas, A. **Sentiment strength detection in short informal text**. 2010

SOUZA, M.; VIEIRA, R. **Sentiment Analysis on Twitter Data for Portuguese Language. 10th International Conference Computational Processing of the Portuguese Language**, 2012. Disponível em 15 de Janeiro de 2021 em: <https://www.inf.pucrs.br/linatural/wordpress/wp-content/uploads/2017/08/PROPOR_2012.pdf>

CHOLLET, François. **Deep Learning with python**. 2017. Disponível em 15 de Janeiro de 2021 em: <https://livebook.manning.com/book/deep-learning-with-python/chapter-6/1>

COELHO, Ticiana Linhares. **Modelos Neurais para Extração de Entidades e Desambiguação em Textos.** 2020

SILGE, J., & ROBINSON, D. (2017). **Text mining with R: A tidy approach.** O'Reilly Media, Inc.

PROVOST, F., & FAWCETT, T. (2013). **Data Science for Business: What you need to know about data mining and data-analytic thinking.** O'Reilly Media, Inc.

PUSTEJOVSKY, J., & STUBBS, A. (2012). **Natural Language Annotation for Machine Learning: A guide to corpus-building for applications.** O'Reilly Media, Inc.