UNIVERSIDADE FEDERAL DE UBERLÂNDIA
CAMPUS MONTE CARMELO

Gabriel Franco Dias Graciano

APRENDIZADO PROFUNDO PARA CLASSIFICAÇÃO DE SENTIMENTOS EM MICROBLOGS

MONTE CARMELO
2019

GABRIEL FRANCO DIAS GRACIANO

APRENDIZADO PROFUNDO PARA CLASSIFICAÇÃO DE SENTIMENTOS EM MICROBLOGS

Trabalho de Conclusão de Curso submetido à Universidade Federal de Uberlândia, como requisito necessário para obtenção do grau de Bacharel em Sistemas de informação

Monte Carmelo, 09 de julho de 2019

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

GABRIEL FRANCO DIAS GRACIANO

Esta Monografia foi julgada adequada para a obtenção do título de Bacharel em Sistemas de Informação, sendo aprovada em sua forma final pela banca examinadora:

Prof. Dr. Murillo Guimarães Carneiro
Orientador
Universidade Federal de Uberândia - UFU

Prof. Dr. Carlos Roberto Lopes
Professor Convidado
Universidade Federal de Uberândia - UFU

Prof. Dr. Luiz Gustavo Almeida Martins
Professor Convidado
Universidade Federal de Uberândia - UFU

Monte Carmelo, 09 de julho de 2019

Agradecimentos

Agradeço primeiramente a Deus, por ter me iluminado e dado sabedoria durante a jornada de conclusão do curso.

Agradeço ao meu pai e à minha mãe, por terem me dado o apoio que precisei e me incentivado a nunca desistir dos meus objetivos. Aos meus amigos e familiares, pois estiveram ao meu lado e me deram a companhia necessária para enfrentar as minhas dificuldades.

Agradeço, sobre tudo, à minha avó, que descanse em paz, por ter me ensinado todos os pilares de bondade, humanidade e humildade que me foram necessários para chegar até aqui.

Meu muito obrigado ao meu orientador, o Prof. Dr. Murillo Guimarães Carneiro, por ter me acolhido na pesquisa e me incentivado a continuar, mesmo quando eu tentei desistir.

Por fim, gostaria de deixar meu agradecimento especial à minha namorada, e futura esposa, Alice Moreira Gomes, por ter sido a materialização do apoio moral nos meus momentos mais turbulentos, durante todo o curso.

Resumo

A Internet, atualmente, representa um dos maiores meios de comunicação e compartilhamento de informações. No Brasil e em vários outros países, a grande maioria da população a utiliza para uma série de atividades. Exemplos incluem as redes sociais e os *microblogs*, ambientes onde são frequentes as manifestações de opiniões e discussões sobre produtos, serviços ou outros assuntos de interesse geral. Neste contexto, a tarefa de classificação de sentimentos pode rotular opiniões expressas em tais veículos como positivas, negativas ou neutras, utilizando algoritmos para Processamento de Línguas Naturais (PLN) e Aprendizado de Máquina. O objetivo desse trabalho foi investigar a aplicação de técnicas de Aprendizado Profundo para classificação de sentimentos em publicações da rede social Twitter, considerando assuntos de interesse para a população. Especificamente, este estudo considerou a manifestação de opiniões sobre a corrida presidencial brasileira no ano de 2018. A coleta dos dados foi realizada por meio da *Tweepy*, uma Interface de Programação de Aplicativos (API) disponibilizada pelo Twitter, em todos os dias que houveram debates televisionados. A base de dados foi pré-processada utilizando técnicas de PLN. Para a realização dos experimentos, um conjunto relevante de algoritmos de classificação de dados foi selecionado a partir da literatura (naive Bayes, árvore de decisão, regressão logística, máquina de vetores de suporte), e comparados com as redes neurais profundas. Os resultados obtidos mostraram que o classificador de regressão logística alcançou o melhor desempenho entre os algoritmos tradicionais, com 54% de acurácia média. O desempenho das redes neurais profundas foi equivalente, alcançando até 54% de acurácia média de acordo com os ajustes dos parâmetros. Este resultado é visto como promissor, uma vez que há espaço para melhora de desempenho dessas técnicas se considerarmos que um número bastante reduzido de parâmetros foi estudado. Ademais, o baixo desempenho de outras técnicas, tais como o naive Bayes, evidenciam que o número de *tweets* (publicações do Twitter) rotulados na base de dados ainda é pequeno e que o desempenho geral das técnicas pode ser melhorado pela anotação de mais deles.

Palavras-chave: aprendizado profundo, aprendizado de máquina, análise de sentimentos, redes neurais profundas, processamento de línguas naturais.

Abstract

The Internet today represents one of the leading means of communication and information sharing. In Brazil and in several other countries, the vast majority of the population uses it for a variety of activities. Examples include social networks and microblogs, environments where there are frequent manifestation of opinions and discussions about products, services or other subjects of general interest. In this context, the task of sentiment analysis may label opinions expressed in such vehicles as positive, negative or neutral, using algorithms of natural language processing (NLP) and machine learning. The objective of this study was to investigate the application of deep learning techniques to sentiment classification on a Twitter corpus, considering subjects of interest to the population. Specifically, this study considered the manifestation of opinions about the Brazilian presidential elections of 2018. Data collection was done through Tweepy, an application programming interface (API) provided by Twitter, on all days of televised debates. The database was preprocessed using NLP techniques. To perform the experiments, a relevant set of data classification algorithms was selected from the literature (naive Bayes, decision tree, logistic regression and support vector machines), and compared with deep neural networks. The results obtained showed that the logistic regression classifier achieved the best performance among traditional algorithms, with an averaged accuracy of 54%. The predictive performance of the deep neural network was equivalent, achieving the same 54% of averaged accuracy in a given parameter setting. This result is promising because there is a large space for improving the performance of such a technique as a small amount of parameters was studied. In addition, the poor performance of other techniques, such as the naive Bayes, show that the number of labeled tweets (Twitter posts) on the database is still small and that the overall performance of the classification techniques can be improved by labeling more of them.

Keywords: deep learning, machine learning, sentiment analysis, neural networks, natural language processing.

Lista de ilustrações

Figura 1 – Funcionamento de aprendizado supervisionado (adaptado de Parikh, 2018). 15
Figura 2 – Funcionamento de aprendizado não-supervisionado (adaptado de Parikh, 2018). 15
Figura 3 – Funcionamento de aprendizado reforçado (Parikh, 2018). 16
Figura 4 – Árvore de decisão gerada com base nos dados. 17
Figura 5 – Hiperplanos traçados pelo algoritmo de SVM (Lorena e Carvalho, 2007). 18
Figura 6 – Funcionamento do algoritmo de *k-fold*. (Scikit Learn Library, 2019) . . 19
Figura 7 – Esquema geral da tarefa de análise de sentimentos usando aprendizado de máquina. 22
Figura 8 – Exemplo de Tweet. (Twitter, 2018) 23
Figura 9 – Perceptron simples (Rosenblatt, 1958). 23
Figura 10 – Perceptron Multicamadas (Riedmiller e Lernen, 2014). 25
Figura 11 – Exemplo de Convolução (Krizhevsky, Sutskever e Hinton, 2012). . . . 26
Figura 12 – Exemplo de max-pooling (Krizhevsky, Sutskever e Hinton, 2012). . . . 27
Figura 13 – Anotação dos *tweets* utilizada na pesquisa do SASA. (Wang et al., 2012) 30
Figura 14 – Impacto da utilização de bigramas e trigramas na acurácia dos algoritmos (Pak e Paroubek, 2010). 31
Figura 15 – Tela inicial da Aplicação web para anotação de tweets. 34
Figura 16 – Tela da aplicação destinada à classificação dos *tweets*. 34

Lista de tabelas

Tabela 1 – Base de dados de exemplo para uma árvore de decisão. 17
Tabela 2 – Exemplos de *tweets* coletados sobre a Apple. 21
Tabela 3 – Termos buscados na API e quantidade de resultados. 32
Tabela 4 – Rótulos obtidos pela aplicação *web* 34
Tabela 5 – Palavras antes e depois de passarem pelo algoritmo de *stemming* . . . 35
Tabela 6 – Matriz bag of words dos documentos apresentados. 36
Tabela 7 – Resultados dos algoritmos tradicionais na base de dados. 40
Tabela 8 – Resultados obtidos pela rede neural profunda. 41
Tabela 9 – Resultados da rede neural profunda com e sem o pré-processamento. . 42

Lista de abreviaturas e siglas

PLN - Processamento de Línguas Naturais

RNA - Rede Neural Artificial

API - Application Programming Interface (Interface de Programação de Aplicativos)

DNN - Deep Neural Network (Rede Neural Profunda)

XOR - Exclusive-or (Ou-exclusivo)

TF - Term Frequency (Frequencia do termo)

IDF - Inverse Document Frequency (Inverso da frequencia do documento)

MLP - Multi Layer Perceptron

MHz - Mega-Hertz

GHz - Giga-Hertz

Sumário

1 INTRODUÇÃO 12
1.1 Objetivos 13
1.2 Organização do trabalho 13

2 *FUNDAMENTAÇÃO TEÓRICA* 14
2.1 Aprendizado de Máquina 14
2.1.1 Algoritmos de classificação de dados 16
2.1.1.1 *Naive Bayes* 16
2.1.1.2 Árvore de decisão 16
2.1.1.3 Máquina de vetores de suporte (SVM) 17
2.1.1.4 Regressão Logística 18
2.1.2 Métodos de validação e avaliação 18
2.1.2.1 K-fold 18
2.1.2.2 Métodos de avaliação 19
2.2 Processamento de Línguas Naturais 20
2.2.1 Análise de sentimentos 21
2.2.2 Twitter 22
2.3 Redes Neurais 23
2.3.1 Perceptron 23
2.3.2 Perceptron Multicamadas 24
2.3.3 *Feed-forward* e *back-propagation* 24
2.3.4 Aprendizado Profundo 25
2.3.4.1 Rede Neural Profunda (DNN) 26
2.3.4.2 Rede Neural Convolucional (CNN) 26
2.3.4.3 Funções de ativação 27
2.4 TensorFlow e Keras 28
2.5 Trabalhos relacionados 29

3 DESENVOLVIMENTO 32
3.1 Coleta de dados 32
3.2 Pré-processamento dos dados 32
3.2.1 Anotação dos Rótulos 33
3.2.2 Remoção de stopwords, acentuação e URLs 34
3.2.3 *Stemming* 35
3.2.4 *Bag of words* 35
3.2.5 Tf-idf 36

3.3 Deep Neural Network (DNN) 37
3.4 Dos recursos e métodos desenvolvidos 38

4 RESULTADOS 39
4.1 Algoritmos tradicionais 39
4.2 Rede Neural Profunda (DNN) 40

5 CONCLUSÃO 43

REFERÊNCIAS 45

1 Introdução

Nos últimos anos a Internet se tornou o ambiente que mais trafega informações e opiniões no mundo. No Brasil, e em diversos outros países, uma grande maioria da população está cadastrada e tem acesso às redes sociais mais populares, como o Facebook e o Twitter (Hoffman, Novak e Schlosser, 2001). Trafegam entre esses dados informações de grande riqueza para levantamento de estudos sobre assuntos de opinião popular, como política, opiniões relacionadas a empresas e mídia, entre outros.

Seria de grande importância e utilidade para uma empresa ou organização conseguir um levantamento de popularidade de seus produtos e serviços, podendo assim melhorá-los e mudar estratégias de negócio. Diante dessa necessidade, a análise de sentimentos é um tópico que se mostra cada vez mais relevante entre assuntos relacionados à Inteligência Artificial (Liu, 2012). O objetivo da análise de sentimentos é encontrar uma maneira automatizada de coletar e classificar opiniões sobre algum tema específico (Pak e Paroubek, 2010).

