UNIVERSIDADE FEDERAL DA FRONTEIRA SUL

CAMPUS DE CHAPECÓ

CURSO DE CIÊNCIA DA COMPUTAÇÃO

ARTUR CONSTANZI SPONCHIADO

APLICAÇÃO DE META-ATRIBUTOS PARA DETECÇÃO DE DISCURSO DE ÓDIO

CHAPECÓ

2019

ARTUR CONSTANZI SPONCHIADO

APLICAÇÃO DE META-ATRIBUTOS PARA DETECÇÃO DE DISCURSO DE ÓDIO

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.
Orientador: Dr. Guilherme Dal Bianco

CHAPECÓ

2019

SPONCHIADO, ARTUR CONSTANZI

Aplicação de meta-atributos para detecção de discurso de ódio / ARTUR CONSTANZI SPONCHIADO. – 2019.

48 f.: il.

Orientador: Dr. Guilherme Dal Bianco.

Trabalho de conclusão de curso (graduação) – Universidade Federal da Fronteira Sul, curso de Ciência da Computação, Chapecó, SC, 2019.

1. Discurso de ódio. 2. Aprendizado de máquina. 3. Processamento de linguaguem natural. 4. Classificação de textos. I. Bianco, Dr. Guilherme Dal, orientador. II. Universidade Federal da Fronteira Sul. III. Título.

E-mail: artursponchi@gmail.com

ARTUR CONSTANZI SPONCHIADO

APLICAÇÃO DE META-ATRIBUTOS PARA DETECÇÃO DE DISCURSO DE ÓDIO

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.

Orientador: Dr. Guilherme Dal Bianco

Este trabalho de conclusão de curso foi defendido e aprovado pela banca avaliadora em: 11/12/2019.

BANCA AVALIADORA

Dr. Guilherme Dal Bianco – UFFS

Dr. Denio Duarte – UFFS

Dr. Fernando Bevilacqua – UFFS

RESUMO

Com o advento das redes sociais online, espaços de interação entre usuários estão surgindo. Tais espaços podem ser desde comentários em um site de vídeos, até a seção de discussão em um site de notícias, por exemplo. Nesses ambientes, pessoas podem conversar, expor suas opiniões, debater ideias, etc. Todavia, existem grupos se formando que, aproveitando-se do anonimato, propagam discursos de ódio, disseminando ideias xenofóbicas, racistas, entre outras. Indivíduos que compartilham mensagens deste tipo tendem a disfarçar suas palavras, com caracteres especiais ou até mesmo o sarcasmo, tornando assim uma tarefa difícil para algoritmos básicos bloquearem tal conteúdo. Assim, faz-se necessário um meio para detectar estes discursos de forma automática, visto que devido a quantidade de mensagens publicadas diariamente ser muito grande, tal tarefa torna-se inviável. Utilizando abordagens de pré-processamento de dados e extração de atributos e meta-atributos, este trabalho obtém resultados promissores na área de detecção de discursos de ódio em textos. O método proposto consiste na *stemização* e remoção de *stopwords* na parte do pré-processamento dos dados e na extração de meta-atributos utilizando o algoritmo *KNN*. Utilizando validação cruzada de 10 vezes e o *SVM* como classificador, o método proposto obtém melhorias de até 9.67% em relação ao *baseline* apresentado.

Palavras-chave: Discurso de ódio. Aprendizado de máquina. Processamento de linguaguem natural. Classificação de textos.

ABSTRACT

With the advent of online social networking, user interaction spaces are emerging. Such spaces can range from comments on a video site to the discussion section on a news site, for example. In these environments, people can chat, expose their opinions, discuss ideas, etc. However, groups are emerging and taking advantage of anonymity to spread hate speech, disseminating xenophobic and racist views, among others. Individuals who share such messages tend to hide their words with special characters or even sarcasm. It makes a difficult task for basic algorithms to block the content. Thus, it is necessary a way to detect these speeches automatically, since the amount of messages published daily is very large, such task becomes impracticable. Using preprocessing methods for features and meta-features extraction, this work obtain promising results in the area of hate speech detection in texts. The proposed method consists of stemization and removal of stopwords for data pre-processing and meta-features extraction with *KNN* algorithm. Using 10-fold cross validation and the *SVM* classifier, the proposed method achieves improvements of up to 9.67% over the baseline presented.

Keywords: Hate speech. Machine learning. Natural language processing. Text classification.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

KNN	*K-Nearest-Neighbors*
PLN	Processamento de Linguagem Natural
TF	*Term Frequency*
IDF	*Inverse Document Frequency*
TF-IDF	*Term Frequency-Inverse Document Frequency*
BOW	*Bag-of-words*
PREP	*Pré-processamento*

SUMÁRIO

1 INTRODUÇÃO

1.1 TEMA

Este trabalho visa a extração de meta-atributos para a identificação de discursos de ódio em textos.

1.2 PROBLEMATIZAÇÃO

Com o advento da internet e a popularização das mídias sociais, novos espaços de interação foram surgindo, onde usuários podem trocar ideias, conversar, expor suas opiniões, etc. Sites como *Facebook*, *Twitter* e até mesmo espaços destinados à notícias, como o *G1*, acabaram servindo como meios de disseminação para uma onda de intolerância por parte de seus usuários. Escondidos atrás de seus celulares e computadores, espalham discursos de ódio, disseminando ideias xenofóbicas, racistas, entre outras. O discurso de ódio é definido por "[...] palavras que tendem a insultar, intimidar ou assediar pessoas em virtude de sua raça, cor, etnicidade, nacionalidade, sexo ou religião, ou que têm a capacidade de instigar a violência, ódio ou discriminação contra tais pessoas [...]" (BRUGGER, 2007, p. 118).

Dado o exposto, fica evidente que a classificação manual (realizada por pessoas) de textos na *web* (entre discurso de ódio ou não) é inviável, visto que em uma plataforma como o *Facebook* por exemplo, o número ativo de usuários diariamente é de aproximadamente 1,56 bilhões de pessoas (NOYES, 2019). Portanto, faz-se necessário encontrar uma maneira para classificar tais textos de forma automática.

Como explica Schmidt e Wiegand (2017), o discurso de ódio cresce diariamente junto com as redes sociais online, assim, a complexidade linguística destes discursos também aumenta. O que define se tais sentenças são consideradas ou não discurso de ódio, pode variar de acordo com o contexto, data da postagem, eventos ocorrendo no mundo em tal período, mídias anexas no texto (fotos, vídeos, etc), entre outros elementos. Por consequência, apenas simples filtros de textos, como por exemplo, uma lista com palavras proibidas, não são suficientes para abranger todo este emaranhado de discursos diversificados (SCHMIDT; WIEGAND, 2017).

É necessário uma maneira de analisar e classificar textos com discurso de ódio de maneira automática. Assim, utilizando métodos supervisionados em aprendizado de máquina, é possível automatizar tal tarefa. Para isso, treina-se um algoritmo com uma ou mais bases de dados

contendo exemplos já rotulados, ou seja, textos rotulados manualmente, em discurso de ódio ou não. Depois disso, é possível introduzir exemplos não rotulados no modelo de predição gerado, para assim verificar a eficiência do algoritmo. A classificação dos textos ocorre baseado nas características do documento. Estas características são atributos que descrevem tal objeto. Em textos, alguns exemplos simples que podem ser observados como características são: número de palavras na sentença, número de palavras em caixa alta, pontuações como pontos de exclamação, entre outros.

No trabalho de Lima e Bianco (2019), é proposto a criação de uma abordagem, a qual procura melhorar a eficiência na identificação de discurso de ódio em textos. São propostos a adição de meta-atributos extraídos a partir do texto, tais meta-atributos propostos apresentaram uma melhora nos resultados quando comparado a abordagem sem a utilização destes. Todavia, mais meta-atributos podem ser adicionados para aprimorar ainda mais os resultados.

Este projeto de conclusão de curso possui a finalidade de melhorar a detecção automática de discursos de ódio em textos. Utilizando o algoritmo *KNN*, a partir de um texto de entrada, meta-atributos são extraídos. Assim, juntamente com aprendizado de máquina, busca-se melhorar a capacidade de identificar as mensagens em discurso de ódio ou não. Ademais, como apresentado no trabalho de Lima e Bianco (2019), a utilização de meta-atributos na detecção de discursos de ódio, podem melhorar significativamente a eficiência de classificação em algoritmos de aprendizado de máquina criados com esta finalidade.

1.3 OBJETIVOS

A seguir são listados os objetivos gerais, bem como os objetivos específicos deste trabalho.

1.3.1 Objetivo geral

Aplicação de meta-atributos com o intuito de complementar os já existentes propostos em Lima e Bianco (2019), buscando uma melhoria na detecção de discurso de ódio em textos.

1.3.2 Objetivos específicos

- Definir novos meta-atributos a serem utilizados;

- Realização de testes em bases de dados rotuladas;
- Avaliar os resultados e comparar meta-atributos existentes com outros meta-atributos extraídos.

1.4 JUSTIFICATIVA

As múltiplas redes sociais online que vieram surgindo com a popularização da internet, oferecem diferentes maneiras para seus usuários interagirem, seja em um bate-papo em tempo real ou até mesmo realizando *check-in* em diferentes lugares. Consequentemente, isto acabou gerando um fenômeno que é comumente visto no ambiente online: a formação de comunidades de usuários que compartilham opiniões similares (ALMEIDA; F. G. NAKAMURA; E. F. NAKAMURA, 2017).

Fazer parte de um grupo de pessoas e de alguma forma compartilhar a mesma ideologia com tal, acaba constituindo uma identidade geral para aquela comunidade. Desse modo, pessoas que disseminam discurso de ódio fortificam suas convicções e tendem a se agruparem com outros que compartem deste mesmo conceito imoral, como por exemplo racismo, preconceito geral, violência, etc (ALMEIDA; F. G. NAKAMURA; E. F. NAKAMURA, 2017).

Assim sendo, é necessário que haja um meio com alta eficiência em detectar tais tipos de textos, para assim diminuir o espalhamento de tais manifestações. De acordo com Nobata et al. (2016), identificar tais declarações é uma tarefa complexa por vários motivos. O contexto por exemplo, é um fator que inviabiliza o uso de técnicas como *blacklist* (conjunto de palavras conhecidas por serem de ódio ou utilizadas para insultos), uma vez que o conjunto de palavras está constantemente mudando, além de que uma mesma palavra que insulta um grupo, pode ser totalmente aceitável em outro. Ademais, existem muitas outras variáveis que dificultam o trabalho de identificação, como o sarcasmo ou frases que dependem de sentenças anteriores para fazer sentido.

Além do mais, é importante salientar que, recentemente citado em ELPAÍS (2019), o *Facebook* afirmou estar trabalhando em um algoritmo de inteligência artificial para detecção de violência extremista, discurso do ódio e a desinformação em sua plataforma. Todavia, é dito que para este sistema bloquear de maneira confiável tal conteúdo, ainda deve haver vários anos de desenvolvimento nesta tecnologia, realçando ainda mais o quão importante são estudos sobre este assunto.

2 EMBASAMENTO TEÓRICO

Neste capítulo serão apresentados os princípios essenciais para a boa compreensão deste trabalho. Na Seção 2.1 é definido o que são características (*features*) no contexto de aprendizado de máquina. As Seções 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8 e 2.10 apresentam diferentes métodos e conceitos sobre extração e manipulação de características de documentos. A Seção 2.9 apresenta a função BM25 para ranqueamento de resultados.

2.1 CARACTERÍSTICAS (*FEATURES*)

Conforme Raschka (2015), *features* são atributos/informações que descrevem as características em um dado conjunto de dados. As *features* são utilizadas no aprendizado de máquina como uma forma de o algoritmo reconhecer padrões presentes na base de dados (no formato textual).

Como visto em Sérgio Canuto, L. F. Gonçalves et al. (2013), alguns conjuntos de características podem ser menores e ao mesmo tempo mais informativas. Em contrapartida por exemplo, a extração de características de texto utilizando o método *bag-of-words*, muitas vezes extrai características irrelevantes para a resolução de um determinado problema, visto que essa técnica simplesmente conta o número de ocorrências de uma palavra em um texto. Portanto, pode-se inferir o quão importante é a seleção de características relevantes para o uso em aprendizado de máquina (BLUM; LANGLEY, 1997).

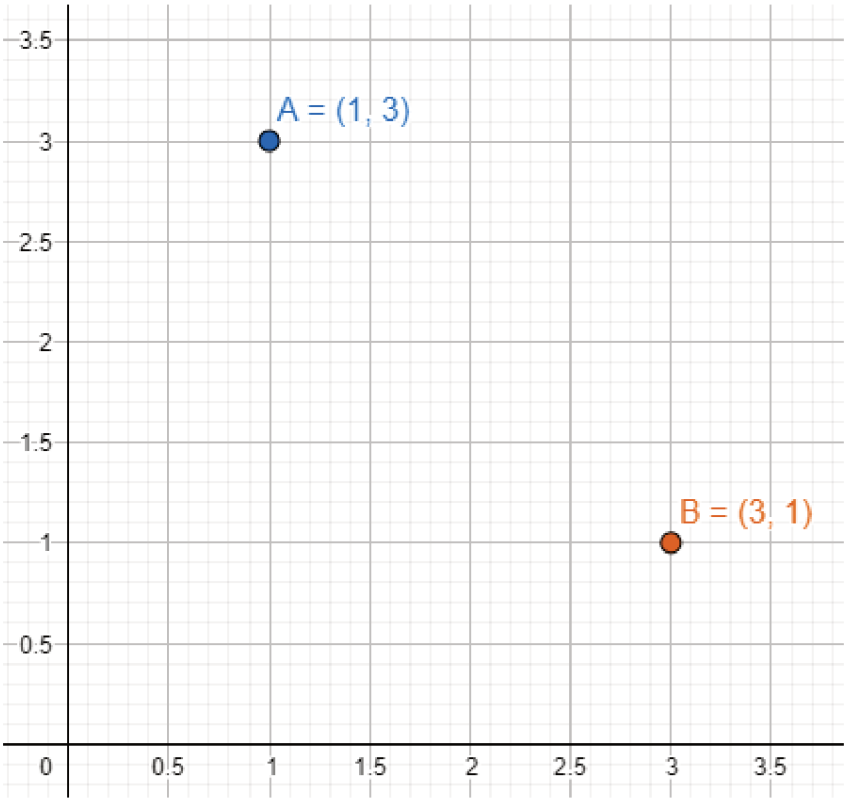
2.2 FUNÇÕES DE SIMILARIDADE

As funções de similaridade são usadas para mensurar o quão similares dois objetos são (HUANG, 2008). Se tratando de dados, essa medida se dá pela distância entre dois objetos, baseado em suas características. Quanto menor a distância, mais caraterísticas os dois objetos possuem em comum.

Uma das funções mais comuns é a da Distância Euclidiana. Conforme Anton (2010), a Distância Euclidiana consiste no comprimento do caminho que liga dois pontos. Para medir a distância entre os pontos A e B da Figura 1, por exemplo, deve-se utilizar a fórmula:

$$\sqrt{(x2 - x1)^2 + (y2 - y1)^2} \tag{2.1}$$

Figura 1 – Exemplo Distância Euclidiana



A partir da Equação 2.1 é possível calcular a distância entre os pontos substituindo as coordenadas X e Y de cada ponto na equação, conforme abaixo:

$$d = \sqrt{(3-1)^2 + (1-3)^2}$$
$$d = \sqrt{(2)^2 + (-2)^2}$$
$$d = \sqrt{4+4}$$
$$d = \sqrt{8}$$
$$d = 2.828427$$

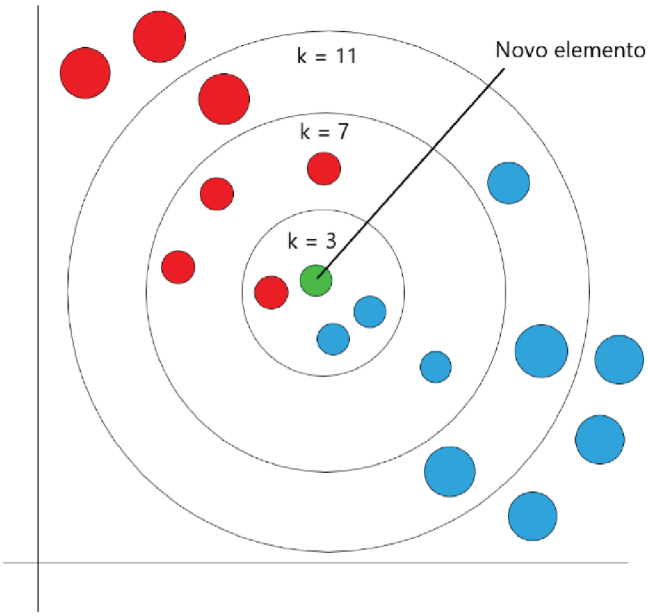
Assim, a distância euclidiana entre os pontos A e B é de 2.828427.

2.3 *KNN*

Segundo Maglogiannis (2007), *KNN* (*K-Nearest Neighbors* ou *K*-vizinhos mais próximos) é um algoritmo de classificação utilizado para identificar o quão similar é um determinado objeto (dado) de outro. O *KNN* é não-paramétrico, ou seja, a estrutura do modelo é determinado a partir dos dados. Além disso, este algoritmo não usa os dados de treinamento para fazer

generalização, ou seja, todos ou a maioria dos dados de treinamento são mantidos e usados durante os testes.

Figura 2 – Exemplo do KNN



Pode-se observar na Figura 2 que dado um novo elemento, busca-se K documentos mais próximos deste novo documento . Assim, utilizando funções de similaridade, como por exemplo a Distância Euclidiana, analisa-se as classes destes K vizinhos mais próximos, para agrupar os dados ou definir a classe mais frequente ao novo elemento. É importante ressaltar que o K é um fator importante no *KNN*, como pode ser visto na Figura 2, para $K = 3$, o novo elemento (círculo verde) seria classificado como azul, todavia quando $K = 7$, seria classificado como vermelho.

2.4 *N-GRAMS*

N-gram é um conceito usado no processamento de linguagem natural (PLN), que consiste basicamente em uma sequência consecutiva de N itens dada uma entrada de texto (DAMASHEK, 1995). Dependendo do tamanho, *n-grams* também são conhecidos por outras definições, como por exemplo: *bigram* ($n = 2$) ou *trigram* ($n = 3$). Veja alguns exemplos a seguir:

- São Paulo (*2-gram* ou *bigram*)
- Você está em São (*4-gram*)
- Você está em São Paulo (*5-gram*)

Figura 3 – Exemplo N-GRAMS

UNIGRAMS

Texto	*Token*	Valor do token
Um dois três quatro	1	"Um"
Um dois três quatro	2	"dois"
Um dois três quatro	3	"três"
Um dois três quatro	4	"quatro"

BIGRAMS

Texto	*Token*	Valor do token
Um dois três quatro	1	"Um dois"
Um dois três quatro	2	"dois três"
Um dois três quatro	3	"três quatro"

TRIGRAMS

Texto	*Token*	Valor do token
Um dois três quatro	1	"Um dois três"
Um dois três quatro	2	"dois três quatro"

Na Figura 3 é possível observar exemplos mais detalhados de *n-grams*, nos quais pode-se notar que se tratando de *unigrams*, cada *token* recebe o valor de somente uma palavra. O mesmo vale para *bigrams* e *trigrams*, que possuem como valores duas e três palavras, respectivamente.

2.5 *TOKENIZAÇÃO*

É o processo de "quebrar" textos em palavras, sendo que cada palavra é um *token*. Esses *tokens* podem ser usados como entrada em um algoritmo. Além disso, é possível usar a *tokenização* em conjunto com *unigram*, *bigrams*, etc. Por exemplo, caso o *bigram* seja utilizado, cada duas palavras irá representar uma característica no conjunto a ser classificado.

Tabela 1 – Exemplo de frequência de *bigrams*.

	Palavra	Palavra	Contagem
1	tudo	bem	32
2	bom	dia	148
3	por	que	64
4	eu	posso	55

Na Tabela 1, é possível observar um exemplo de *tokenização* utilizando *bigrams*. Pode-se notar que o número de ocorrências do *token* 2 ("bom dia") ocorre 148 vezes na coleção de

documentos hipotética.

2.6 *BAG-OF-WORDS*

Para a entrada de um algoritmo de aprendizado de máquina, é necessário uma maneira de "traduzir" as informações para que seja possível que um algoritmo entenda estes dados. Em outras palavras, é necessário converter tais informações para um formato numérico. O método *bag-of-words*, ou em português, bolsa-de-palavras, é um dos métodos mais simples para se extrair características de palavras. Para isso, é feita a *tokenização* de uma sentença, analisando e guardando a informação de quantas vezes um *token* aparece em um determinado texto ou sentença, ou seja, sua frequência (RASCHKA, 2015). Por exemplo:

"Com grandes poderes vêm grandes responsabilidades."

A partir da *tokenização* da frase acima, tem-se o seguinte resultado: "Com", "grandes", "poderes", "vêm", "responsabilidades". É importante ressaltar que neste processo a pontuação não é contabilizada. Cada *token* (palavra) é tratada como um documento distinto. O método permite criar vetores para guardar a informação sobre a frequência de cada *token* no texto.

A Tabela 2 apresenta um exemplo da utilização do método *bag-of-words* a partir da frase de exemplo exposta anteriormente.

Tabela 2 – Exemplo *Bag-of-words*

"Com"	"grandes"	"poderes"	"vêm"	"responsabilidades"
1	2	1	1	1

2.7 *STEMMING*

Segundo Alvares (2014), o *stemming* tem sido muito usado para corrigir (ou diminuir) o problema da variações das palavras, transformando formas variantes de palavras em uma representação mais exata e breve deste conjunto.