Ainda assim, fazer com que o computador entenda dados em línguas naturais é uma tarefa complexa que requer atenção. Esse problema ainda é acentuado pela linguagem informal e abreviada que eventualmente é utilizada nas redes sociais. Tendo isso em vista, surge a necessidade de que textos sejam pré-processados para serem entendidos (Mostafa, 2013). Para isso, podem ser aplicadas técnicas de Processamento de Línguas Naturais (PLN), um ramo da Inteligência Artificial que visa medir e diminuir os esforços que um computador precisa aplicar para entender textos em línguas naturais. Com os dados transformados de maneira mais propícia para o entendimento do algoritmo, podem ser utilizadas técnicas de aprendizado de máquina, como algoritmos de Redes Neurais Artificiais (RNA), para "ensinar" ao computador como classificar um certo *tweet* (publicação do Twitter) em sentimentos pré-definidos (Malheiros, 2014).

A utilização de redes neurais para a resolução de problemas de análise de sentimentos vem se popularizando (Pak e Paroubek; Tumasjan et al., 2010), visto que estas técnicas se mostram eficientes e apresentam resultados promissores na classificação de publicações de *microblogs* em sentimentos conhecidos (Araújo, Gonçalves e Benevenuto, 2013).

Utilizando essas técnicas, espera-se conseguir taxas de acerto que provem a viabilidade de um algoritmo para a automação da classificação de sentimentos. Tal algoritmo que possa levantar informações e dizer com grande efetividade o que as pessoas pensam sobre qualquer assunto em alta pode ser de grande valor para a população em geral.

1.1 Objetivos

Diante do exposto, o objetivo geral desse trabalho é desenvolver uma ferramenta de análise de sentimentos que consiga rotular sentenças em classes pré-estabelecidas, com foco nas publicações dos usuários do *Twitter* sobre as eleições presidenciais de 2018, automatizando assim uma pesquisa de opinião.

Seguindo a proposta do trabalho, bem como o objetivo geral, os objetivos específicos a serem alcançados são:

- Coletar e rotular a base de dados que será utilizada no trabalho, por meio da API disponibilizada pelo *Twitter*.
- Aplicar técnicas de PLN na base de dados e submetê-la a testes em diversos algoritmos de aprendizado de máquina.
- Medir a eficácia do modelo, bem como a sua viabilidade para ser utilizado em pesquisas de opinião.

1.2 Organização do trabalho

Este trabalho foi organizado em cinco capítulos. No capítulo 2, serão apresentados conceitos de técnicas e algoritmos importantes para o entendimento do estudo. No terceiro capítulo, será abordada a metodologia utilizada para a realização do trabalho, desde a coleta de dados à execução de algoritmos. No capítulo 4, serão apresentados e discutidos os resultados alcançados com as técnicas utilizadas. Por fim, no capítulo 5, será apresentada a conclusão do trabalho, bem como a discussão de passos para trabalhos futuros.

2 *Fundamentação teórica*

Neste capítulo serão apresentados e discutidos os principais conceitos, técnicas e algoritmos de aprendizado de máquina, processamento de línguas naturais e análise de sentimentos utilizados no desenvolvimento deste trabalho. Além disso, serão abordados os principais trabalhos que foram utilizados como referência.

2.1 Aprendizado de Máquina

Computadores são ferramentas muito poderosas de processamento de dados. Executam tais tarefas com maestria e agilidade, ultrapassando a eficácia de qualquer ser humano nas mesmas. O homem, desde os primórdios da computação, tenta se aproveitar desse poder criando algoritmos para auxiliá-lo a realizar tarefas, das mais simples às mais complexas. Algumas tarefas, no entanto, não possuem um algoritmo definido para serem executadas. Como exemplos simples, podemos citar a classificação de sentenças, reconhecimento facial e a separação de e-mails legítimos de spam.

O aprendizado de máquina emerge a cada dia como uma das áreas mais importantes da ciência da computação (Mohri, Rostamizadeh e Talwalkar, 2018). Partindo de modelos baseados em matemática e lógica, é possível fazer com que o computador utilize experiências passadas e aplique os conhecimentos adquiridos ao longo de seu aprendizado em novas tarefas, reproduzindo, e muitas das vezes ultrapassando, a eficácia de seres humanos. Essa área da computação abrange várias subáreas de grande relevância para o meio científico, como a análise de sentimentos.

Existem três paradigmas principais de aprendizados de máquina, definidos da seguinte maneira:

- Aprendizado supervisionado: neste tipo de aprendizado, o "supervisor" do modelo interage com a execução, fornecendo rótulos e dados de treino (Jiang et al., 2016). O modelo, então, pode mapear os dados utilizando desde funções simples até funções mais complexas. A Figura 1 mostra o funcionamento de um algoritmo de aprendizado supervisionado. São fornecidos ao algoritmo os dados de treino e os rótulos, além dos dados a serem processados. O algoritmo então processa estes dados e gera predições para os dados de teste.

- Aprendizado não-supervisionado: no aprendizado não-supervisionado, os únicos dados de entrada do modelo são a base de dados, sem rótulos. O próprio modelo analisa as informações da base para separá-la em grupos de acordo com algum critério de

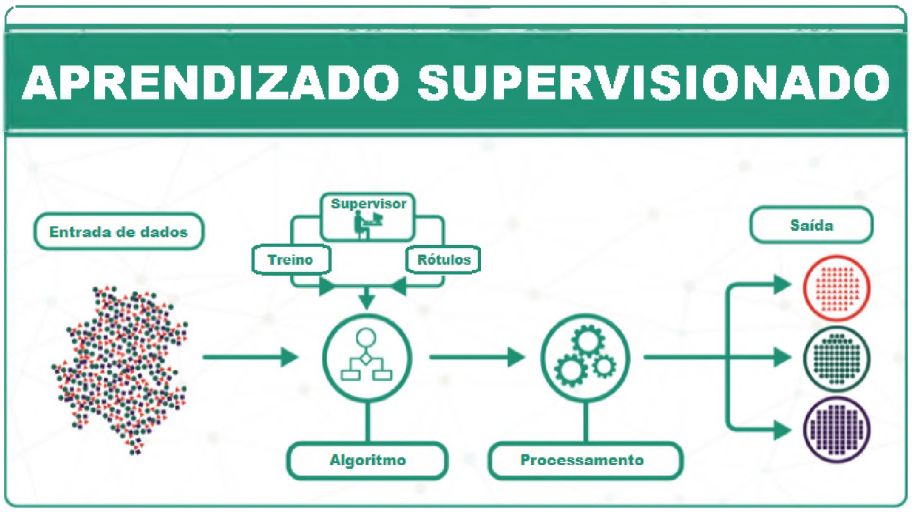

Figura 1 – Funcionamento de aprendizado supervisionado (adaptado de Parikh, 2018).

similaridade (Jiang et al., 2016). Como mostra a Figura 2, os dados de entrada não possuem rótulos e o algoritmo não recebe dados de treino. Os dados são interpretados e processados, e o algoritmo os classifica de maneira automática, desprovido de dados de treino.

Figura 2 – Funcionamento de aprendizado não-supervisionado (adaptado de Parikh, 2018).

- Aprendizagem por reforço: este paradigma de aprendizado de máquina é representado pela evolução por recompensa. Basicamente, o agente observador modela o comportamento ideal do algoritmo diante das circunstâncias a que este é submetido. Decisões corretas são recompensadas, ao mesmo tempo que as incorretas são punidas, com o intuito de melhorar a tomada de decisão do modelo (Jiang et al., 2016). Como podemos ver na Figura 3, a aprendizagem por reforço é um ciclo, composto por um lado representado pelas decisões tomadas pelo modelo, e pelo outro representado pelas recompensas e punições aplicadas pelo agente observador.

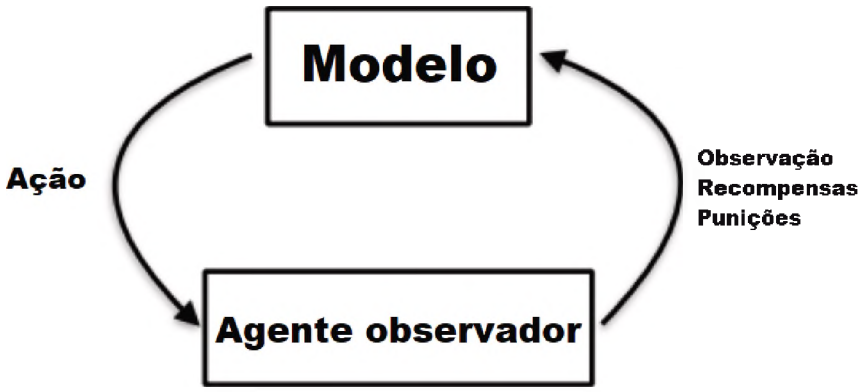

Figura 3 – Funcionamento de aprendizado reforçado (Parikh, 2018).

2.1.1 Algoritmos de classificação de dados

Para a execução dos testes deste trabalho, foram selecionados alguns algoritmos de classificação comumente utilizados em tarefas de aprendizado de máquina. Estes algoritmos serão apresentados e discutidos nesta seção.

2.1.1.1 *Naive Bayes*

Classificadores *Naive Bayes* são modelos probabilísticos que aplicam o teorema de Bayes. Estes modelos assumem que todas as características de uma base de dados são independentes, e tentam predizer a classe de um documento (entrada da base de dados) em classes pré-definidas. Sendo as classes da base definidas por:

$$C_1, C_2 ... C_n$$

o algoritmo gera a probabilidade P do documento pertencer à cada classe C. Este classificador é conhecido por ser simples, linear, mas ainda assim eficiente em várias tarefas de aprendizado de máquina (Raschka, 2014).

2.1.1.2 Árvore de decisão

Algoritmos de árvore de decisão trabalham com a estratégia de dividir e conquistar, e conseguem extrair padrões de bases de dados para definir regras de classificação, o que torna seu uso viável em várias tarefas de aprendizado de máquina (Myles et al., 2004). Os nós da árvore representam os atributos, as arestas são os possíveis valores destes atributos, e os nós-folha representam a classe de saída para aquele caminho (Shiba et al., 2005).

Um exemplo de base de dados simples para uma árvore de decisão é mostrado na Tabela 1. Os dados em questão são utilizados para decidir se uma pessoa deve ou não sair de casa, dadas algumas condições climáticas:

Tabela 1 – Base de dados de exemplo para uma árvore de decisão.

Dia	**Perspectiva**	**Umidade**	**Vento**	**Sair?**
1	Sol	Alta	Fraco	Não
2	Sol	Normal	Fraco	Sim
3	Sol	Normal	Forte	Sim
4	Nublado	Alta	Fraco	Sim
5	Nublado	Normal	Forte	Sim
6	Chuva	Alta	Forte	Não
7	Chuva	Alta	Fraco	Sim

São definidas, a partir dos atributos dos dados, várias regras de decisão. Então, durante a classificação, o algoritmo prediz a classe de dados desconhecidos seguindo o caminho de seus atributos pela árvore. De acordo com a árvore de decisão representada pela Figura 4, por exemplo, todos os documentos que contêm o valor "Nublado" para o atributo "Perspectiva" são classificados como "Sim".

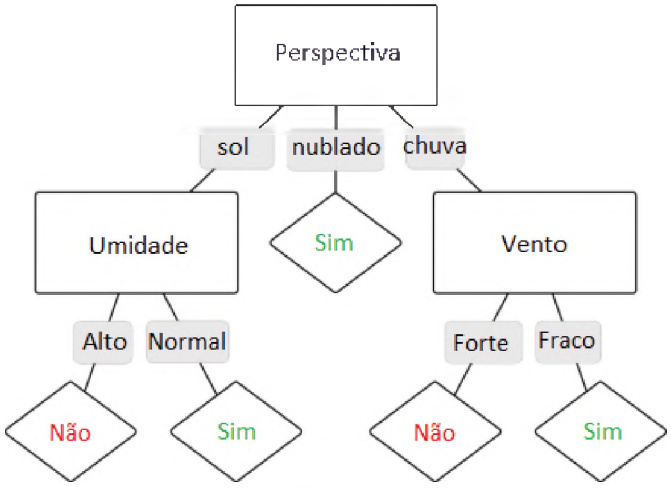

Figura 4 – Árvore de decisão gerada com base nos dados.