Alvares (2014) ainda diz que algoritmos de *stemming* estabelecem uma representação precisa para palavras que apontem para o mesmo significado base, estes podem ser criados fazendo a retirada do sufixo e/ou prefixo de uma palavra. Como pode-se observar na Tabela

Tabela 3 – Exemplo de *stemming*

estudo
estudas
estuda
estudamos
estudais
estudam

3, todas as palavras são diferentes, contudo possuem o mesmo prefixo e remetem a um mesmo significado: estudar.

A utilização dos métodos de *stemming* resultam, se bem aplicados, em uma quantidade de atributos menor e em um melhor desempenho no processo de predição (ALVARES, 2014).

2.8 TF-IDF

O TF-IDF (*frequency–inverse document frequency*, em português, frequência do termo inverso da frequência nos documentos) tem como objetivo atribuir peso para palavras com base na sua importância e frequência. Como explica Ramos et al. (2003), o TF-IDF opera determinando a frequência relativa de palavras em um documento específico e compara com a proporção inversa da palavra analisada em todo o texto do documento. Em textos muito grandes, o *token* "a" por exemplo, pode ocorrer mais frequentemente que outras palavras, porém sem apresentar grande relevância para o contexto.

O TF (*Term frequency*, em português, frequência do termo) conta quantas vezes um *token* aparece em um texto. Além disso, muitas vezes essa contagem é dividida pelo número total de *tokens* do texto, como forma de normalizar os dados, visto que quanto maior for o documento, maior é a probabilidade de certo *token* ocorrer.

$$tf = \frac{tj}{di} \tag{2.2}$$

Onde *tj* significa quantas vezes um *token t* esta presente em um documento, já *di* indica o número total de *tokens* no documento.

$$idf = log(1 + \frac{d}{t}) \tag{2.3}$$

Onde d representa o número total de documentos e t o número de documentos que contém determinado *token*.

Assim, pode-se observar a fórmula completa do TF-IDF na Equação 2.4.

$$tfidf = \frac{tj}{di} * log(1 + \frac{d}{t}) \tag{2.4}$$

Pode-se observar a seguir um exemplo[1] do uso do *TF-IDF*:

- Sentença 1: "*The car is driven on the road.*";
- Sentença 2: "*The truck is driven on the highway.*".

As duas sentenças acima estão em documentos distintos, assim, calcula-se o *TF-IDF* para cada um dos documentos. Os resultados podem ser observados na Tabela 4.

Tabela 4 – Resultado *TF-IDF* (TRIPATHI, 2018)

Word	TF (A)	TF (B)	IDF	TF*IDF (A)	TF*IDF (B)
The	1/7	1/7	log(2/2) = 0	0	0
Car	1/7	0	log(2/1) = 0.3	0.043	0
Truck	0	1/7	log(2/1) = 0.3	0	0.043
Is	1/7	1/7	log(2/2) = 0	0	0
Driven	1/7	1/7	log(2/2) = 0	0	0
On	1/7	1/7	log(2/2) = 0	0	0
The	1/7	1/7	log(2/2) = 0	0	0
Road	1/7	0	log(2/1) = 0.3	0.043	0
Highway	0	1/7	log(2/1) = 0.3	0	0.043

Dados os resultados da Tabela 4, pode-se inferir que as palavras "*car*", "*truck*", "*road*" e "*highway*" são as de maior importância, tendo em vista que possuem um valor superior do *TF-IDF*.

[1] Exemplo disponível em https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/

2.9 BM25

É uma função de ranqueamento que possui como objetivo encontrar resultados relevantes com base em uma entrada de texto.

Conforme Lixin Xu, Guang Chen e Lei Yang (2014), para cada termo de uma sentença, utilizando funções de pontuação, o BM25 irá definir um "peso" para cada documento. Desse modo, baseado no peso total calculado, é possível mensurar a relevância de um texto inicial (pesquisa) com cada documento.

Existem várias funções de pontuação utilizadas no BM25, são elas que possuem o papel de definir o peso para cada documento. Como explica Lixin Xu, Guang Chen e Lei Yang (2014), existem diferentes variantes do BM25, uma das mais difundidas por exemplo, pode ser observada abaixo:

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) * \frac{f(q_i,D) * (k_1+1)}{f(q_i,D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})} \tag{2.5}$$

$$IDF(q_i) = log\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \tag{2.6}$$

A fórmula da Equação 2.5 representa uma pontuação de um dado documento D para uma consulta de texto Q (*query*). O detalhamento dos parâmetros é dado a seguir:

- q_i = Representa o termo i de uma *query* que passou pelo processo de *tokenização*. Por exemplo, supondo que a *query* seja "receita fácil", então tem-se que: $q0$ = *"receita"* e $q1$ = *"fácil"*;
- $f(q_i, D)$ = É a frequência do termo q_i no documento D;
- $\sum_{i=1}^{n} IDF(qi)$ = É a frequência inversa do documento sobre o termo i. Ou seja, quanto mais um termo ocorre em um documento, mais a pontuação deste documento é reduzida. Isto porquê o termo "a" por exemplo, ocorre praticamente em qualquer texto escrito, assim, dada uma *query* "a casa", muito provavelmente o termo *casa* é o mais importante entre os dois. Quanto a fórmula *IDF*, N representa o número total de documentos, enquanto $n(q_i)$ é o número de documentos contendo o termo q_i;
- $\frac{|D|}{avgdl}$ = $|D|$ é o tamanho do documento (número de palavras no documento) e $avgdl$ é a média do tamanho dos documentos de uma coleção. Ou seja, se o tamanho do documento

em questão for maior que a média, consequentemente sua pontuação vai ser menor, caso for menor que a média, então sua pontuação é maior;

- b e k_1 são parâmetros que podem ser ajustados. Por exemplo, quanto maior o valor de b, maior a influência do tamanho do documento em sua pontuação. Já o parâmetro k_1 é utilizado para determinar a saturação da frequência de termos.

2.10 *GANHO DE INFORMAÇÃO*

O Ganho de Informação mede o quanto de informação um atributo/característica fornece a respeito de uma classe (QUINLAN, 1986). Algoritmos de árvores de decisão utilizam o Ganho de Informação para construir tais árvores.

O Ganho de informação utiliza a entropia como medida para a pureza ou impureza de um determinado conjunto. Basicamente controla como uma árvore de decisão decide dividir os dados.

Tabela 5 – Exemplo de divisão.

x1	1	2	3	4	5	6	7	8
y	0	0	0	1	1	1	1	1

Um exemplo pode ser observado na Tabela 5. Supondo que se deseja dividir a variável *x1* baseado em seus valores. Caso for dividido com $x1 < 3.5$, tem-se uma boa divisão, visto que todos os valores onde $x1 = 0$ e $x1 = 1$ estão separados. Todavia, pode ocorrer uma divisão errônea, como por exemplo, se $x1 < 4.5$.

3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados os trabalhos relacionados que são utilizados como base para este projeto. De uma forma geral, serão explicitados diferentes artigos que, utilizando atributos e meta-atributos, tem como objetivo a classificação de textos.

3.1 CLASSIFICAÇÃO AUTOMÁTICA DE TEXTO

Os trabalhos de Sérgio Canuto, L. F. Gonçalves et al. (2013) e Sergio Canuto et al. (2014) têm como objetivo propor meta-atributos para classificação automática de texto. Meta-atributos são atributos derivados de outros atributos originalmente disponíveis (Sérgio CANUTO; L. F. GONÇALVES et al., 2013). Estes podem ser utilizados para substituir ou complementar os atributos originais.

Como os meta-atributos podem capturar relações sobre uma distribuição de dados, é proposto um novo conjunto de meta-atributos baseado no método *KNN*. Assim, além da possibilidade de se observar a distribuição de classes da vizinhança, as distâncias também podem ser mensuradas. Por exemplo, dado um documento x, deve-se analisar a distribuição de classes entre os k documentos mais próximos.

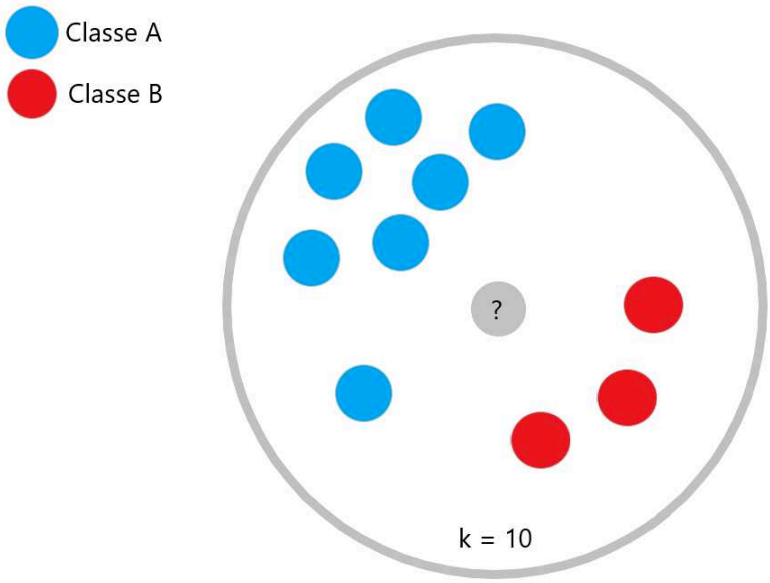
O conjunto de meta-atributos propostos em Sérgio Canuto, L. F. Gonçalves et al. (2013) pode ser observado a seguir:

a) Um vetor unidimensional produzido a partir da contagem dos n vizinhos (dentre os k vizinhos) que contém exemplos de treino positivos em uma categoria c. Como por exemplo na Figura 4, se 7 dos vizinhos estão classificados como uma categoria qualquer A e 3 como uma categoria qualquer B, as duas novas características seriam os valores 7 e 3;

b) Um vetor unidimensional produzido a partir de um documento x não rotulado somando a similaridade do cosseno entre os k vizinhos já rotulados. A normalização é feita a partir da maior soma encontrada. Por exemplo, seguindo o exemplo anterior, as novas características seriam os valores 1.0 ($\frac{7}{7}$) e 0.42 ($\frac{3}{7}$), respectivamente;

c) Um vetor de 5 dimensões produzido a partir de 5 pontos que caracterizam a distribuição das similaridades de um documento x com seus j vizinhos de dada categoria. A distância entre dois documentos é calculada pela similaridade do cosseno. Assim, cinco pontos são

Figura 4 – Exemplo meta-atributo a), dado pela contagem dos vizinhos.