2.1.1.3 Máquina de vetores de suporte (SVM)

Uma Máquina de Vetores de Suporte (Carneiro et al., 2017) é um algoritmo de aprendizado supervisionado que tenta traçar um hiperplano com o intuito de dividir os documentos de uma base de dados em dois ou mais planos pré definidos (no caso as classes das sentenças). A distância entre o hiperplano e o ponto (documento) mais próximo de cada plano é chamada de margem, e o algoritmo tenta maximizar essa margem, demarcando um espaço tão bem definido quanto seja possível. A Figura 5 mostra de maneira simplificada como o algoritmo traça o hiperplano entre os dados que tem classes representadas por círculos e triângulos.

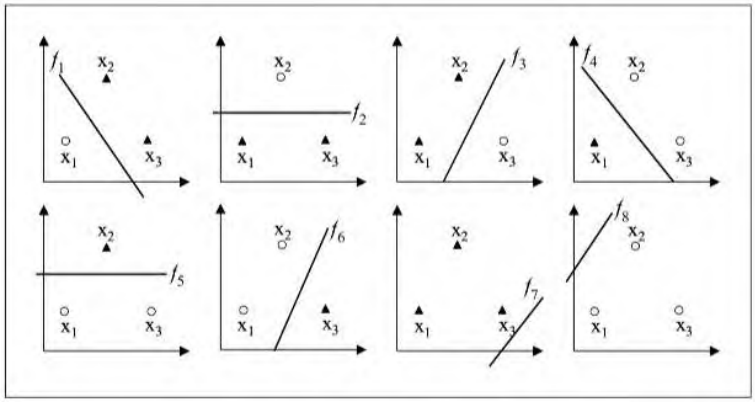

Figura 5 – Hiperplanos traçados pelo algoritmo de SVM (Lorena e Carvalho, 2007).

2.1.1.4 Regressão Logística

Modelos de regressão logística são técnicas estatísticas que modelam a relação funcional entre as características do documento, e tem como saída a probabilidade deste documento pertencer a cada uma das classes pré-definidas (Kleinbaum et al., 2002).

Este modelo gera um gráfico, onde um eixo é indicado pelo documento e o outro mostra as classes a que este documento podem pertencer. O algoritmo traça uma curva em "s" no gráfico. Essa curva é criada pela função sigmóide, que pode transformar qualquer valor real em um número entre zero e um, e é calculada pela seguinte fórmula:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

2.1.2 Métodos de validação e avaliação

2.1.2.1 K-fold

O *k-fold* é um método de validação muito utilizado no contexto de aprendizado de máquina. Consiste na divisão da base de dados em k amostras, executando o algoritmo k vezes, cada vez utilizando uma das amostras como teste e as outras como treino (Scikit Learn Library, 2019). A Figura 6 exemplifica como é feita a execução do algoritmo. As partes das barras que estão na cor vermelha representam a porção dos dados que estão sendo utilizados para teste, enquanto a azul representa o teste.

A utilização do *k-fold* é importante para evitar o sobreajuste (*overfitting*), que é quando um algoritmo se mostra muito eficiente em uma base de treino, mas ineficiente quando é submetido a dados desconhecidos.

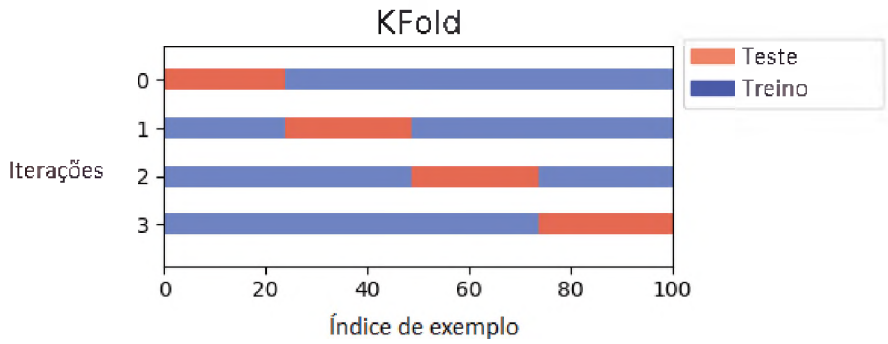

Figura 6 – Funcionamento do algoritmo de *k-fold*. (Scikit Learn Library, 2019)

2.1.2.2 Métodos de avaliação

Existem algumas métricas de avaliação que necessitam ser explicadas para o melhor entendimento do trabalho.

- Acurácia: representa a taxa de acerto do modelo (a porcentagem dos rótulos reais que o modelo conseguiu predizer) em relação aos dados de teste. É o resultado da divisão do número de predições corretas pelo total de previsões feitas.

- Matriz de confusão: esta matriz representa estatísticas sobre as classes reais e as predições do modelo. São compostas de uma linha e uma coluna para cada classe definida da base de dados, sendo as linhas indicadas pelo resultado esperado e as colunas pela saída real da rede neural.

  Assim, considerando-se um problema binário com as possíveis classes "sim" e "não", a matriz de confusão é composta por:

  - Verdadeiros positivos: casos onde o modelo prediz "sim" e a saída esperada é "sim";
  - Verdadeiros negativos: casos onde o modelo prediz "não" e a saída esperada é "não";
  - Falsos positivos: casos onde o modelo prediz "sim" e a saída esperada é "não";
  - Falsos negativos: casos onde o modelo prediz "não" e a saída esperada é "sim".

- Pontuação *F1*: medida representada pela média harmônica entre as medidas de precisão e cobertura obtidas por um classificador:

$$F1 = 2 * \frac{1}{\frac{1}{precisão} + \frac{1}{recall}},$$

  sendo que:

– precisão:

$$\frac{verdadeiros\ positivos}{(verdadeiros\ positivos + falsos\ positivos)}$$

– cobertura:

$$\frac{verdadeiros\ positivos}{(verdadeiros\ positivos + falsos\ negativos)}$$

2.2 Processamento de Línguas Naturais

A identificação, análise e classificação de objetos para um ser humano é uma tarefa fácil. Quando observamos algo, nosso cérebro instintivamente coleta características daquele objeto e o assimila a informações guardadas, podendo dizer o que é aquilo apenas por similaridade de experiências passadas.

Por outro lado, a execução de tal tarefa para um computador pode ser extremamente complexa, tendo em vista que fatores externos, como diferentes perspectivas, podem influenciar. Os computadores podem facilmente executar tarefas bem definidas em linguagens de programação (como por exemplo C, Java, Python). São linguagens estruturadas onde cada comando define uma operação a ser executada. De maneira diferente, na língua natural, cada frase pode ser ambígua, tendo vários significados diferentes dependentes de contexto.

Segundo Pereira (2012), o Processamento de Línguas Naturais pode ser definido como "o desenvolvimento de modelos computacionais para a realização de tarefas que dependem de informações expressas em uma língua natural". Martins, Kataoka e Trindade (2010) definem cinco níveis de entendimento de línguas naturais:

- Fonético: relacionado à percepção das palavras pela sua sonoridade.
- Morfológico: relacionado à forma das palavras, bem como a sua redução a unidades de significado primitivas.
- Sintático: relacionamento das palavras entre si em uma frase.
- Semântico: relacionado ao significado das palavras em uma frase de modo que a mesma seja compreensível.
- Pragmático: relacionado ao contexto em que as palavras são utilizadas, o que pode mudar totalmente seu significado.

A máquina não consegue ler e entender textos em línguas naturais como os seres humanos. Portanto, os dados devem ser transformados de maneira que possam ser utilizados no aprendizado. O processamento de texto, neste contexto, é a manipulação do conteúdo desses dados, de forma a facilitar o entendimento e aprendizado baseado nos mesmos

por parte do computador. Este processamento pode ser representado por um conjunto de operações, tendo como exemplo a tokenização (retirada das palavras-chave do texto), remoção de *stopwords* (palavras sem valor semântico, como "e", "ou", "se"), retirada de caracteres especiais e a conversão de todo o texto em letras minúsculas.

2.2.1 Análise de sentimentos

A análise de sentimentos em texto consiste no processamento dos mesmos para a polarização de suas sentenças em sentimentos conhecidos. Na grande maioria das vezes essa atividade utiliza o Processamento de Línguas Naturais, juntamente com técnicas de aprendizado de máquina supervisionadas (Malheiros, 2014). Informalmente, tal tarefa consiste em detectar a polaridade das mensagens em função da opinião dos usuários, isto é, classificar cada mensagem como positiva ou negativa. Um exemplo ilustrativo do problema são *tweets* capturados sobre a empresa *Apple*, os quais refletem diferentes opiniões dos usuários sobre tecnologias, produtos e serviços oferecidos pela empresa. A Tabela 2 apresenta quatro destas mensagens bem como a polaridade (classe) associada a cada uma delas.

Tabela 2 – Exemplos de *tweets* coletados sobre a Apple.

Message	**Class**
#Siri now knows who my dad, mom, brother and girlfriend is. Thanks @apple	+
Well at least the @apple store has amazing call waiting music! #need4s	+
none of my apps work after the new ios from @apple. what do i do?!?	-
Hey @apple, the SMS full message is complete shit. Yes, I'm annoyed.	-

A Figura 7 apresenta um esquema geral da tarefa de análise de sentimentos (AS). Na etapa de treino, um conjunto de *tweets* cujos sentimentos já estão classificados é usado para treinar um algoritmo de aprendizado. Na etapa de teste, *tweets* cujos sentimentos são desconhecidos precisam ser preditos pelo classificador. Note que os métodos de AS detectam sentimentos baseados em um conjunto de atributos. Logo, a seleção e combinação destes atributos possui contribuição importante no desempenho da tarefa. No contexto do Twitter, há quatro grupos principais de atributos: semântico, sintático, estilístico e do próprio Twitter. Os atributos semânticos incluem termos que revelam sentimentos positivos ou negativos a partir de seu conceito semântico; atributos sintáticos incluem a análise morfológica e sintática; atributos estilísticos estão relacionados ao estilo de escrita usado em mídias sociais; e atributos próprios do Twitter incluem termos e símbolos específicos como *retweets* ou *hashtags* [Giachanou e Crestani 2016].

Essa área está sendo cada vez mais estudada e tem aplicações diversas. Para levantar pesquisas sobre o que os clientes de uma empresa estão comentando sobre seus

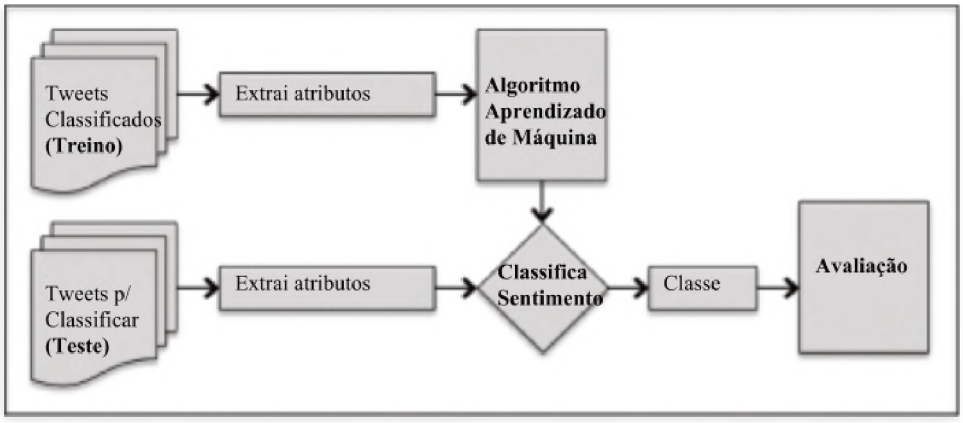

Figura 7 – Esquema geral da tarefa de análise de sentimentos usando aprendizado de máquina.

produtos, ou como está a popularidade de um político diante de seus eleitores, podem ser gastos vários dias de trabalho manual, e esse trabalho acaba não sendo eficiente. Treinar um algoritmo para rotular dados de textos de opinião pode ser muito importante nesse sentido, possibilitando o levantamento desse tipo de informação de maneira automática, eficiente e com boa precisão nos resultados.