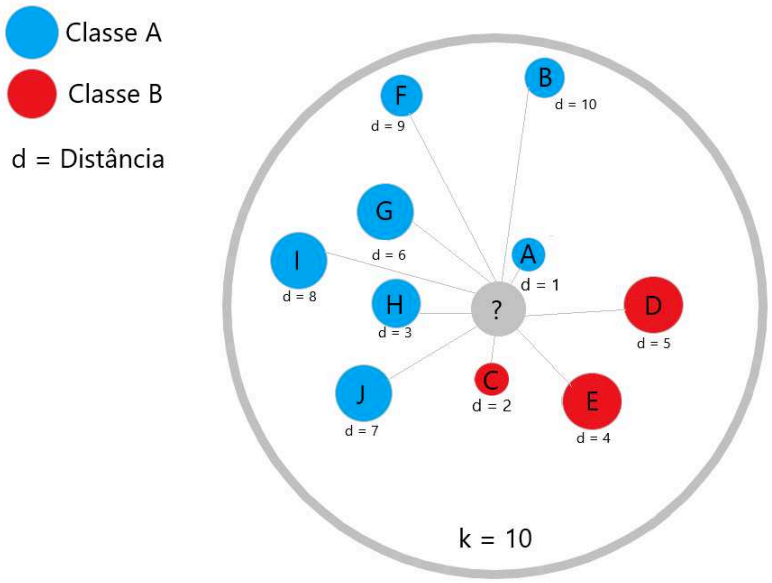


extraídos: a menor distância (dentre todos os vizinhos, é escolhido o vizinho mais próximo ao documento), a maior distância (dentre todos os vizinhos, é escolhido o vizinho mais distante ao documento), a distância média (é calculada a distância média dentre todos os vizinhos), o quartil inferior (valor que delimita os 25% das menores distâncias) e o quartil superior (valor que delimita os 25% das maiores distâncias). Por exemplo, como pode ser visto na Figura 5, a menor distância é o elemento A da classe A, a maior distância é o elemento B também da classe A, a distância média é 5.5 ($\frac{55}{10}$), os elementos pertencentes ao quartil inferior são o A e o C, e finalmente, os elementos pertencentes ao quartil superior são o B e o F;

Em um trabalho posterior, Sergio Canuto et al. (2014) propôs outros meta-atributos:

(i) Um vetor de 5 dimensões relacionado ao item c). Primeiramente, para cada documento de treino, obtêm-se o elemento mais próximo, para assim calcular o centroide deste documento. O mesmo processo repete-se para os vizinhos de maior similaridade, centroide para os vizinhos na mediana, centroides para vizinhos no quartil inferior e quartil superior. Após isso, para um documento de teste, obtêm-se o vizinho com menor similaridade, com maior similaridade, o vizinho na mediana, vizinhos no quartil inferior e superior. Desse modo, calcula-se a similaridade do cosseno entre cada um desses documentos e o seu respectivo centroide;

Figura 5 – Exemplo meta-atributo c), dado pela distribuição das similaridades.



(ii) Um vetor de dimensão C (onde C é o número total de classes) contendo a similaridade média entre o centroide de uma classe x e todo o conjunto dos vizinhos pertencentes a esta classe x;

(iii) De acordo com a medida do Ganho de Informação (BREIMAN et al., 1984), um vetor de três dimensões é produzido, considerando o ponto que melhor divide os vizinhos de x pertencendo a uma categoria c. O ponto escolhido é aquele que melhor divide a classificação em duas partes. Este ponto é o que maximiza o ganho de informação para a classe c.

(iv) Um vetor de uma dimensão que captura a correlação entre os vizinhos de um documento x que pertencem a uma categoria c e os vizinhos restantes que não pertencem a tal categoria (considerando uma vizinhança de tamanho k). Assim, é computada a correlação de Fisher: $\frac{|u_i - u_j|}{\sqrt{V_i + V_j}}$. Onde u_i corresponde a similaridade média da vizinhança positiva (documentos pertencentes a categoria c) e u_j a similaridade média da vizinhança negativa (documentos não pertencentes a categoria c). Ademais, $\sqrt{V_i + V_j}$ corresponde as variâncias (dispersões entre os documentos) calculadas para cada positiva e negativa similaridade, respectivamente.

As características extraídas pelo item a) exploram a distribuição de classes da vizinhança de um documento. Já no item b), os atributos capturam a relação entre a classe majoritária e as

informações de similaridade entre os vizinhos e um documento. A distribuição das similaridades é extraído por c), que considera os três quartis juntamente com o mínimo e o máximo valor de similaridade. Em (i), os quartis são utilizados avaliando seu desvio para valores esperados, o que fornece informações relacionadas a cada classe. Já no item (ii) a coesão das classes na vizinhança de um documento é avaliada. O item (iii) analisa a entropia da vizinhança, se o ponto de divisão estiver baixo entre os vizinhos mais próximos, destaca-se então um cenário de baixa entropia, visto que a maioria dos exemplos mais semelhantes pertencem à mesma classe. Finalmente, o item (iv) utiliza a correlação entre as similaridades dentre os exemplos positivos e negativos dentro da vizinhança de um documento. Valores baixos indicam regiões de baixa entropia na vizinhança.

Os trabalhos de Sérgio Canuto, L. F. Gonçalves et al. (2013) e Sergio Canuto et al. (2014) compartilham algumas semelhanças, ambos possuem os meta-atributos descritos nos itens a), b) e c).

Para a realização dos testes do trabalho de Sérgio Canuto, L. F. Gonçalves et al. (2013), os atributos originais, os meta-atributos do estado-da-arte e os meta-atributos propostos, são utilizados em 5 coleções textuais: *4 Universities (4UNI)*, *Reuters (REUT)*, *ACM-DL (ACM)*, *20 Newsgroups (20NG)* e *Spambase (SPAM)*. Os novos meta-atributos propostos foram analisados isoladamente, bem como, em conjunto com outros meta-atributos e atributos originais. Assim, pode-se observar ganhos de até 19% em algumas coleções, em comparação aos atributos estado-da-arte. Além disso, pode-se notar que quando utilizados em conjunto com outros atributos e meta-atributos, os resultados são ainda melhores, chegando em ganhos de até 20%.

Já no trabalho Sergio Canuto et al. (2014), a fim de utilizar os meta-atributos apresentadas acima, foram adotadas duas estratégias. A primeira tenta aprender o peso "ideal" de cada grupo de *features* e a melhor forma de combiná-las. A segunda tenta reduzir o número de *features* utilizadas, enquanto ainda maximiza sua eficiência.

Para os testes foram utilizas 6 coleções textuais: *4 Universities (4UNI)*, *Reuters (REUT90)*, *ACM-DL (ACM)*, *20 Newsgroups (20NG)*, *MEDLINE (MED)* e *RCV1Uni*.

Quantos aos resultados, os meta-atributos propostos mostraram-se superiores enquanto as características originais (oriundos do método *bag-of-words*, com resultados até 45% melhores, também obteve resultados superiores de até 7% quanto aos meta-atributos do estado-da-arte. Além disso, o desempenho para gerar os meta-atributos foi três vezes mais rápido do que os propostos no estado-da-arte.

3.2 *EXPLOITING NEW SENTIMENT-BASED META-LEVEL FEATURES FOR EFFECTIVE SENTIMENT ANALYSIS*

No trabalho de Sérgio Canuto, M. A. Gonçalves e Benevenuto (2016), o objetivo é utilizar o aprendizado de máquina supervisionado para classificar o "sentimento" do texto. Para isso, são propostas características chamadas de *meta-level features*. Estes meta-atributos são gerados com duas funções de similaridade: Cosseno e o BM25.

As *meta-level features* propostas são:

- Um vetor de k dimensões, criado a partir dos k vizinhos mais próximos pertencentes a uma classe. Desse modo, k *meta-features* são geradas considerando a pontuação do *BM25* ou da similaridade do cosseno entre os vizinhos;
- Um vetor de uma dimensão é criado a partir da soma dos pesos de polaridade dos k vizinhos mais próximos pertencentes a uma determinada classe. A similaridade entre um exemplo x com seu(s) vizinho(s) mais próximo(s) é utiliza para avaliar a polaridade do documento. A polaridade é dada por métodos como o *SentiStrength*, *Vader* e *SentiWordNet*. O *SentiStrength* atribui um "peso" para uma mensagem de acordo com a palavra com a pontuação de positividade ou negatividade mais alta em uma sentença (THELWALL, 2017). Já o *Vader* soma as pontuações das palavras em uma sentença (HUTTO; GILBERT, 2014). Finalmente, o *SentiWordNet* explora a relação entre palavras neutras para encontrar a polaridade. Por exemplo, as palavras "cachorro" e "morder" podem evoluir para um termo como "dor" (BACCIANELLA; ESULI; SEBASTIANI, 2010);
- Um vetor de 2 dimensões criado a partir do valor máximo e mínimo do peso de uma polaridade de um documento dos k vizinhos mais próximos pertencentes a uma classe. A similaridade entre eles é usada para dar um "peso" ao documento;
- Um vetor de uma dimensão contendo a soma da pontuação de polaridade dos k vizinhos mais próximos;
- Um vetor de uma dimensão criado a partir de uma ou várias saídas de um método léxico. A saída corresponde a pontuação de polaridade do documento.

Para os testes, foram utilizados um total de 19 conjunto de dados. Um pré-processamento foi aplicado aos dados, e as mensagens são classificadas em dois tipos: positiva ou negativa.

Todos os experimentos foram executados usando o procedimento *5-fold cross-validation*, e para avaliar o desempenho foi utilizado o *SVM*. Mais ao final, pode-se observar que em comparação ao método do cosseno, o BM25 sempre tem melhor desempenho (em pequenos textos), com ganhos de até 2%.

Ao longo dos resultados dos testes, pode-se observar que geralmente os meta-atributos propostos obtiveram melhores resultados em comparação ao método *bag-of-words*. O método proposto foi melhor em 17 dos 19 conjuntos de dados, obtendo melhorias em até 16%. Além disso, em comparação aos meta-atributos estado-da-arte propostos por Gopal e Yang (2010), o método proposto foi melhor em todos os conjuntos.

3.3 EXTRAÇÃO DE CARACTERÍSTICA PARA IDENTIFICAÇÃO DE DISCURSO DE ÓDIO EM DOCUMENTOS

Em Lima e Bianco (2019), é explorado o uso das meta-features (propostas originalmente em Sérgio Canuto, L. F. Gonçalves et al. (2013)) com o intuito de identificar textos contendo discurso de ódio.

Inicialmente o algoritmo *KNN* é utilizado para criar novas características com as informações da vizinhança mais próxima. Desse modo, é possível capturar informação do conjunto previamente rotulado de 3 diferentes formas: contagem de exemplos rotulados, relação da classe com maior número de vizinhos com as outras classes, e a análise das distribuição das distâncias para cada classe. Por exemplo, na primeira forma, se 10 dos vizinhos estão classificados como discurso de ódio e os outros 20 como sendo de outra categoria, as duas novas características são representadas como um vetor com os valores 10 e 20. A segunda forma refere-se a normalização, onde para cada número de vizinhos em cada classe, divide-se este número pelo número de vizinhos da maior classe. Na terceira forma, a distância para cada classe é analisada nas seguintes características: menor distância, maior distância, distância média, quartil inferior e quartil superior.

No total foram 24 experimentos realizados, 12 para cada base de dados. Os meta-atributos propostos no trabalho tiveram melhor desempenho quando somados com as características do *baseline* (atributos originais). Além disso, o *SVM* foi melhor na maioria dos casos. Já o método LIMA (quantidade de características do método proposto para cada experimento) apresentou ganhos de até 3,3%, e para o classificador NB (Naive Bayes) os resultados, em sua maioria, estiveram abaixo do *baseline*.