Grandes desafios à análise de sentimentos podem ser facilmente observados. Textos com palavras similares podem ter significados totalmente diferentes quando seus contextos são diferentes. A linguagem informal utilizada nas redes sociais, bem como o uso de abreviações, ironias e emoticons devem ser tratados para que a tarefa de entendimento da língua natural por parte do computador possa ser bem executada (Benevenuto, Ribeiro e Araújo, 2015).

2.2.2 Twitter

O Twitter é uma rede social criada em 2006 que possibilita a curta troca de publicações entre usuários. Atualmente, é uma das redes sociais mais populares do planeta, tendo mais de 300 milhões de usuários mensais (Statista, 2019). Os usuários da rede usualmente publicam textos de opiniões (tweets) de até 140 caracteres (Figura 8) sobre os mais diversos assuntos, desde entretenimento até política. Também é possível marcar como favoritos, compartilhar e responder a tweets de outros usuários.

Para a extração da base de dados utilizada neste trabalho, foi utilizada a API disponibilizada pelo Twitter, integrada à linguagem Python, o Tweepy. Essa API possibilita a busca de tweets por palavras-chave, data de publicação, bem como vários outros parâmetros.

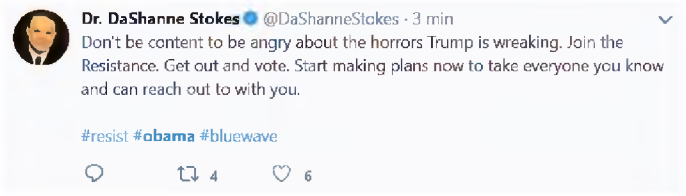

Figura 8 – Exemplo de Tweet. (Twitter, 2018)

2.3 Redes Neurais

Uma rede neural artificial é um modelo matemático que tenta funcionar de maneira semelhante a um sistema neural biológico. Neurônios são células do tecido nervoso dos seres vivos e funcionam, basicamente, da seguinte maneira: são ativados por impulsos elétricos, e trocam informações com outros neurônios sempre que são ativados. Tentando reproduzir artificialmente esse sistema, uma rede neural artificial simula vários neurônios trabalhando em conjunto para classificar ou predizer um dado ou informação (Rosenblatt, 1958).

2.3.1 Perceptron

Frank Rosenblatt (Rosenblatt, 1958) apresentou um modelo de rede neural artificial, denominado Perceptron, baseado no neurônio biológico. O Perceptron contêm dados de entrada e dados de saída do modelo. Cada nó é conhecido como neurônio. As entradas de cada nó são representadas por dados e pesos que se ajustam até que o modelo consiga mostrar um desempenho apropriado em certa tarefa. Cada perceptron em um modelo calcula sua saída fazendo um somatório das entradas multiplicadas por seus respectivos pesos, passando então esse resultado por uma função de ativação (Figura 9).

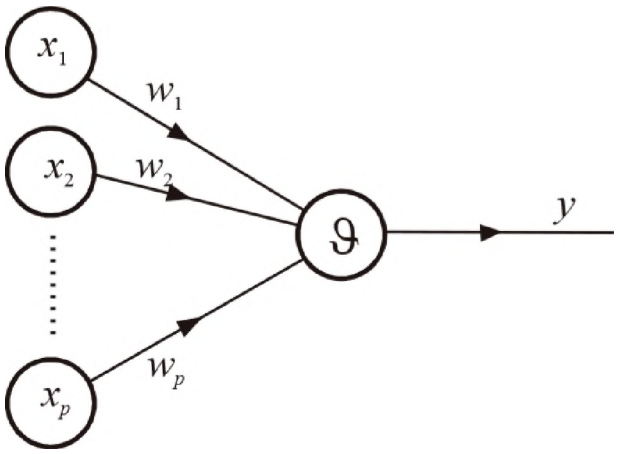

Figura 9 – Perceptron simples (Rosenblatt, 1958).

Como o Perceptron só podia resolver problemas lineares, não podendo resolver,

por exemplo, um ou-exclusivo (XOR), vários pesquisadores desencorajaram as pesquisas em redes neurais afirmando que essa seria uma limitação que não poderia ser resolvida de nenhuma maneira, utilizando o poder computacional da época. Somente nos anos 80 as redes neurais começaram a ganhar força novamente, quando foi introduzido o conceito de *back-propagation*, que mostrava que redes neurais multicamadas conseguiriam aprender a resolver problemas não-lineares.

Os dados utilizados para o aprendizado são divididos em treino, validação e teste. Os dados de treino, geralmente a maior parte dos dados, são utilizados para ajustar os pesos da rede neural de forma que ela possa predizer a classe de sentenças semelhantes. Os dados de validação servem para ajustar a eficácia do modelo. Por fim, os dados de teste, são usados para avaliar o quanto a rede conseguiu generalizar o que "aprendeu" com os dados de treino.

2.3.2 Perceptron Multicamadas

Esse modelo tem um funcionamento mais complexo do que um simples Perceptron. Nele, existem várias camadas entre as de entrada e saída, chamadas de camadas ocultas, cada uma representada por vários neurônios. Cada neurônio da camada n tem seu valor computado de saída direcionado à entrada de cada neurônio da camada $n+1$ e assim por diante.

A quantidade e valores ótimos de camadas, neurônios, e parâmetros de um perceptron multicamadas são diferentes em cada caso, levando em consideração a complexidade do problema a ser resolvido. Riedmiller e Lernen (2014) definem uma rede neural multicamada como um grafo finito acíclico, onde os nós são neurônios com funções de ativação (Figura 10). A entrada de cada neurônio é calculada de acordo com um combinador linear, que basicamente é representado pelo somatório dos produtos das entradas e seus respectivos pesos. O valor resultante dessa combinação é submetido a uma função de ativação, gerando a saída do neurônio.

De acordo com Gardner e Dorling (1998), modelos matemáticos eficientes podem ser gerados utilizando-se de um bom conhecimento teórico do problema, juntamente com uma base de dados adequada. Ainda ressaltam que o modelo de perceptron multicamadas tem se mostrado mais eficiente do que a maioria das técnicas estatísticas mais tradicionais, pelo fato deste modelo conseguir modelar dados mal distribuídos e resolver funções não-lineares.

2.3.3 *Feed-forward* e *back-propagation*

O *feed-forward* representa o fluxo dos dados de um Perceptron multicamadas, e é assim chamado devido ao fato de cada camada da rede neural ser diretamente conectada à sua sucessora, sem caminho de volta. Ou seja, os dados que entram na rede são processados

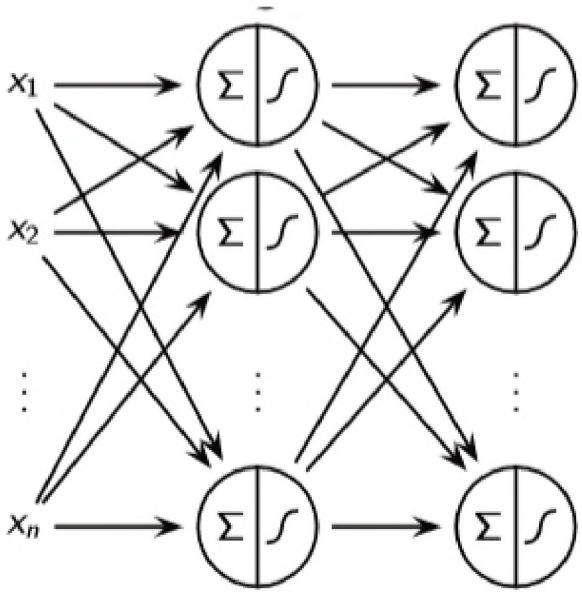

Figura 10 – Perceptron Multicamadas (Riedmiller e Lernen, 2014).

em caminho direto em direção à saída.

O *back-propagation* é um método utilizado para treinar redes neurais com eficácia. O objetivo principal da técnica é ajustar os pesos da rede de forma que o erro seja minimizado. A cada época, o algoritmo de *back-propagation* calcula como os pesos das camadas anteriores devem ser ajustados com o intuito de atingir os resultados esperados. É importante ressaltar que as funções de ativação de neurônios precisam ser deriváveis para que o *back-propagation* possa ser executado (Vogl et al., 1988).

2.3.4 Aprendizado Profundo

O conceito de aprendizado profundo se popularizou com o avanço do poder computacional para executar redes neurais que podem ter várias camadas ocultas e neurônios. Cada uma destas camadas aumenta o nível de abstração de características que a rede pode executar na base de dados pelo fato de incorporar o conhecimento das camadas anteriores, tornando a rede capaz de trabalhar com dados multi-dimensionais com vários níveis de representação (como imagens, áudios e texto).

O aprendizado profundo permite, basicamente, que o algoritmo "aprenda" conceitos muito complicados partindo de conceitos mais simples. Esse ramo do aprendizado de máquina vem se popularizando e sendo amplamente utilizado em trabalhos que tem uma proposta mais complexa e utilizam bases de dados não-estruturadas, como reconhecimento de voz (Hinton et al., 2012), reconhecimento facial (Sun et al., 2014), e a própria análise de sentimentos [Santos e Gatti 2014], [Severyn e Moschitti 2015], que é a proposta deste trabalho.

2.3.4.1 Rede Neural Profunda (DNN)

Uma rede neural profunda é uma rede neural artificial *feed-forward*, que possui pelo menos uma camada de processamento (camada oculta) entre as camadas de entrada e saída (Hinton et al., 2012).

É capaz de processar dados de maneiras complexas, e o aumento do número de suas camadas também aumenta a complexidade do mapeamento das características. Cada camada representa um nível diferente de transformação e abstração de informações, agregando e combinando características das camadas anteriores. Esta característica é chamada de *feature hierarchy* (hierarquia de características).

Diferentemente de outros algoritmos de aprendizado de máquina que mostram estagnação quando são alimentados com uma quantidade muito grande de dados, uma característica importante da DNN é que sua acurácia tende a aumentar cada vez mais de acordo com a quantidade de dados que são inseridos na rede (Ng, Andrew Yan-Tak, 2017).

2.3.4.2 Rede Neural Convolucional (CNN)

As redes neurais convolucionais conseguem identificar e diferenciar aspectos dos dados de entrada, fazendo o mapeamento e transformação espacial dos mesmos. São muito utilizadas no processamento de imagens. Diferentemente de redes neurais simples, que tem vetores de atributos como entrada, a CNN trata imagens como atributos multidimensionais, sendo consideradas a altura, largura e a profundidade, que é a representação dos canais de cores (Krizhevsky, Sutskever e Hinton, 2012).

Para representar as imagens de entrada em um vetor simples (por exemplo, uma imagem de tamanho 32×32 com 3 canais de cores formaria um vetor de 3072 posições), o custo computacional seria muito alto. A CNN opera, então, com o intuito de reduzir estas dimensões. O processo é chamado Convolução, e está representado na Figura 11:

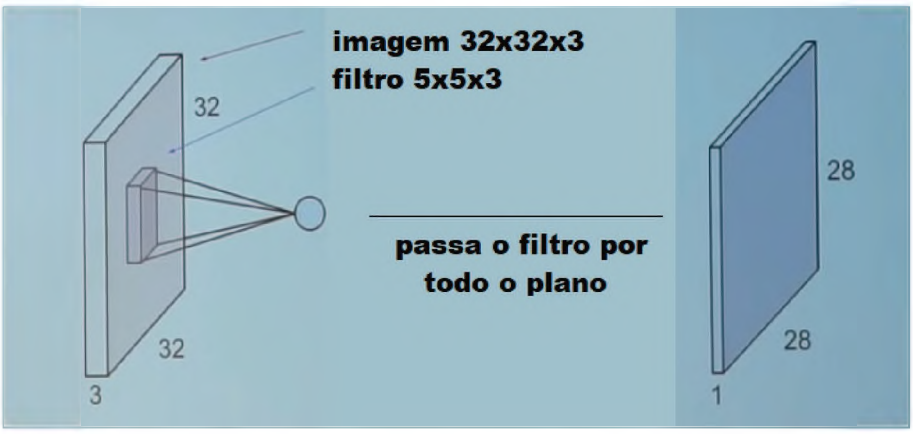

Figura 11 – Exemplo de Convolução (Krizhevsky, Sutskever e Hinton, 2012).

A rede aplica um filtro de dimensões menores e gera um mapa de características

menor que os dados de entrada originais. No exemplo da Figura 11, o filtro tem tamanho de $5 \times 5 \times 3$, e quando é aplicado na imagem de tamanho $32 \times 32 \times 3$ gera um mapa de características de tamanho $28 \times 28 \times 1$.

As camadas de convolução da CNN tem, cada uma, seu próprio conjunto de filtros para a transformação dos dados, com o poder de identificar novas características do modelo para treino. Os filtros são inicializados randomicamente e alterados com o intuito de encontrar uma representação espacial ótima dos dados da rede.

A CNN ainda tem as camadas de *pooling*, que visam diminuir de maneira escalar a quantidade de dados dos mapas de características. Uma técnica muito utilizada é o *max pooling*, que divide os mapas em n partes e geram novos mapas que contém n características, mantendo sempre o maior valor em cada área do mapa anterior, como mostra a Figura 12:

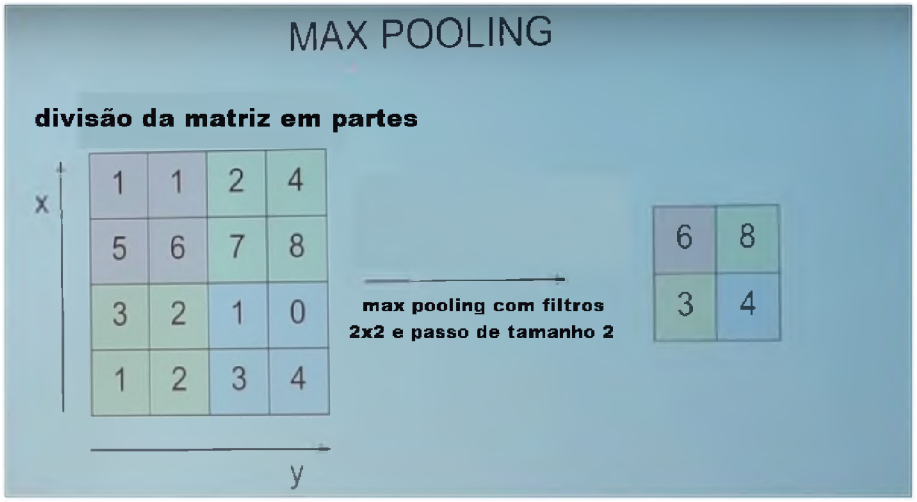

Figura 12 – Exemplo de max-pooling (Krizhevsky, Sutskever e Hinton, 2012).

As camadas de convolução e *pooling* da CNN não são fortemente conectadas às camadas anteriores, diferentemente da camada de saída da rede, Esse tipo de rede neural opera, então, visando automatizar o processo de extração de características das imagens.

2.3.4.3 Funções de ativação

Os dados de entrada de cada neurônio da rede neural precisam ser submetidos a uma função de ativação para a geração de uma saída. Assim, funções de ativação são obrigatórias para o desenvolvimento de tais redes. As duas funções de ativações utilizadas neste trabalho foram a Unidade Linear Retificada (ReLU) e a Sigmóide, que serão explicadas a seguir.

- Unidade linear retificada: a unidade linear retificada, ou ReLU, é a função de ativação mais utilizada em redes neurais (Li e Yuan, 2017), sendo tomada como padrão para algoritmos de várias bibliotecas. Esta função é maticamente simples, e

foi utilizada neste trabalho nas camadas de entrada e processamento da rede neural. É representada pela fórmula:

$$f(x) = max(0, x)$$

- Sigmóide: a função de ativação sigmóide representa uma distribuição estatística que tem como saída um valor entre 0 e 1. Esta função é amplamente utilizada em pesquisas de aprendizado de máquina (Marreiros et al., 2008). Neste trabalho, ela foi utilizada na camada de saída da rede neural. A fórmula da função sigmóide é a seguinte:

$$h_\theta(x) = \frac{1}{1 + e^{-}\theta^T x}$$

2.4 TensorFlow e Keras

O TensorFlow é um pacote de código aberto criado e disponibilizado pela Google para atuação na área do aprendizado profundo. Essa biblioteca disponibiliza várias técnicas de pré-processamento, além de algoritmos de aprendizado de máquina, medidas de avaliação e uma documentação didática sobre todos esses recursos. É uma ferramenta multiplataforma que pode ser utilizada em diversos dispositivos, tem uma comunidade de colaboradores ampla e ativa, e é melhorada constantemente.

O Keras é um pacote de aprendizado de máquina para a linguagem Python, que abrange algoritmos de vários outros pacotes, incluindo o TensorFlow. Ele representa uma *interface* para uso dos recursos das mesmas, permitindo a execução e configuração dos algoritmos de maneira mais simplificada, e executando "por trás dos panos" as configurações mais complicadas. De acordo com o autor da biblioteca, Francois Chollet, "a mesma foi desenvolvida com o intuito de experimentação rápida. A possibilidade de ir da ideia ao resultado em tempo mínimo é a chave para o sucesso da pesquisa".

O algoritmo do TensorFlow utilizado neste trabalho foi o DNNClassifier (*deep neural network classifier*, ou classificador de rede neural profunda). É um algoritmo de uso simples e totalmente configurável, podendo ter seus parâmetros ajustados para trabalhar como uma rede neural simples ou densa, dependendo da necessidade da pesquisa (Han et al., 2018). A variação do DNNClassifier apresentada pelo Keras é chamada de modelo sequencial, e tem uma execução mais simples e intuitiva.

2.5 Trabalhos relacionados

Yuri Malheiros (Malheiros, 2014), em sua ferramenta *web* Emotte, utiliza técnicas de aprendizado de máquina supervisionado e processamento de línguas naturais para classificar, em tempo real, *tweets* relacionados a diversos temas. Os *tweets* são classificados em uma das duas categorias: positivo ou negativo. O Emotte ainda consegue comparar resultados de pesquisas relacionados a dois temas, em tempo real, durante um período de tempo especificado. A ferramenta é dividida em três componentes, sendo o primeiro utilizado para a obtenção dos *tweets* pela API oficial fornecida pelo Twitter. O segundo componente utiliza os *tweets* recuperados anteriormente como entrada para técnicas de aprendizado de máquina, executando o pré-processamento (remoção de espaços, conversão do texto para caracteres minúsculos, tokenização) e sua classificação em uma das duas classes citadas anteriormente. O terceiro componente da aplicação é representado pela *interface* de comunicação com o usuário. A abordagem do Emotte é bem semelhante ao propósito desse trabalho, se mostrando mais abrangente no sentido de poder comparar resultados de várias pesquisas em tempo real.

A ferramenta iFeel, de Matheus Araújo e outros (Araújo et al., 2014), disponibilizada gratuitamente na *web*, é capaz de classificar sentimentos de textos inseridos em diferentes linguagens, como positivo, negativo, ou neutro, utilizando sete ferramentas de análise de sentimentos. Cada uma destas ferramentas tem abordagens diferentes, ainda que semelhantes. A ferramenta PANAS-t (*positive affect negative affect scale*) define onze sentimentos expressados em diferentes sentenças. Os pesquisadores utilizaram estes sentimentos para dividir a base de dados entre sentenças positivas negativas. Utilizaram também uma abordagem de emoticons, que pode ser interessante para pesquisas baseadas em texto. Apesar de todas as abordagens serem totalmente voltadas para *corpus* da língua inglesa, algumas linhas de pensamento podem ser interessantes para a realização deste trabalho.

O SailAIl Sentiment Analyser, chamado de "SASA" (Wang et al., 2012) foi utilizado nas eleições presidenciais dos Estados Unidos de 2012, coletando dados do Twitter relacionados a essas eleições por um provedor comercial da rede social, chamado *Gnip Power Track*. Após coletados os dados, os *tweets* foram pré-processados. O SASA utiliza o código de Christopher Potts (Potts, 2011), disponibilizado gratuitamente na *web* para realizar a tokenização dos tweets. Os autores utilizaram a *Amazon Mechanical Turk* (Amazon, 2005) para classificação da base de dados existente entre positiva, negativa ou neutra. A ferramenta utiliza um classificador Naive-Bayes para treino e teste da base de dados. Os métodos de tokenização e classificação da base de dados utilizada por Wang podem ser utilizados como referência neste trabalho, tendo em vista que a abordagem é muito semelhante ao nosso objetivo. Esse trabalho ainda apresentou uma ideia de aplicação de anotação semelhante à utilizada no nosso trabalho, como mostra a Figura 13.

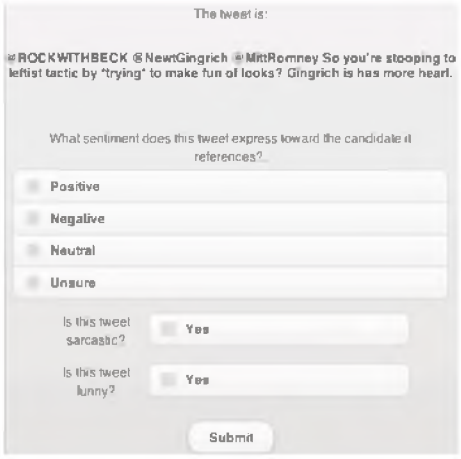

Figura 13 – Anotação dos *tweets* utilizada na pesquisa do SASA. (Wang et al., 2012)

Adrian Tumasjan et. al (Tumasjan et al., 2010) utilizaram uma base de dados de mais de cem mil *tweets* relacionados à eleição do parlamento nacional da Alemanha de 2009. Os tweets foram buscados utilizando como palavras-chave os nomes das seis chaves participantes da eleição, além de nomes dos políticos relacionados a estas chaves. Foram adquiridos 70 mil tweets relacionados às chaves e 35 mil relacionados aos políticos. Os autores utilizaram o *Linguistic Inquiry and Word Count*, (Pennebaker, Francis e Booth, 2001) uma ferramenta desenvolvida para classificação de textos com base em fatores emocionais, cognitivos e estruturais do texto dos *tweets*. A base de dados foi adquirida na Alemanha e traduzida para o inglês para a sua utilização na ferramenta. O principal objetivo deste trabalho era predizer o resultado das eleições, separando os tweets em diferentes emoções, como raiva, ansiedade, tristeza, certeza, emoções positivas e negativas, entre outras.