Resumindo, pode-se entender que os experimentos com melhores resultados foram os que utilizaram os meta-atributos combinados com outras características, ou seja, os que utilizaram redução de atributos, onde as características mais relevantes são selecionadas. Além disso, também pode-se notar que o classificador *SVM* gerou os melhores resultados.

3.4 *OFFENSIVE COMMENTS IN THE BRAZILIAN WEB: A DATASET AND BASELINE RESULTS*

O trabalho de Pelle e Moreira (2017) explica que os brasileiros são uns dos povos mais ativos em redes sociais, o que torna o trabalho de classificar conteúdo mal-intencionado ainda mais árdua. Para se ter um bom modelo preditivo, é necessário uma boa base de dados com mensagens rotuladas. Como tais bases são difíceis de se encontrar no idioma português, este artigo relacionado tem como contribuição a construção de uma base de dados de textos rotulados, além de apresentar resultados atingidos com algoritmos de classificação, com base no acervo criado.

Os comentários foram extraídos do *website* de notícias G1, das seções de notícias e esportes. Cerca de 90% das notícias analisadas possuíam pelo menos um comentário ofensivo.

Na parte de classificação das mensagens extraídas, foram utilizados três usuários, com o objetivo de identificar se o texto era ofensivo ou não, caso a resposta for afirmativo, o usuário também deveria indicar a(s) classe(s) da ofensa, como por exemplo racismo, sexismo, entre outros.

As bases, ambas contendo 1250 comentários, foram separadas em duas:

- *OffComBR-2*: o comentário era classificado como ofensivo ou não, pela resposta da maioria, neste caso, 2 juízes;
- *OffComBR-3*: este conjunto de dados possui apenas comentários onde os três juízes atribuíram a mesma classificação para cada mensagem.

Quanto as estatísticas, o conjunto *OffComBR-2* apresentou 32,5% dos comentários considerados como ofensivo. Já em *OffComBR-3*, este número ficou em 19,5%. Comentários considerados como blasfêmia foram maioria nos dois grupos, bem como somente 1 comentário identificado como intolerância religiosa.

A fim de preparar o processo de classificação automática destes discurso, algumas medidas de pré-processamento foram adotadas:

- *Case folding*: Duas abordagens foram adotadas neste quesito, a de converter todos os comentários para letras de caixa baixa, e a de manter a forma original da mensagem;
- *Tokenização*: Fazendo o uso da abordagem *bag-of-words*, utilizar *unigrams, bigrams e trigrams* como meta-atributos. Explica-se que usando maiores *n-grams*, pode-se capturar a estrutura do discurso em questão.
- Seleção de características (*features*): Foram selecionadas somente características relevantes para o trabalho. Isto é, utilizando o software *Weka*, foi possível comparar o uso de todos os *n-grams* com somente o uso de informações importantes. Estas, foram definidas pelo método *Information Gain*, que mede a correlação de cada *feature* com a classe que se deseja prever.

Após a execução da classificação, foi notado que os melhores resultados foram produzidos utilizando a base *OffComBR-3*. Além disso, o classificador *SVM* mostrou-se como melhor escolha, visto que obteve uma melhor pontuação (0.80) quando comparado com *Naive Bayes* (0.75). Além de que encontrou menos casos falso positivos (15 casos contra 184 encontrados pelo classificador *Naive Bayes*.

Quanto a eficiência sobre os métodos de pré-processamento aplicados, pode-se concluir os seguintes pontos:

- Converter o texto para caixa-baixa proporcionou melhores resultados;
- Não houve resultados significativos entre *unigrams, bigrams e trigrams*. Portanto, é preferível utilizar somente *unigrams*, visto que *bigrams e trigrams* geram mais meta-atributos;
- A seleção de características proporcionou melhores resultados, bem como redução do número de *features* em 4,8%.

3.5 *ABUSIVE LANGUAGE DETECTION IN ONLINE USER CONTENT*

Em Nobata et al. (2016), o objetivo é detectar linguagem abusiva utilizando aprendizado de máquina. “Linguagem abusiva” é a definição utilizada pelo trabalho para englobar qualquer sentença que ridicularize ou assedie um ou mais indivíduos.

Foram usados 3 conjunto de dados, todos extraídos da seção de finanças e notícias do site *Yahoo!*. Afim de rotular cada mensagem, diferentes maneiras foram testadas. A

primeira, utilizando juízes treinados, a taxa de concordância foi de 92,2%. O segundo modo, foi utilizando juízes não treinados, provenientes da plataforma *Amazon Turk*[1]. Neste caso, a taxa de concordância foi de apenas 40,5%.

O método de classificação supervisionada utilizado foi o modelo de regressão *Vowpal Wabbit's* (sistema de aprendizado de máquina desenvolvido originalmente no *Yahoo!* Pesquisa). As *features* utilizadas neste trabalho são dividias em 4 classes:

- *N-grams*: Foram utilizados *n-grams* na abordagem de caracteres (de 3 a 5 caracteres, incluindo espaços) e *n-grams* na abordagem de *tokens* (*unigrams* e *bigrams*);
- Linguística: Para lidar com ruído nas mensagens, várias *features* foram extraídas, como por exemplo: média do tamanho de palavras, número de letras capitalizadas, número de pontuações, etc;
- Sintática: O intuito desta classe de *features* é capturar longas relações entre palavras, que por exemplo, *n-grams* falham em obter. Desse modo, utilizando *tags*, marcações são feitas em palavras que fazem parte de um discurso, estas podem ser verbos, substantivos, etc. Para isso, um dos métodos utilizados neste grupo é o *POS (Part-of-Speech)*;
- Distribuição semântica: Foram utilizadas diferentes técnicas para extrair tais *features*, como por exemplo o *Word2vec*, que tem como papel entender contextos de palavras utilizadas em um texto.

O modelo foi treinado e testado separados em suas duas categorias: finanças e notícias. Foi utilizado 80% do conjunto para treinamento e 20% para testes. Combinando todas as *features* extraídas foram atingidos os melhores resultados, sendo 79,5% para a categorias Finanças e 81,7% para Notícias. A categorias de Notícias tem um vantagem em seu desempenho que pode ter sido influenciado por sua base de testes ser ligeiramente maior, comparado a base de Notícias.

3.6 *DETECTING OFFENSIVE LANGUAGE IN SOCIAL MEDIA TO PROTECT ADOLESCENT ONLINE SAFETY*

Em Chen et al. (2012), apresenta-se o problema da exposição a conteúdos ofensivos em redes sociais, principalmente por parte de adolescentes. Só em 2011, 70% dos adolescentes americanos já usavam as mídias sociais, acessando-as pelo menos 10 vezes por dia.

[1] Serviço de contratação de trabalhadores *online*, que são pagos para realizar determinada tarefa.

A fim de desenvolver algo eficiente para a detecção de ofensas na internet, este artigo apresenta o *Lexical Syntactic Feature-based (LSF) language model*, um modelo de linguagem baseado em características sintáticas lexicais.

O LSF consiste principalmente de dois componentes: a medição da ofensividade de uma sentença e a estimativa da ofensividade do usuário. Durante a etapa de pré-processamento, o histórico de conversas do usuário é separado em sentenças. Cada sentença tem seu grau de ofensividade derivado de duas características: a ofensividade das palavras e do contexto. Características léxicas são utilizadas para representar a ofensividade das palavras em uma sentença, já para o contexto, características sintáticas são empregadas.

Para calcular a ofensividade das palavras, dois dicionários foram construídos, baseado no grau de ofensa representado por uma palavra. Depois, o conceito de "intensificadores" foi introduzido para medir a ofensividade baseado no contexto. Um intensificador é uma palavra que relaciona duas palavras em uma sentença. Estes são úteis para identificar se as palavras ofensivas estão sendo usadas para descrever usuários ou outras palavras ofensivas. Finalmente, para cada sentença, agregando os valores de cada palavra, um valor de ofensividade é gerado para as sentenças.

Os resultados mostraram que, dado os atributos propostos com a adição do LSF, obteve-se um complemento nos métodos tradicionais de aprendizado de máquina. Apesar de conseguir bons resultados, este trabalho não utiliza meta-atributos e se baseia em características a nível de usuário, diferente do método proposto neste trabalho, onde únicos dados em que se tem acesso é o texto escrito pelo usuário.

4 PROPOSTA

Como descrito neste trabalho, vários métodos de processamento de linguagem natural podem ser utilizados para a detecção de discursos de ódio em textos. A utilização de algorítimos como o KNN contribuem para a extração de atributos complementares para a classificação, tornando-a assim, mais eficiente.

Tendo como objetivo a melhoria na taxa de acerto de um classificador, algumas abordagens podem ser utilizadas. Como visto no trabalho de Lima e Bianco (2019), a redução de atributos aliada com a seleção dos mais importantes, tende a melhorar os resultados. Para isso, métodos como o pré-processamento dos dados e a utilização dos próprios atributos existentes para criação de outros, mostrou-se uma abordagem promissora.

Inspirado nos trabalhos de Lima e Bianco (2019), Sérgio Canuto, L. F. Gonçalves et al. (2013) e Sergio Canuto et al. (2014), este trabalho possui como proposta a adição de meta-atributos, o pré-processamento de conjuntos de dados e o teste em novas bases de dados, buscando desse modo, melhorias na identificação de discursos de ódio em textos.

4.1 MÉTODO PROPOSTO

Como já mencionado anteriormente, uma das contribuições deste trabalho é a adição de meta-atributos para complementação dos já existentes. Para isso, utiliza-se do algoritmo de classificação KNN para encontrar a vizinhança dos documentos já rotulados e obter a distância desses, a qual é conseguida pela similaridade do cosseno. Ademais, a base de dados utilizada é a mesma abordada por Pelle e Moreira (2017) e Gibert et al. (2018) e os meta-atributos adicionados foram extraídos do método proposto por Sergio Canuto et al. (2014).

Possuindo a informação da vizinhança de cada documento, é possível realizar a extração de características:

- 1) Vetor de dimensão C (número de classes) que captura a coesão da classe comparando a similaridade da vizinhança de um documento x com o centroide da classe correspondente à x. Quanto maior a similaridade entre eles, maior a chance de pertencerem a mesma classe;
- 2) Vetor unidimensional que captura a semelhança entre os exemplos positivos e negativos dentro da vizinhança, avaliando assim, a entropia desta. Valores baixos indicam uma

vizinhança menos dispersa e mais previsível.

Apesar de serem 2 meta-atributos adicionados, ao final, estes totalizam um total de 4, pois cada meta-atributo é atribuído para sua respectiva classe. O problema analisado nesse trabalho possui cada texto classificado em 2 classes: discurso de ódio e não discurso de ódio, justificando assim o total de 4 meta-atributos resultantes.