No trabalho de Alexander Pak e Patrick Paroubek (Pak e Paroubek, 2010) é utilizada uma base de dados retirada pela API oficial do Twitter, utilizando como palavras-chave para a busca os emotes, símbolos que representam faces, como ":-)" ou ";(". Nessa abordagem, um *tweet* com um emote feliz é considerado como positivo, e um *tweet* com emote triste é considerado como negativo. O corpo das mensagens passa por técnicas de pré-processamento, como tokenização e remoção de stopwords. Foram então utilizados três classificadores, e o que trouxe melhores resultados foi o Naive-Bayes. A abordagem de classificação de tweets por emotes pode ser eficaz na maioria dos casos, mas pode gerar ambiguidade em *tweets* irônicos, por exemplo. Este trabalho traz ainda algumas técnicas que podem ser utilizadas para melhorar a acurácia de algoritmos de aprendizado de máquina, como a utilização de *n-grams* (visualização de tokens como conjuntos de termos). Podemos perceber, na Figura 14, que a utilização de bigramas melhorou significativamente a acurácia dos algoritmos.

Yoon Kim (Kim, 2014) propõe um modelo de classificação de sentimentos utilizando

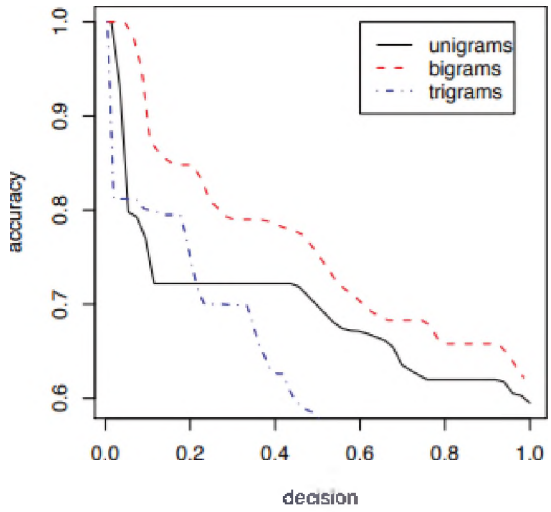

Figura 14 – Impacto da utilização de bigramas e trigramas na acurácia dos algoritmos (Pak e Paroubek, 2010).

aprendizado profundo. Em seu trabalho, ele treina uma rede neural convolucional utilizando como base de dados vetores de palavras treinados sobre 100 bilhões de palavras do site *Google News*, vetores estes que estão disponíveis publicamente. Cada palavra em uma sentença é representada por um vetor multidimensional, e cada sentença é representada pela concatenação desses vetores. O modelo foi testado em várias bases de dados, De maneira geral, Kim deixa claro que uma rede neural convolucional com apenas uma camada de convolução pode mostrar resultados significativos, além de mostrar que vetores pré-treinados de maneira não-supervisionada podem ajudar bastante nos resultados quando se usa aprendizado profundo e processamento de línguas naturais. Este trabalho é uma base interessante para o nosso, pois também estamos transformando os nossos dados em vetores de atributos.

Henrico Brum e Maria Nunes (Brum e Nunes, 2017) desenvolveram uma pesquisa que apresenta uma abordagem semelhante à deste trabalho. Utilizaram uma API para coletar sua base de dados também do Twitter, durante todo o primeiro semestre de 2017. Para anotação, disponibilizaram uma ferramenta para os colaboradores com um *cookbook* (orientações sobre o processo de anotação), apresentaram definições das classes positiva, negativa e neutra, e submeteram a base de dados rotulada a dois algoritmos de aprendizado de máquina, mais especificamente Naive-Bayes e máquina de vetores de suporte (SVM). Os autores acrescentam que a adição da classe neutra na anotação dificulta o aprendizado dos modelos, e propõem para trabalhos futuros a utilização de métodos de aprendizado não-supervisionado.

3 Desenvolvimento

A classificação de sentimentos em *microblogs* pode se mostrar como uma tarefa bastante complexa e nada trivial. Há vários trabalhos na literatura que abordam esse tipo de problema, e várias técnicas podem ser utilizadas para a progressão de uma pesquisa nesta área. Neste capítulo serão apresentados todos os recursos e métodos desenvolvidos para a coleta de dados, pré-processamento e execução dos algoritmos.

3.1 Coleta de dados

O foco dessa etapa do trabalho é a construção da base de dados que será processada e trabalhada. No contexto deste trabalho, onde o alvo do estudo é uma rede social, foi utilizada uma API chamada Tweepy, que é disponibilizada de maneira restrita para usuários comuns pelo próprio Twitter, e permite a extração de publicações buscando por vários parâmetros, como palavras-chave, usuários, data, número de compartilhamentos, etc. Essa API possui várias limitações, como o número de *tweets* que podem ser buscados a cada execução, e o tempo que deve ser esperado para executar a busca novamente.

Devido ao grande número de publicações relacionadas ao tema, optou-se por filtrar a busca das publicações relacionadas à campanha presidencial em relação aos debates presidenciais das eleições de 2018. As próprias emissoras, ao transmitir o debate, orientaram a população a utilizar *hashtags* (palavras chaves antecedidas pelo sinal de cerquilha, que tem o intuito de identificar o tema ou assunto do qual se está falando). Usando esse fato à favor da pesquisa, estas palavras-chave fornecidas pelas emissoras foram utilizadas como parâmetro para a coleta dos dados. Foram buscados, no dia de cada debate, *tweets* relacionados às *hashtags* mostradas na Tabela 3.

Tabela 3 – Termos buscados na API e quantidade de resultados.

Hashtag buscada	**Quantidade de tweets**
#DebateNaBand	21347
#DebateNaGlobo	18545
#DebateNoSBT	40000
#DebateRedeTV	20412

3.2 Pré-processamento dos dados

Retirar o significado semântico das palavras quando se lê uma frase é uma tarefa comum para seres humanos. Por outro lado, é uma tarefa muito complexa para o compu-

tador. Quando se utiliza uma base de dados tão informal como publicações de uma rede social, essa tarefa se torna ainda mais difícil. Abreviações, erros de ortografia e ironias são obstáculos a serem tratados no processamento dos dados (Brum e Nunes, 2017).

Assim sendo, é necessário transformar os dados de maneira que fiquem mais acessíveis para serem entendidos pela máquina. No processamento de línguas naturais existem várias técnicas que podem ser utilizadas para atingir este objetivo. As técnicas de pré-processamento utilizadas neste trabalho serão explicadas nas próximas subseções.

3.2.1 Anotação dos Rótulos

Algoritmos de aprendizado supervisionado necessitam de uma base de dados que será utilizada para fins de treinamento. Para alimentar esses algoritmos, então, é necessário que se tenha uma base de dados pré-rotulada. Textos em línguas naturais são de interpretação difícil e muitas das vezes confusa, até mesmo para seres humanos. Utilizando-se ainda de uma base de dados vinda de uma rede social, são encontrados diversos problemas na anotação das sentenças. Os usuários utilizam sentenças irônicas, bipolares e até mesmo sentenças que muitas das vezes não expressam opiniões ou não têm valor semântico.

Devido a estes problemas enfrentados, é necessário que se encontre uma maneira mais confiável de rotular os dados. Ter apenas uma pessoa no encargo dessa atividade pode resultar, por exemplo, em erros de interpretação. Um bom critério de confiabilidade é ter mais de uma pessoa anotando uma mesma sentença igualmente (Brum e Nunes, 2017).

A alternativa encontrada para contornar este problema foi o desenvolvimento de uma aplicação *web* para anotação dos tweets. A aplicação foi desenvolvida utilizando-se da linguagem de programação *PHP*, mais especificamente do framework *Laravel*, e um banco de dados *MySQL* para armazenamento das sentenças e seus respectivos rótulos. O banco de dados foi alimentado com 2400 tweets sobre as eleições de 2018.

Esta ferramenta conta com duas telas, sendo a primeira uma tela de apresentação que contém orientações para os colaboradores, bem como definições do que é uma sentença positiva, negativa ou neutra. A Figura 15 apresenta uma imagem da tela inicial do sistema de anotação.

Após ler as orientações e clicar no botão “classificar”, o colaborador é redirecionado para a segunda tela, onde efetivamente poderá classificar as sentenças como positivas, negativas, ou neutras. As sentenças mostradas ao colaborador são somente as que ainda não atingiram o critério de confiabilidade (dois votos em positivo, dois votos em negativo ou dois votos em neutro). O colaborador pode classificar as sentenças enquanto quiser, ou voltar para a página inicial. A Figura 16 apresenta uma imagem relacionada ao processo de anotação de um *tweet*.

Essa ferramenta foi publicada online e divulgada nas redes sociais, bem como pela

Figura 15 – Tela inicial da Aplicação web para anotação de tweets.

Figura 16 – Tela da aplicação destinada à classificação dos *tweets*.

coordenação do curso de Sistemas de Informação da UFU - Monte Carmelo, a todos os alunos e professores. Os rótulos obtidos estão representados na Tabela 4.

Tabela 4 – Rótulos obtidos pela aplicação *web*

Classe	**Quantidade**
Positivo	317
Neutro	540
Negativo	422
Total anotado	1279

É importante ressaltar que dentre os 2400 tweets que foram inicialmente inseridos no banco de dados, 600 (25%) tiveram votos destoantes entre os colaboradores. Este fato observado reforça o problema supracitado da interpretação das sentenças e evidencia o alto nível de desafio do problema com que estamos lidando.

3.2.2 Remoção de stopwords, acentuação e URLs

Para melhorar a legibilidade da base de dados, foi proposta uma remoção da acentuação dos dados, tendo em vista que uma palavra que apresenta ou não a devida acentuação deve ser tratada como o mesmo termo. Também é necessária a remoção de

URLs, que são *links* que direcionam a outras páginas, e *stopwords*, que são palavras que não acrescentam sentido semântico a uma frase. Alguns exemplos de *stopwords*, no português, são "para", "em", "no" e "do".

3.2.3 *Stemming*

O *stemming*(Frakes, 1992) é uma técnica utilizada para diminuir o tamanho do dicionário (conjunto de termos considerados no aprendizado), reduzindo palavras aos seus radicais e ocasionando no agrupamento de palavras que tem sentidos semânticos iguais, mas morfologia diferente. Assim, o estudo da base de dados é facilitado. A Tabela 5 mostra a entrada e saída de algumas palavras por um algoritmo de stemming:

Tabela 5 – Palavras antes e depois de passarem pelo algoritmo de *stemming*

Antes do *stemming*	**Após o *stemming***
copiar	copi
copiando	copi
engenheiro	engenh
engenharia	engenh

3.2.4 *Bag of words*

O modelo *bag of words* (Zhang, Jin e Zhou, 2010) é um dos modelos mais famosos utilizados para representação de dados, quando se trabalha com texto. O objetivo deste modelo é transformar os dados em uma matriz de inteiros que pode ser facilmente entendida pelo computador.

Esta matriz é representada por uma linha para cada documento no dicionário (no contexto deste trabalho, os *tweets*), e uma coluna para cada termo distinto no dicionário (palavras diferentes encontradas em toda a base de dados). Sendo X a matriz, i as linhas e j as colunas, temos:

- $X^{i,j} = 0$, se o termo j não estiver presente no documento i;
- $X^{i,j} = k$, se o termo j aparece k vezes no documento i.

Transformando os dados desta maneira, cada documento da base de dados passará a ser representado por um vetor de inteiros, indicando a presença ou ausência de cada termo (Tabela 3). Acredita-se que os dados nessa forma podem ser entendidos e utilizados pela maioria dos algoritmos de aprendizado de máquina. Levando em consideração, então, os documentos:

d^1 [candidato, legal, eu, legal]

d^2 [eu, legal, respondeu, claramente, candidato]

d^3 [legal, claramente]

Teríamos os seguintes termos:

t^1 candidato

t^2 legal

t^3 eu

t^4 respondeu

t^5 claramente

A Tabela 6 mostra a matriz *bag of words* que seria gerada pelos dados apresentados.

Tabela 6 – Matriz bag of words dos documentos apresentados.

·	t^1	t^2	t^3	t^4	t^5
d^1	1	2	1	0	0
d^2	1	1	1	1	1
d^3	0	1	0	0	1

3.2.5 Tf-idf

A atribuição de valor Tf-idf (*term frequency inverse document frequency*, Ramos et al., 2003) é uma técnica amplamente utilizada em mineração de dados que atribui valores a cada termo, indicando assim sua importância. A frequência de termos (TF) é um fator importante para determinar similaridade entre documentos; quanto mais vezes um termo aparece em um documento, maior o seu peso relativo a este documento.

No entanto, palavras que possuem pouco valor semântico podem aparecer várias vezes nos documentos, retirando assim a ênfase de termos mais importantes. Para evitar isso, o inverso da frequência dos termos nos documentos (IDF) diminui o valor de um termo de maneira inversa à quantidade de vezes que esse termo se repete em toda a base de dados, dando mais ênfase para termos mais raros que podem ter grande valor semântico.

O valor do termo w no documento d em um corpus D é dado pela seguinte equação (Ramos et al., 2003):

$$wd = fw, d * log(|D|/fw, D) \tag{3.1}$$

em que fw, d representa o número de vezes que o termo w aparece no documento d, $|D|$ representa o tamanho do *corpus* (número de documentos) e fw, D é o número de documentos em D que contêm o termo w.

A aplicação de normalizações por meio de medidas de relevância nos termos pode aumentar a eficácia de algoritmos de aprendizado de máquina. Após a base de dados passar por essa normalização, cada termo recebe um valor diretamente ligado à sua importância para a classificação (Trstenjak, Mikac e Donko, 2014), facilitando o aprendizado.

3.3 Deep Neural Network (DNN)

Foi realizada, após o pré-processamento dos dados, uma execução sistemática de algoritmos de aprendizado de máquina na base de dados. É importante ressaltar alguns parâmetros e métodos escolhidos e configurados na execução dos algoritmos, tendo em vista que um algoritmo bem configurado gera melhores resultados. Os parâmetros considerados nos experimentos foram:

- Quantidade máxima de características: é importante definir quando se deve ou não limitar a quantidade máxima de características da matriz bag of words, por exemplo. Definindo este valor como k, a matriz bag of words terá k colunas, uma para cada um dos termos mais frequentes. Quando este valor é muito pequeno, o algoritmo pode perder informações valiosas para o aprendizado, bem como o valor definido muito alto pode causar a absorção de características não importantes para o estudo.

- Quantidade de épocas: a quantidade de épocas de uma rede neural é a quantidade de ciclos(*feed-forward* e *back-propagation*) que a rede executará ajustando os parâmetros para melhorar a sua acurácia. Em alguns casos é viável não especificar o número de épocas, deixando a rede rodar até que não haja melhoria significativa na acurácia.

- Quantidade de camadas ocultas: este parâmetro representa quantas camadas existirão na rede neural entre a camada de entrada e a camada de saída. O valor ótimo varia muito com o problema. Alguns problemas podem ser bem modelados apenas com uma camada, e problemas mais complexos necessitam de mais camadas para se obter melhores resultados.

- Quantidade de neurônios: indica quantos neurônios existirão em cada camada oculta. Quanto maior este parâmetro, mais complexo e pesado é o processamento. Um valor exagerado pode causar uma não-convergência da rede neural a um valor ótimo.

- Tamanho do lote (*batch size*): em problemas onde a base de dados é muito grande, é necessário que ela seja dividida em partes, também chamadas de lotes. Este parâmetro

indica o tamanho de cada lote. A rede neural será treinada separadamente utilizando cada um dos lotes.

3.4 Dos recursos e métodos desenvolvidos

Toda a metodologia do trabalho foi executada utilizando a linguagem de programação *Python3*, em um ambiente Linux, mais especificamente a distro Ubuntu 18.04. A máquina utilizada é composta de um processador AMD Ryzen 7 1800x, de 16 núcleos a 3.6GHz, e 8gb de memória RAM. O algoritmo DNNClassifier foi executado em uma GPU Geforce GTX 1070, com 8gb de VRAM, e frequência de 1708 MHz.

A aplicação web desenvolvida para anotação da base de dados foi feita na linguagem de programação PHP, utilizando-se um banco de dados MySQL, e hospedada em domínio privado em um servidor Apache.

A base de dados coletada utilizando a API do Twitter foi armazenada em planilhas no formato csv, para facilitar a leitura das mesmas, e foram geradas planilhas no formato csv com os resultados obtidos pelos algoritmos.

4 Resultados

Este capítulo descreve os resultados obtidos na análise de sentimentos de *tweets* relacionados à campanha presidencial do Brasil no ano de 2018. Todos os experimentos foram executados utilizando-se o método de validação cruzada estratificada, com 10 *folds*. A medida de desempenho adotada foi a acurácia média dentre as dez execuções. Para fins de organização, o capítulo é dividido em duas seções: a primeira referente aos resultados obtidos simulando algoritmos tradicionais de aprendizado de máquina; e a segunda referente aos resultados obtidos usando redes neurais profundas.

4.1 Algoritmos tradicionais

Os algoritmos tradicionais de aprendizado de máquina utilizados no trabalho foram o Naive Bayes, Árvore de decisão, Regressão Logística e o SVM, todos da biblioteca *Scikit Learn* (Pedregosa et al., 2011). A Tabela 7 mostra os resultados de cada algoritmo, em cada parâmetro configurado. A primeira coluna da tabela representa o algoritmo utilizado, a segunda coluna denota o número limite de características na bag-of-words, a terceira coluna indica se a normalização Tf-Idf foi utilizada, e a quarta e quinta coluna apresentam a acurácia média nos conjuntos de dados de validação e teste, respectivamente. As análises apresentadas a seguir são baseadas nos resultados da tabela.

- O algoritmo de Regressão Logística apresentou os melhores resultados. Isso pode ser explicado pela propriedade do algoritmo de tratar as características dos documentos em conjunto, de modo que características que atrapalham o processo de classificação dos dados podem ser ignoradas. O relacionamento sintático das palavras em uma frase é essencial para o entendimento e a anotação de um texto, logo, esse algoritmo é um dos mais utilizados em PLN (Carneiro et al., 2017).

- Por outro lado, o classificador Naive Bayes foi o que mostrou os piores resultados. Este classificador trata as características dos dados de maneira independente, e por esse motivo já se mostrou menos eficaz também em outros trabalhos, quando é utilizado em tarefas onde é necessário que seja tratada a relação sintática entre as palavras (Wang e Manning, 2012). Ademais, outro fator importante é o tamanho da base de dados, a qual afeta diretamente na suposição de indepência dos atributos.

- O algoritmo de árvore de decisão também apresentou baixa acurácia em relação aos outros algoritmos. Isso se deve ao fato da árvore falhar em classificar dados que apresentam padrões nunca vistos no seu treinamento, tendo em vista que um

Tabela 7 – Resultados dos algoritmos tradicionais na base de dados.

Algoritmo	**Max-feat**	**Tf-Idf**	**Validação**	**Teste**
Naive-Bayes	500	não	0.32	0.34
Árv. de Decisão	500	não	0.46	0.46
Reg. Logística	500	não	0.48	0.51
SVM	500	não	0.52	0.48
Naive-Bayes	1000	não	0.36	0.41
Árv. de Decisão	1000	não	0.47	0.46
Reg. Logística	1000	não	0.45	0.51
SVM	1000	não	0.50	0.48
Naive-Bayes	ilimitado	não	0.32	0.35
Árv. de Decisão	ilimitado	não	0.51	0.45
Reg. Logística	ilimitado	não	0.53	0.49
SVM	ilimitado	não	0.48	0.48
Naive-Bayes	500	sim	0.29	0.34
Árv. de Decisão	500	sim	0.44	0.44
Reg. Logística	500	sim	0.53	0.49
SVM	500	sim	0.52	0.48
Naive-Bayes	1000	sim	0.36	0.42
Árv. de Decisão	1000	sim	0.44	0.46
Reg. Logística	1000	sim	0.48	0.51
SVM	1000	sim	0.49	0.48
Naive-Bayes	ilimitado	sim	0.35	0.39
Árv. de Decisão	ilimitado	sim	0.47	0.45
Reg. Logística	**ilimitado**	**sim**	**0.50**	**0.54**
SVM	ilimitado	sim	0.51	0.48

documento que não se encaixa em nenhuma regra pré-definida pela árvore não será classificado de maneira eficiente.

Em síntese, acredita-se que os resultados apresentados pelos algoritmos podem ser melhorados com o aumento da base de dados anotada e uma seleção mais profunda de parâmetros configurados.

4.2 Rede Neural Profunda (DNN)

O algoritmo de aprendizado profundo utilizado neste trabalho foi a implementação de rede neural profunda disponibilizada no TensorFlow, denominada DNNClassifier. O algoritmo foi executado em 500 épocas, a função de ativação utilizada na camada de entrada e nas camadas ocultas da rede neural foi a ReLU, e na camada de saída foi a Sigmóide. O número de camadas, neurônios e o tamanho do lote foram ajustados para diferentes simulações. Os resultados do algoritmo estão representados na Tabela 8, que

mostra que para um número pequeno de camadas, melhores resultados são obtidos com um número maior de neurônios e tamanho do lote, contudo, a medida que o número de camadas aumenta, os melhores resultados são obtidos reduzindo o número de neurônios e o tamanho do lote, o que pode significar que o aumento do número de camadas ou neurônios não é mais significativo para a melhoria do desempenho preditivo.

Tabela 8 – Resultados obtidos pela rede neural profunda.

Camadas	**Neurônios**	**Tam. lote**	**Valid.**	**Teste**
1	10	10	0.4919	0.4876
1	20	20	0.5118	0.5023
1	50	50	0.4919	0.4876
1	50	100	0.5114	0.5204
2	10	10	0.4991	0.4925
2	20	20	0.5205	0.5193
2	**50**	**50**	**0.5631**	**0.5312**
2	50	100	0.5012	0.4912
3	10	10	0.5114	0.5018
3	20	20	0.5356	0.5295
3	50	50	0.5101	0.5156
3	50	100	0.4910	0.4919

Outro experimento conduzido com redes neurais profundas visou analisar a influência dos passos de pré-processamento (remoçáo de stopwords, acentuação, stemming, transformação em minúsculas) no desempenho preditivo da DNN. Nesse sentido, foram realizadas duas simulações: uma com pré-processamento e outra sem pré-processamento dos *tweets*. Ambas as simulações foram realizadas transformando os dados em uma matriz *bag of words*, e variando o número de épocas. Também foram considerados, neste experimento, os parâmetros que geraram melhores resultados no experimento anterior. A Tabela 9 mostra um comparativo dos resultados obtidos.

É interessante ressaltar o fato de a rede neural profunda ter conseguido uma acurácia maior sem o pré-processamento dos dados. Isso mostra que esse algoritmo tem uma grande capacidade de representação e modelagem de dados, sendo mais independente nesse sentido que a maioria dos outros algoritmos. Aliás, a acurácia média de 54% na Tabela 9 é compatível com o melhor resultado obtido na Tabela 7, usando o classificador de regressão logística. Tal resultado é promissor e abre espaço para mais investigações sobre o assunto, especialmente utilizando redes neurais profundas mais sofisticadas, capazes de considerar a sequência dos dados.

Tabela 9 – Resultados da rede neural profunda com e sem o pré-processamento.