O pseudocódigo da extração dos meta-atributos apresentados estão na Listagem 4.1. Nele, é possível observar a criação de um *dataframe* para guardar as informações dos meta-atributos. Após, com os dados e seus respectivos rótulos, treina-se o algoritmo KNN, para a partir desse, percorrer cada texto para fazer a extração dos meta-atributos a partir de sua vizinhança.

```
1  funcao gerarCaracteristicas(textos, rotulos):
2      dataframe = DataFrame()
3      documentos = KNN()
4      documentos.treinar(textos, rotulos)
5      vizinhosIndices, vizinhosDistancias = documentos.vizinhos()
6
7      centroideClasse0 = Centroid(documentos[classe == 0])
8      centroideClasse1 = Centroid(documentos[classe == 1])
9
10     i = 0
11     PARA CADA textos FACA:
12         vizinhosClasse0 = textos[classe == 0]
13         vizinhosClasse1 = textos[classe == 1]
14
15         // Meta-atributo A
16         similaridade = cos(vizinhosClasse0, centroideClasse0)
17         dataframeMetaAtributos.loc[i, 'atributo_A_Classe0'] = similaridade
18
19         similaridade = cos(vizinhosClasse1, centroideClasse1)
20         dataframeMetaAtributos.loc[i, 'atributo_A_Classe1'] = similaridade
21
22         // Meta-atributo B
23         similaridade =
24             (correlacaoVizinhancaNegativa0 - correlacaoVizinhancaPositiva0)
25             / sqrt(varianciaVizinhancaPositiva0 +
   varianciaVizinhancaNegativa0)
26         dataframeMetaAtributos.loc[i, 'atributo_B_Classe0'] = similaridade
```

```
27
28          similaridade =
29              (correlacaoVizinhancaNegativa1 - correlacaoVizinhancaPositiva1)
30              / sqrt(varianciaVizinhancaPositiva1 +
    varianciaVizinhancaNegativa1)
31          dataframeMetaAtributos.loc[i, 'atributo_B_Classe1'] = similaridade
32
33          i = i + 1
```

Listing 4.1 – Pseudocódigo da extração de meta-atributos

5 ANÁLISE EXPERIMENTAL

Neste capítulo são apresentados os experimentos utilizados para avaliar a eficiência dos métodos propostos, além do resultado desses. Em comparação a proposta apresentado por Lima e Bianco (2019), o método proposto nesse trabalho tem o objetivo de melhorar a eficiência na classificação de discursos de ódio em textos por meio do pré-processamento dos dados e a adição de alguns dos meta-atributos apresentados por Sergio Canuto et al. (2014).

A Seção 5.1 apresenta os conjuntos de dados que foram utilizados para os experimentos, bem como suas características e processos aplicados. Já na Seção 5.2, é apresentado como foram feitos os experimentos e explanado suas nomenclaturas. Na Seção 5.3, demonstra-se as configurações utilizadas para avaliar a eficácia do método proposto. Na Seção 5.4 são apresentados os resultados dos experimentos em cada base de dados. Finalmente, na Seção 5.5 esses são debatidos.

5.1 BASE DE DADOS

O primeiro conjunto de dados utilizados neste trabalho é o mesmo apresentado em Pelle e Moreira (2017) e está disponível para *download*[1]. Constituída de comentários retirados da web, esta base de dados foi rotulada por três juízes e dividida em duas partes: *OffComBR-2* e *OffComBR-3*. A primeira parte é composta por 1250 comentários, sendo 419 rotulados como discurso de ódio e 831 como neutros e/ou não ofensivos. O sufixo "2" indica que para este conjunto, cada comentário foi rotulado como discurso de ódio por pelo menos dois juízes. Já a segunda divisão possui 1033 comentários, sendo 202 considerados como discurso de ódio e 831 como neutro e/ou não ofensivos. O sufixo "3" indica que para este conjunto, só foram selecionados comentários onde os 3 juízes concordaram na mesma classificação (entre discurso de ódio ou não).

O segundo conjunto de dados utilizado é o exposto em Gibert et al. (2018) e está disponível para *download*[2]. É constituída de 1914 textos classificados perante discurso de ódio por 3 juízes. Dentre os textos, 957 foram considerados como classificação positiva (que contém discurso de ódio) e 957 como classificação negativa. Os textos são comentários retirados aleatoriamente de publicações de um fórum supremacista branco. Além disso, a sigla usada para esse conjunto de dados nos experimentos será *GIBERT*.

[1] https://github.com/rogersdepelle/OffComBR
[2] https://github.com/aitor-garcia-p/hate-speech-dataset

Base de dados	Comentários	Classificação positiva	Taxa de exemplos positivos
OffComBR-2	1250	419	33.52%
OffComBR-3	1033	202	19.55%
GIBERT	1914	957	50.00%

Tabela 6 – Bases de dados e suas características.

Um resumo das bases de dados e suas características pode ser observado na Tabela 6.

5.1.1 Pré-processamento dos dados

No trabalho de Lima e Bianco (2019), os métodos de pré-processamento de texto utilizados são variações de *n-grams* e a modificação do texto para caixa baixa. Buscando o maior refinamento do conjunto de dados, neste trabalho foram aplicados mais dois métodos de pré-processamento: remoção de *stopwords* e *stemização*. A remoção de *stopwords* consiste na remoção das palavras mais comuns de uma língua, além das palavras funcionais, como "o", "a", "em", etc. Tal pré-processamento reduz o número de características irrelevantes, auxiliando assim, a eficácia em alguns casos. Já a *stemização* reduz palavras que apontem para um mesmo significado para sua base. Elas podem ser criadas removendo o sufixo e/ou prefixo das palavras. Um exemplo de *stemização* pode ser visto na Tabela 3 na Seção 2.8 do Capítulo 2.

Para avaliar a eficácia desta configuração, foi executado um experimento apresentado na Tabela 7 comparando a utilização do modelo *bag-of-words* com e sem o pré-processamento proposto. Os experimentos com o sufixo *PREP* indicam que o conjunto de dados passou pelo pré-processamento antes de serem executados, além disso, as colunas *BR-2*, *BR-3* e *GIBERT*, indicam as bases de dados *OffComBR-2*, *OffComBR-3* e *GIBERT*, respectivamente. Os experimentos foram validados utilizando validação cruzada de 10 vezes e a métrica *f1_weighted*.

Os resultados mostram que foi possível identificar ganhos quando utilizado o *PREP*, além da redução no número de características. Por exemplo, a categoria *lower_1G_2G_3G_FS_PREP* teve ganho de até 9.67% quando comparada com o mesmo experimento sem a aplicação do pré-processamento, desse modo, pode-se concluir uma ascensão significante no benefício do uso desse método.

	BR-2		BR-3		GIBERT	
Configuração	**Atributos**	***F1***	**Atributos**	***F1***	**Atributos**	***F1***
lower_1G	4.122	71.50%	3.646	77.47%	5951	74.25%
lower_1G_FS	253	67.94%	148	80.48%	298	77.43%
lower_1G_2G	15.898	70.49%	13.881	77.62%	29403	72.59%
lower_1G_2G_FS	265	67.97%	149	82.14%	308	74.87%
lower_1G_2G_3G	29.125	69.15%	25.302	77.23%	60198	72.12%
lower_1G_2G_3G_FS	270	68.06%	149	81.21%	314	74.81%
lower_1G_PREP	3.753	69.87%	3.309	76.61%	4645	71.23%
lower_1G_FS_PREP	248	75.13%	146	82.26%	354	78.81%
lower_1G_2G_PREP	12.170	70.19%	10.535	76.78%	20861	72.43%
lower_1G_2G_FS_PREP	250	77.25%	146	82.25%	359	79.22%
lower_1G_2G_3G_PREP	20.044	69.11%	17.299	76.98%	36782	70.64%
lower_1G_2G_3G_FS_PREP	248	77.73%	145	82.99%	361	78.40%

Tabela 7 – *BOW* X *BOW* + pré-processamento

5.2 CONFIGURAÇÃO DOS EXPERIMENTOS

Visando a melhor exploração do método proposto, os experimentos foram divididos em categorias, buscando assim a melhor combinação de características provenientes de alguns métodos de processamento de linguagem natural. Ainda, devido a melhora considerável quando os conjuntos foram submetidos ao pré-processamentos de dados, todos os experimentos aqui expostos foram executados com essa característica.

Na Figura 6 e na Figura 7, é possível observar diagramas contendo os passos para a extração dos meta-atributos extraídos neste trabalho. Dada uma entrada de características pré-processadas (PREP), os vizinhos e as distâncias para cada documentos são obtidos. Assim, extrai-se os meta-atributos do método proposto por Lima e Bianco (2019) (LIMA), além dos adicionados neste trabalho e propostos por Sergio Canuto et al. (2014) (PROP 1 e PROP 2). Além disso, em alguns experimentos foi aplicado a selação de atributos na entrada PREP, reduzindo assim o espaço de atributos para apenas os mais relevantes.

Os experimentos com o prefixo *lower* tiveram seus textos reduzidos para caixa baixa, diminuindo assim a quantidade de características. Já as siglas *1G*, *2G* e *3G*, indicam a aplicação de *unigrams*, *bigrams* e *trigrams* ao texto, respectivamente. Além disso, a sigla *FS* aponta o

emprego da seleção de melhores características utilizando o índice GINI.

Com o objetivo de comparar os resultados deste trabalho com os apresentados em Lima e Bianco (2019), as categorias criadas para os experimentos e os métodos utilizados em tais se assemelham significativamente em ambos, diferenciando-se em 2 meta-atributos e no pré-processamento dos dados. Desse modo, nos experimentos que serão apresentados, a coluna *LIMA* indica o número de características em dada categoria no trabalho citado. Por outro lado, a coluna *PROP* indica a quantidade de características do trabalho de Lima e Bianco (2019) em complemento com as obtidas neste trabalho.

Figura 6 – Diagrama da extração do meta-atributo 1

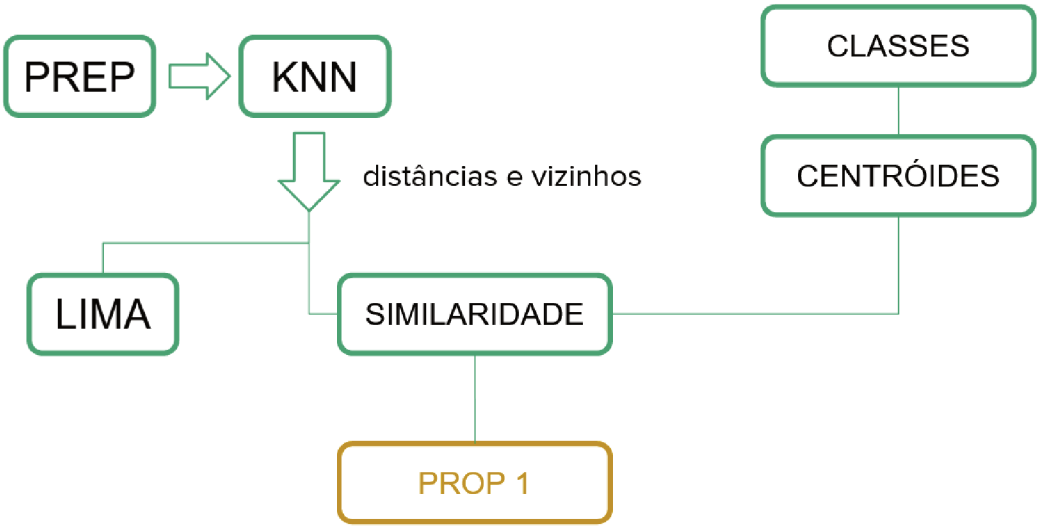
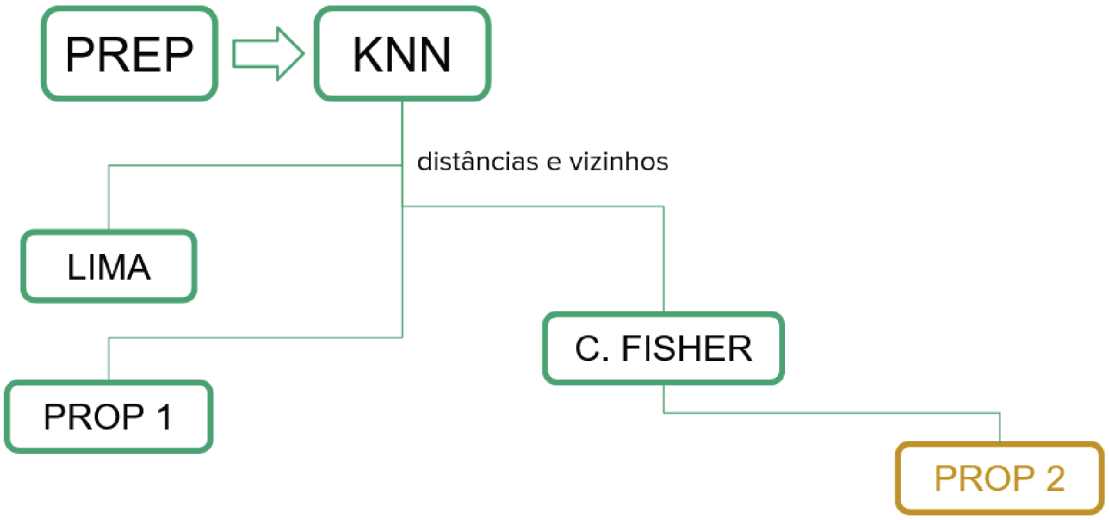


Figura 7 – Diagrama da extração do meta-atributo 2



5.3 MÉTRICAS E PARAMETRIZAÇÃO

Para avaliar a capacidade do método proposto e fazer-se possível a comparabilidade direta com o trabalho de Lima e Bianco (2019), foram utilizadas as mesmas configurações de parâmetros. A experimentação foi realizada utilizando o algoritmo de classificação *SVM* com os parâmetros em seus valores padrões, com exceção do *kernel*, em que "linear" foi a opção utilizada.

Para a extração de meta-atributos da vizinhança, foi utilizado o algoritmo *KNN*, com $K = 30$, o que indica o número máximo de vizinhos que podem ser encontrados para cada documento analizado.

A métrica para a avaliação dos resultados foi o *f1_weighted*, que mede a média harmônica entre a Precisão e Revocação (SASAKI et al., 2007), levando em consideração o peso das classes. Além do mais, é importante ressaltar que para garantir a confiabilidade dos testes, foi utilizada a validação cruzada de 10 vezes em cada conjunto de testes.

Para certificar a existência de mudanças significativas na comparação dos resultados, também foi utilizado o *Student's T-Test* (RAJU, 2005) com 10 amostras na medida do *f-score*.

5.4 RESULTADOS E DISCUSSÕES

Nesta seção serão apresentados o produto das execuções dos experimentos descritos previamente. Cada subseção é destinada para uma base de dados específica. Além disso, para medir ganho, perda e empate estatístico, foram criados os seguintes conjuntos de testes para comparação dos resultados:

1. BOW **X** LIMA+PROP;
2. LIMA **X** LIMA+PROP;
3. BOW+LIMA **X** BOW+LIMA+PROP;
4. BOW **X** BOW+LIMA+PROP.

O conjunto de testes BOW **X** LIMA+PROP compara o a utilização das características extraídas pelo modelo *bag-of-words* com a utilização da proposta apresentada por Lima e Bianco (2019) (definida como "LIMA") somadas a este trabalho (definida como "PROP"). Já o conjunto de testes LIMA **X** LIMA+PROP compara a eficiência da classificação utilizando

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	**69.87**%	**68.82**%	**69.13**%	**69.63**%	**69.68**%
lower_1G_FS_PREP	75.13%	78.73%	75.88%	**79.40**%	**79.07**%
lower_1G_2G_PREP	70.19%	68.22%	67.97%	**70.67**%	**70.49**%
lower_1G_2G_FS_PREP	77.25%	80.43%	82.59%	81.62%	**82.80**%
lower_1G_2G_3G_PREP	69.11%	68.33%	67.69%	**70.12**%	**70.12**%
lower_1G_2G_3G_FS_PREP	77.73%	**84.73**%	**84.28**%	**84.97**%	**84.93**%

Tabela 8 – Pontuação *F1* obtida nos experimentos na base *OffComBR-2*

somente os atributos extraídos em LIMA contra LIMA complementado com PROP. O conjunto BOW+LIMA **X** BOW+LIMA+PROP tem o objetivo de medir a diferença entre a utilização do BOW e LIMA juntos, contra eles mesmos complementados pelas características PROP. Ao final, a última configuração BOW **X** BOW+LIMA+PROP mede puramente o uso do BOW contra a proposta deste trabalho.

Os melhores resultados exibidos na Tabela 8, na Tabela 10 e na Tabela 12 estão com a fonte em negrito. O empate estatístico é indicado por experimentos que possuem mais de um resultado em negrito. Por exemplo, na Tabela 8 tem-se empate estatístico no experimento *lower_1G_2G_3G_PREP* entre *BOW+LIMA* e *BOW+LIMA+PROP*.

5.4.1 Experimentos na base *OffComBR-2*

Conforme apresentado na Tabela 8, quando compara-se somente os meta-atributos de LIMA+PROP com BOW pode-se observar que na configuração *lower_1G_2G_3G_FS_PREP* apresentou ganho estatístico de até 7%. Já quando comparado BOW+LIMA+PROP com o BOW+LIMA, pode-se inferir um ganho de 1.18% em relação à *lower_1G_2G_FS_PREP*. Além disso, LIMA+PROP quando comparado com LIMA teve ganho de 2.85% na configuração *lower_1G_FS_PREP*. Por fim, quando comparado BOW X BOW+LIMA+PROP na configuração *lower_1G_2G_3G_FS_PREP*, tem-se ganho estatístico de 7.2%.

A partir deste experimento pode-se notar que os meta-atributos atingiram resultados positivos na maioria das configurações apresentadas. Além disso, os resultados que apresentaram melhoria foram os que possuíam menor número de características, como visto na Tabela 9.

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	4122	18	14	3767	3771
lower_1G_FS_PREP	257	18	14	139	136
lower_1G_2G_PREP	15898	18	14	12184	12188
lower_1G_2G_FS_PREP	266	18	14	184	175
lower_1G_2G_3G_PREP	29125	18	14	20058	20062
lower_1G_2G_3G_FS_PREP	276	18	14	147	166

Tabela 9 – Número de características das configurações na base *OffComBR-2*

5.4.2 Experimentos na base *OffComBR-3*

Conforme apresentado na Tabela 10, entre somente os meta-atributos de LIMA+PROP com BOW pode-se observar que na configuração *lower_1G_2G_FS_PREP* houve ganho estatístico de 3.79% por parte de LIMA+PROP. Já na configuração *lower_1G_2G_3G_FS_PREP*, BOW+LIMA+PROP tem ganho de 1.76% em relacão a BOW+LIMA. Além disso, LIMA+PROP comparado com LIMA obteve ganho de 1.42% no arranjo *lower_1G_2G_FS_PREP*. Finalmente, na configuração *lower_1G_2G_3G_FS_PREP*, as características BOW+LIMA+PROP superaram BOW com 5.16% de melhoria.

De acordo com o evidenciado nesses conjuntos de testes, pode-se inferir que os meta-atributos atingiram bons resultados em comparação ao *baseline* visto em (LIMA; BIANCO, 2019). Também é possível notar que a redução de características teve influência nos resultados, conforme apresentado na Tabela 11.

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	**76.61**%	71.95%	71.73%	**76.80**%	**76.99**%
lower_1G_FS_PREP	82.26%	84.46%	84.50%	86.05%	**87.24**%
lower_1G_2G_PREP	**76.78**%	71.95%	71.73%	**76.04**%	**76.67**%
lower_1G_2G_FS_PREP	82.25%	**86.04**%	84.62%	84.86%	**86.94**%
lower_1G_2G_3G_PREP	**76.98**%	72.12%	71.73%	75.84%	**76.40**%
lower_1G_2G_3G_FS_PREP	82.99%	85.96%	86.38%	86.39%	**88.15**%

Tabela 10 – Pontuação *F1* obtida nos experimentos na base *OffComBR-3*

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	3309	18	14	3323	3327
lower_1G_FS_PREP	147	18	14	113	125
lower_1G_2G_PREP	10535	18	14	10549	10553
lower_1G_2G_FS_PREP	146	18	14	107	120
lower_1G_2G_3G_PREP	17299	18	14	17313	17317
lower_1G_2G_3G_FS_PREP	146	18	14	105	111

Tabela 11 – Número de características das configurações na base *OffComBR-3*

5.4.3 Experimentos na base *GIBERT*

Em conformidade com os dados demonstrados na Tabela 12, a comparação entre LIMA+PROP com BOW apresenta ganho de até 4.96% para BOW na configuração de experimento *lower_1G_FS_PREP*. Para BOW+LIMA+PROP com BOW+LIMA, foi observado empate estatístico em todos os casos. Além do mais, LIMA+PROP quando comparado com LIMA obteve ganho estatístico de até 1.95% em *lower_1G_2G_3G_FS_PREP*. Ao final, BOW+LIMA+PROP na configuração *lower_1G_2G_3G_FS_PREP* alcançou até 2.73% de melhoria quando comparado com BOW.

Em conformidade com o que foi exibido neste experimento, pode-se notar que os meta-atributos obtiveram melhorias em alguns casos, mantendo o padrão dos melhores resultados correlacionados com a redução de atributos, como evidenciado na Tabela 13.