Pré-proc?	**Camadas**	**Neurônios**	**Épocas**	**Validação**	**Teste**
Não	2	50	500	0.5314	0.5397
Sim	2	50	500	0.5123	0.5278
Não	1	10	200	0.5312	0.5211
Sim	1	10	200	0.4549	0.4709
Não	1	20	200	0.5060	0.5182
Sim	1	20	200	0.4479	0.4768
Não	2	20	500	0.5125	0.5312
Sim	2	20	500	0.4417	0.4729
Não	3	10	1000	0.5378	0.5415
Sim	3	10	1000	0.4925	0.5076

5 Conclusão

Durante a coleta de dados deste trabalho, conseguimos uma base muito extensa e diversificada, com tweets que expressam vários sentimentos e inseridos em vários contextos diferentes. Essa variedade é recorrente quando se trabalha com tarefas de PLN. As tarefas que envolvem esta área são complexas, tanto em sua representação computacional quanto para os seres humanos, uma vez que as frases podem estar inseridas em diferentes contextos, e uma má interpretação pode levar a uma anotação errada. Esse fato pôde ser observado durante a fase de anotação dos tweets, onde 25% dos rótulos foram discrepantes entre os colaboradores.

A base de dados vinda do Twitter é muito diversa e complicada, muitas das vezes sendo apresentada por sentenças difíceis de serem rotuladas. Boa parte dos *tweets* coletados não tinham valor semântico suficiente para serem classificados como positivos ou negativos, e acabaram sendo classificados pelos colaboradores como neutros. Isso pode ter gerado uma má classificação de certos *tweets*, o que pode atrapalhar no aprendizado. No entanto, existem diversas técnicas de processamento de línguas naturais, e apenas algumas foram aplicadas neste trabalho. Tais técnicas podem impactar muito nos resultados de algoritmos de aprendizado de máquina (Wang e Manning, 2012).

As redes neurais profundas entregaram bons resultados tendo em vista o baixo número de dados anotados e parâmetros configurados, e se mostraram poderosas ferramentas para a solução do problema proposto, no entanto ainda precisam ser melhoradas. Esses algoritmos conseguem abstrair informações e características de dados desconhecidos com maestria, e vem sendo utilizados amplamente na literatura em várias tarefas na área de análise de sentimentos (Severyn e Moschitti, 2015), portanto, novos testes devem ser conduzidos através delas.

Em suma, os resultados são promissores e sugerem que é possível melhorar a acurácia dos algoritmos, executando-os de diferentes maneiras e tratando os dados com outras perspectivas. Outras abordagens podem ser estudadas em trabalhos futuros, como uma melhor separação das bases de treino, teste e validação, a avaliação de *underfitting* (quando o modelo não consegue abstrair informações suficientes da base de treino, de modo que seja eficaz em dados desconhecidos) e novos testes considerando uma *overfitting* da rede, maior variedade de parâmetros e otimizadores.

A utilização de *n-grams* (termos da bag of words representados como conjuntos de mais de uma palavra) pode ser um bom caminho para melhorar a acurácia das redes neurais (Pak e Paroubek, 2010). Além disso, vários pesquisadores da área (Ng, Andrew Yan-Tak, 2017) mostram que redes neurais profundas são beneficiadas com o aumento

da base de dados anotada. Logo, também é nosso objetivo aumentar a base por meio da aplicação *web* desenvolvida.

Referências

Amazon. *Amazon Mechanical Turk*. 2005. Acessado em : 10 jun. 2018. Disponível em: <https://www.mturk.com>. Citado na página 29.

ARAÚJO, M.; GONÇALVES, P.; BENEVENUTO, F. Measuring sentiments in online social networks. In: ACM. *Proceedings of the 19th Brazilian symposium on Multimedia and the web*. [S.l.], 2013. p. 97–104. Citado na página 12.

ARAÚJO, M. et al. ifeel: a system that compares and combines sentiment analysis methods. In: ACM. *Proceedings of the 23rd International Conference on World Wide Web*. [S.l.], 2014. p. 75–78. Citado na página 29.

BENEVENUTO, F.; RIBEIRO, F.; ARAÚJO, M. *Métodos para Análise de Sentimentos em mídias sociais*. 2015. Citado na página 22.

BRUM, H. B.; NUNES, M. d. G. V. Building a sentiment corpus of tweets in brazilian portuguese. *arXiv preprint arXiv:1712.08917*, 2017. Citado 2 vezes nas páginas 31 e 33.

CARNEIRO, M. G. et al. Semi-supervised semantic role labeling for brazilian portuguese. *JIDM*, v. 8, n. 2, p. 117–130, 2017. Citado 2 vezes nas páginas 17 e 39.

FRAKES, W. B. *Stemming Algorithms*. 1992. Citado na página 35.

GARDNER, M. W.; DORLING, S. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, Elsevier, v. 32, n. 14-15, p. 2627–2636, 1998. Citado na página 24.

GIACHANOU, A.; CRESTANI, F. Like it or not: A survey of twitter sentiment analysis methods. *ACM Computing Surveys (CSUR)*, ACM, v. 49, n. 2, p. 28, 2016. Citado na página 21.

HAN, S. et al. Using the tensorflow deep neural network to classify mainland china visitor behaviours in hong kong from check-in data. *ISPRS International Journal of Geo-Information*, Multidisciplinary Digital Publishing Institute, v. 7, n. 4, p. 158, 2018. Citado na página 28.

HINTON, G. et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, IEEE, v. 29, n. 6, p. 82–97, 2012. Citado 2 vezes nas páginas 25 e 26.

HOFFMAN, D. L.; NOVAK, T. P.; SCHLOSSER, A. E. The evolution of the digital divide: Examining the relationship of race to internet access and usage over time. *The digital divide: Facing a crisis or creating a myth*, Cambridge, MA: MIT Press, p. 47–97, 2001. Citado na página 12.

JIANG, C. et al. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, IEEE, v. 24, n. 2, p. 98–105, 2016. Citado 2 vezes nas páginas 14 e 15.

KIM, Y. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014. Citado na página 30.

KLEINBAUM, D. G. et al. *Logistic regression*. [S.l.]: Springer, 2002. Citado na página 18.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105. Citado 3 vezes nas páginas 7, 26 e 27.

LI, Y.; YUAN, Y. Convergence analysis of two-layer neural networks with relu activation. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2017. p. 597–607. Citado na página 27.

LIU, B. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, Morgan & Claypool Publishers, v. 5, n. 1, p. 1–167, 2012. Citado na página 12.

LORENA, A. C.; CARVALHO, A. C. de. Uma introdução às support vector machines. *Revista de Informática Teórica e Aplicada*, v. 14, n. 2, p. 43–67, 2007. Citado 2 vezes nas páginas 7 e 18.

MALHEIROS, Y. Emotte: Uma ferramenta de análise de sentimentos para o twitter. In: *XX Brazilian Symposium on Multimedia and the Web-Webmedia*. [S.l.: s.n.], 2014. v. 2014. Citado 3 vezes nas páginas 12, 21 e 29.

MARREIROS, A. C. et al. Population dynamics: variance and the sigmoid activation function. *Neuroimage*, Elsevier, v. 42, n. 1, p. 147–157, 2008. Citado na página 28.

MARTINS, D.; KATAOKA, K.; TRINDADE, L. Processamento de linguagem natural. p. 11, 2010. Citado na página 20.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. [S.l.]: MIT press, 2018. Citado na página 14.

MOSTAFA, M. M. More than words: Social networks' text mining for consumer brand sentiments. *Expert Systems with Applications*, Elsevier, v. 40, n. 10, p. 4241–4251, 2013. Citado na página 12.

MYLES, A. J. et al. An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, Wiley Online Library, v. 18, n. 6, p. 275–285, 2004. Citado na página 16.

Ng, Andrew Yan-Tak. 2017. Citado 2 vezes nas páginas 26 e 43.

PAK, A.; PAROUBEK, P. Twitter as a corpus for sentiment analysis and opinion mining. In: *LREc*. [S.l.: s.n.], 2010. v. 10, n. 2010. Citado 5 vezes nas páginas 7, 12, 30, 31 e 43.

PARIKH, D. *Learning Paradigms in Machine Learning*. 2018. Acessado em: 02 fev. 2019. Disponível em: <https://medium.com/datadriveninvestor/learning-paradigms-in-machine-learning-146ebf8b5943>. Citado 3 vezes nas páginas 7, 15 e 16.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, v. 12, n. Oct, p. 2825–2830, 2011. Citado na página 39.

PENNEBAKER, J. W.; FRANCIS, M. E.; BOOTH, R. J. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, v. 71, n. 2001, p. 2001, 2001. Citado na página 30.

PEREIRA, S. do L. Processamento de linguagem natural. 2012. Citado na página 20.

POTTS, C. C. *Potts basic Twitter Tokenizer*. 2011. Acessado em : 11 jun. 2018. Disponível em: <http://sentiment.christopherpotts.net/code-data/happyfuntokenizing.py>. Citado na página 29.

RAMOS, J. et al. Using tf-idf to determine word relevance in document queries. In: PISCATAWAY, NJ. *Proceedings of the first instructional conference on machine learning*. [S.l.], 2003. v. 242, p. 133–142. Citado na página 36.

RASCHKA, S. Naive bayes and text classification i-introduction and theory. *arXiv preprint arXiv:1410.5329*, 2014. Citado na página 16.

RIEDMILLER, M.; LERNEN, A. M. Multi layer perceptron. *Machine Learning Lab Special Lecture, University of Freiburg*, 2014. Citado 3 vezes nas páginas 7, 24 e 25.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, 1958. Citado 2 vezes nas páginas 7 e 23.

SANTOS, C. dos; GATTI, M. Deep convolutional neural networks for sentiment analysis of short texts. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. [S.l.: s.n.], 2014. p. 69–78. Citado na página 25.

Scikit Learn Library. *K-fold - Validação Cruzada*. 2019. Citado 3 vezes nas páginas 7, 18 e 19.

SEVERYN, A.; MOSCHITTI, A. Twitter sentiment analysis with deep convolutional neural networks. In: ACM. *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. [S.l.], 2015. p. 959–962. Citado 2 vezes nas páginas 25 e 43.

SHIBA, M. H. et al. Classificação de imagens de sensoriamento remoto pela aprendizagem por árvore de decisão: uma avaliação de desempenho. *Anais do XII Simpósio Brasileiro de Sensoriamento Remoto, Goiânia-GO*, p. 4319–4326, 2005. Citado na página 16.

Statista. *Twitter - Number of active users*. 2019. Citado na página 22.

SUN, Y. et al. Deep learning face representation by joint identification-verification. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 1988–1996. Citado na página 25.

TRSTENJAK, B.; MIKAC, S.; DONKO, D. Knn with tf-idf based framework for text categorization. *Procedia Engineering*, Elsevier, v. 69, p. 1356–1364, 2014. Citado na página 37.

TUMASJAN, A. et al. Predicting elections with twitter: What 140 characters reveal about political sentiment. *Icwsm*, v. 10, n. 1, p. 178–185, 2010. Citado 2 vezes nas páginas 12 e 30.

TWITTER. *Exemplo de tweet*. 2018. Citado 2 vezes nas páginas 7 e 23.

VOGL, T. P. et al. Accelerating the convergence of the back-propagation method. *Biological cybernetics*, Springer, v. 59, n. 4-5, p. 257–263, 1988. Citado na página 25.

WANG, H. et al. A system for real-time twitter sentiment analysis of 2012 us presidential election cycle. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the ACL 2012 System Demonstrations*. [S.l.], 2012. p. 115–120. Citado 3 vezes nas páginas 7, 29 e 30.

WANG, S.; MANNING, C. D. Baselines and bigrams: Simple, good sentiment and topic classification. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2*. [S.l.], 2012. p. 90–94. Citado 2 vezes nas páginas 39 e 43.

ZHANG, Y.; JIN, R.; ZHOU, Z.-H. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, Springer, v. 1, n. 1-4, p. 43–52, 2010. Citado na página 35.