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	**71.23**%	70.13%	70.12%	**71.47**%	**71.68**%
lower_1G_FS_PREP	78.81%	73.85%	72.13%	**79.76**%	**79.82**%
lower_1G_2G_PREP	**72.43**%	69.28%	69.51%	**71.76**%	**71.98**%
lower_1G_2G_FS_PREP	79.22%	74.92%	73.17%	**80.49**%	**80.69**%
lower_1G_2G_3G_PREP	**70.64**%	68.94%	68.94%	**70.98**%	**71.54**%
lower_1G_2G_3G_FS_PREP	78.40%	76.00%	74.05%	**80.90**%	**81.13**%

Tabela 12 – Pontuação *F1* obtida nos experimentos na base *GIBERT*

Experimento	BOW	LIMA+PROP	LIMA	BOW+LIMA	BOW+LIMA+PROP
lower_1G_PREP	4645	18	14	4659	4663
lower_1G_FS_PREP	354	18	14	368	372
lower_1G_2G_PREP	20861	18	14	20875	20879
lower_1G_2G_FS_PREP	359	18	14	373	377
lower_1G_2G_3G_PREP	36782	18	14	36796	36800
lower_1G_2G_3G_FS_PREP	361	18	14	375	379

Tabela 13 – Número de características das configurações na base *GIBERT*

5.5 DISCUSSÃO SOBRE OS RESULTADOS

Analisando os experimentos expostos, pode-se inferir que os meta-atributos adicionados neste trabalho, juntamente com o aperfeiçoamento do pré-processamento dos dados, proporcionaram resultados promissores quando comparados ao método proposto por Lima e Bianco (2019). As configurações observadas que mais obtiveram ganhos em todas as bases de dados testadas foram *lower_1G_FS_PREP*, *lower_1G_2G_FS_PREP* e *lower_1G_2G_3G_FS_PREP*. Os maiores benefícios observados foram obtidos pelo pré-processamento dos dados, melhorando os resultados em até 9.67% em relação ao baseline (LIMA; BIANCO, 2019). A base de dados que obteve melhores resultados foi a *OffComBr-3*, isto pode-se dar ao fato de ela possuir características mais significativas e/ou possuir tamanho reduzido quando comparado a *OffComBr-2* ou *GIBERT*.

Comparando os resultados entre a utilização do *bag-of-words* e somente os meta-atributos propostos complementados com os utilizados por Lima e Bianco (2019), pode-se observar resultados similares e em alguns casos melhores do que somente utilizando o método BOW. Desse modo, é preferível utilizar somente meta-atributos em alguns casos, principalmente levando em conta o fato de que o método BOW gera muito mais características, e, consequentemente mais processamento é necessário, quando comparado a extração de meta-atributos, a qual possui resultado similar com uma quantidade baixa de características extraídas, demonstrando assim a eficácia do método proposto e a baixa relevância proporcional ao número de características extraídas utilizando o método *bag-of-words*. Todavia, também há alguns casos onde ocorre a perda de pontuação quando utilizado somente os meta-atributos, demonstrando assim que deve haver um equilíbrio e análise do problema a ser resolvido, para saber quais atributos selecionar para obter melhores resultados.

6 CONCLUSÃO

O desenvolvimento do presente trabalho de conclusão de curso teve como objetivo analisar e obter formas de aperfeiçoamento na detecção de discursos de ódio em textos. Para isso, foram propostos o acréscimo de 2 meta-atributos para cada classe do conjunto de dados, além do processamento de dados antes do treinamento. Ademais, métodos de processamento de linguagem natural e aprendizagem de máquina foram utilizados.

Para a incorporação de mais meta-atributos, foi utilizado o método apontado no trabalho de Sérgio Canuto, L. F. Gonçalves et al. (2013), que utiliza informações da vizinhança de um documento para extrair características relevantes. Além de tudo, os meta-atributos acrescentados a este trabalho foram descritos por Sergio Canuto et al. (2014) e baseados no método proposto por Lima e Bianco (2019).

Para avaliar a eficácia do método proposto, múltiplos experimentos foram efetuados nas bases de dados propostas por Pelle e Moreira (2017) e Gibert et al. (2018). Obteve-se resultados promissores no acréscimo dos meta-atributos propostos e no aprimoramento do pré-processamento dos dados.

6.1 TRABALHOS FUTUROS

Para futuras complementações nesse trabalho, várias abordagens podem ser realizadas para a melhoria dos resultados.

Como visto em (Sergio CANUTO et al., 2014), mais meta-atributos podem ser adicionados para promover a melhora na eficácia do método. Além disso, em (Sérgio CANUTO; M. A. GONÇALVES; BENEVENUTO, 2016) a adição de atributos que levem em consideração o sentimento exposto em um texto também se mostrou uma abordagem promissora.

Finalmente, outro meio de contribuir com este trabalho é com a construção de novas bases de dados rotuladas. A quantidade de trabalho necessária para classificação manual de textos torna a escassez desses conjuntos de dados um problema real, visto que para a obter resultados suficientemente bons para a aplicação de tal método no mercado, exige-se uma quantidade de dados significativa.

REFERÊNCIAS

ALMEIDA, Thais G; NAKAMURA, Fabıola G; NAKAMURA, Eduardo F. Uma abordagem para identificar e monitorar haters em redes sociais online. In: SBC. ANAIS Estendidos do XXIII Simpósio Brasileiro de Sistemas Multimıdia e Web. [S.l.: s.n.], 2017. p. 41–46.

ALVARES, Reinaldo Viana. **ALGORITMOS DE STEMMING E O ESTUDO DE PROTEOMAS**. 2014. Tese (Doutorado) – Universidade Federal do Rio de Janeiro.

ANTON, Howard. **Elementary Linear Algebra**: Applications version. 10. ed. [S.l.]: Wiley, 2010. 1276 p.

BACCIANELLA, Stefano; ESULI, Andrea; SEBASTIANI, Fabrizio. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: 2010. LREC. [S.l.: s.n.], 2010. v. 10, p. 2200–2204.

BLUM, Avrim L; LANGLEY, Pat. Selection of relevant features and examples in machine learning. **Artificial intelligence**, Elsevier, v. 97, n. 1-2, p. 245–271, 1997.

BREIMAN, L. et al. **Classification and Regression Trees**. [S.l.]: Taylor & Francis, 1984. (The Wadsworth and Brooks-Cole statistics-probability series). ISBN 9780412048418. Disponível em: <https://books.google.com.br/books?id=JwQx-WOmSyQC>.

BRUGGER, Winfried. Proibição ou proteção do discurso do ódio? Algumas observações sobre o direito alemão e o americano. IOB; IDP, 2007.

CANUTO, Sérgio; GONÇALVES, Luiz Felipe et al. Um estudo sobre meta-atributos para classificacao automatica de texto, 2013.

CANUTO, Sérgio; GONÇALVES, Marcos André; BENEVENUTO, Fabrıcio. Exploiting New Sentiment-Based Meta-level Features for Effective Sentiment Analysis. In: PROCEEDINGS of the Ninth ACM International Conference on Web Search and Data Mining. San Francisco, California, USA: ACM, 2016. (WSDM '16), p. 53–62. ISBN 978-1-4503-3716-8. DOI: 10.1145/2835776.2835821. Disponível em: <http://doi.acm.org/10.1145/2835776.2835821>.

CANUTO, Sergio et al. On efficient meta-level features for effective text classification. In: ACM. PROCEEDINGS of the 23rd ACM International Conference on Conference on Information and Knowledge Management. [S.l.: s.n.], 2014. p. 1709–1718.

CHEN, Ying et al. Detecting offensive language in social media to protect adolescent online safety. In: IEEE. 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing. [S.l.: s.n.], 2012. p. 71–80.

DAMASHEK, Marc. Gauging similarity with n-grams: Language-independent categorization of text. **Science**, American Association for the Advancement of Science, v. 267, n. 5199, p. 843–848, 1995.

ELPAÍS. **Os robôs do Facebook aprendem sozinhos e mais rápido que nunca**. 2019. Disponível em: <https://brasil.elpais.com/brasil/2019/05/22/tecnologia/1558532554_995798.html>. Acesso em: 25 maio 2019.

GIBERT, Ona de et al. Hate Speech Dataset from a White Supremacy Forum. In: PROCEEDINGS of the 2nd Workshop on Abusive Language Online (ALW2). Brussels, Belgium: Association for Computational Linguistics, out. 2018. p. 11–20. DOI: 10.18653/v1/W18-5102. Disponível em: <https://www.aclweb.org/anthology/W18-5102>.

GOPAL, Siddharth; YANG, Yiming. Multilabel classification with meta-level features. In: ACM. PROCEEDINGS of the 33rd international ACM SIGIR conference on Research and development in information retrieval. [S.l.: s.n.], 2010. p. 315–322.

HUANG, Anna. Similarity measures for text document clustering. In: PROCEEDINGS of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand. [S.l.: s.n.], 2008. v. 4, p. 9–56.

HUTTO, Clayton J; GILBERT, Eric. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: EIGHTH international AAAI conference on weblogs and social media. [S.l.: s.n.], 2014.

LIMA, Cleiton; BIANCO, Guilherme Dal. Extração de característica para identificação de discurso de ódio em documentos. In: ANAIS da XV Escola Regional de Banco de Dados. Chapecó: SBC, 2019. p. 61–70. DOI: 10.5753/erbd.2019.8479. Disponível em: <https://sol.sbc.org.br/index.php/erbd/article/view/8479>.

LIXIN XU; GUANG CHEN; LEI YANG. Incremental clustering in short text streams based on BM25. In: 2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems. [S.l.: s.n.], nov. 2014. p. 8–12. DOI: 10.1109/CCIS.2014.7175694.

MAGLOGIANNIS, Ilias G. **Emerging Artificial Intelligence Applications in Computer Engineering**: Real Word AI Systems with Applications in EHealth, HCI, Information Retrieval and Pervasive Technologies. 160. ed. [S.l.]: IOS Press, 2007. 407 p.

NOBATA, Chikashi et al. Abusive language detection in online user content. In: INTERNATIONAL WORLD WIDE WEB CONFERENCES STEERING COMMITTEE. PROCEEDINGS of the 25th international conference on world wide web. [S.l.: s.n.], 2016. p. 145–153.

NOYES, Dan. **The Top 20 Valuable Facebook Statistics – Updated May 2019**. 2019. Disponível em: <https://zephoria.com/top-15-valuable-facebook-statistics/>. Acesso em: 25 maio 2019.

PELLE, Rogers P. de; MOREIRA, Viviane P. Offensive Comments in the Brazilian Web: a dataset and baseline results, 2017.

QUINLAN, J. Ross. Induction of decision trees. **Machine learning**, Springer, v. 1, n. 1, p. 81–106, 1986.

RAJU, Tonse NK. William Sealy Gosset and William A. Silverman: two "students" of science. **Pediatrics**, Am Acad Pediatrics, v. 116, n. 3, p. 732–735, 2005.

RAMOS, Juan 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.: s.n.], 2003. v. 242, p. 133–142.

RASCHKA, Sebastian. **Python Machine Learning**. [S.l.]: Packt Publishing, 2015. ISBN 1783555130, 9781783555130.

SASAKI, Yutaka et al. The truth of the F-measure. **Teach Tutor mater**, v. 1, n. 5, p. 1–5, 2007.

SCHMIDT, Anna; WIEGAND, Michael. A survey on hate speech detection using natural language processing. In: PROCEEDINGS of the Fifth International Workshop on Natural Language Processing for Social Media. [S.l.: s.n.], 2017. p. 1–10.

THELWALL, Mike. The Heart and soul of the web? Sentiment strength detection in the social web with SentiStrength. In: CYBEREMOTIONS. [S.l.]: Springer, 2017. p. 119–134.

TRIPATHI, Mayank. **How to process textual data using TF-IDF in Python**. Edição: freeCodeCamp. 2018. Disponível em: <https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/>. Acesso em: 16 jun. 2019.