UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

ROGERS PRATES DE PELLE

# IDENTIFICAÇÃO DE COMENTÁRIOS OFENSIVOS NA WEB

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Profa. Dra. Viviane P. Moreira

Porto Alegre
2019

**CIP — CATALOGAÇÃO NA PUBLICAÇÃO**

Pelle, Rogers Prates de

IDENTIFICAÇÃO DE COMENTÁRIOS OFENSIVOS NA WEB / Rogers Prates de Pelle. – Porto Alegre: PPGC da UFRGS, 2019.

59 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2019. Orientador: Viviane P. Moreira.

1. Identificação de comentários ofensivos. 2. Classificação de texto. 3. Processamento de Linguagem Natural. 4. Word Embeddings. I. Moreira, Viviane P.. II. Título.

*“Tweet others the way you want to be tweeted”* — GERMANY KENT

## AGRADECIMENTOS

A Deus pela vida e força pra segui-la.

Ao Instituto de Informática da Universidade Federal do Rio Grande do Sul por meio do Programa de Pós-Graduação em Computação por me conceder a oportunidade de chegar ao mestrado, mesmo vindo de um lugar onde raramente as pessoas conseguem chegar a graduação.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico que possibilitou o desenvolvimento desta pesquisa através da concessão da bolsa de estudos na modalidade de mestrado.

A minha orientadora Profª. Dra. Viviane Pereira Moreira, que acreditou no meu potencial mesmo vindo de uma universidade pequena do interior do Mato Grosso do Sul e se dedicou ao máximo para que este trabalho fosse concluído da melhor forma possível. E aos demais professores do programa por contribuírem de alguma forma para o desenvolvimento do mesmo.

Aos colegas Lucas Pessutto, Paula Burguêz, Cassio Garcia, Diego Feijó, que colaboraram com rotulação dos dados para a construção do *dataset*.

Aos colegas do Edimar Manica, Danny Vargas, Cleber Alcântara e Vinicius Dani pela prazerosa convivência no laboratório 215.

A minha esposa Geisiane Martini, família, amigos e principalmente aos meus pais, Carlos Alberto de Pelle e Márcia Aparecida Prates, que tiveram que deixar seus sonhos de lado, para que eu realizasse o meu.

## RESUMO

Com a Web 2.0, os usuários deixaram de ser apenas consumidores da informação disponível e passaram a ser autores da maior parte do conteúdo produzido. Usuários postam suas opiniões sob a forma de blogs, tweets, posts em redes sociais e comentários em portais de notícias. Postagens ofensivas são um incômodo constante em muitas plataformas da Web e vêm causando constrangimentos, brigas e processos judiciais. Como consequência, tem havido um crescente interesse em criar métodos para identificar automaticamente este tipo de conteúdo. A identificação automática de conteúdo ofensivo é uma tarefa desafiadora que precisa lidar com uma série de questões tais como: as diversas formas que as ofensas podem ser escritas; o fato de que os autores costumam disfarçar palavrões para tentar burlar os filtros; a dinamicidade do vocabulário da Internet, entre outas. Neste trabalho, é proposta uma abordagem para detectar comentários ofensivos na Web, denominada `Hate2Vec`, que é composta por um *ensemble* de classificadores no qual um meta-classificador decide se um comentário é ou não ofensivo com base na saída de três classificadores base: ($i$) um classificador baseado em léxico que utiliza a proximidade semântica das representações vetoriais de palavras; ($ii$) um classificador de regressão logística baseado em representações vetoriais de comentários; e ($iii$) um classificador *bag-of-words* baseado nos uni-gramas do texto. Nos experimentos realizados com conjuntos de dados em inglês e português, o `Hate2Vec` produziu bons resultados de classificação (medida F acima de 0,9) e superaram significativamente o *baseline*.



**Palavras-chave:** Identificação de comentários ofensivos. Classificação de texto. Processamento de Linguagem Natural. Word Embeddings.

**Identification of offensive comments on the web**


## ABSTRACT

With Web 2.0, users went from being consumers of the available information to becoming the authors of most of the content produced. Users post their opinions in the form of blogs, tweets, posts on social networks, and comments on news portals. Offensive posts are a constant nuisance on many web platforms and have been causing embarrassment, arguments and litigation. As a consequence, there has been a growing interest in creating methods to automatically identify this type of content. Automatically identifying offensive content is a challenging task that needs to address a range of issues such as: the various ways that offenses can be written; the fact that the authors usually disguise profanity to try to circumvent the filters; the dynamism of the Internet vocabulary, among others. In this work, we propose `Hate2Vec` an approach to detect offensive comments on the Web. `Hate2Vec` is composed of a classifier's ensemble in which a meta-classifier decides whether or not a comment is offensive based on the output of three base classifiers: $(i)$ a lexicon-based classifier which leverages the semantic relatedness of word embeddings; $(ii)$ a logistic regression classifier based on comment embeddings; $(iii)$ and a standard bag-of-words classifier based on unigram features. Our experiments with datasets in English and Portuguese have yielded high classification results (F-measure above 0.9) and significantly outperformed a traditional BOW classifier used as baseline.



**Keywords:** Identification of offensive comments, Text Classification, Natural Language Processing, Word Embeddings .

## LISTA DE ABREVIATURAS E SIGLAS

**BOW** *Bag of Words*

**CBOW** *Continuous Bag of Words*

**CNN** *Convolutional Neural Networks*

**GPU** Unidades de Processamento Gráfico

**LSTM** *Long Short Term Memory*

**NER** *Named Entity Recognition*

**PT-BR** Português do Brasil

**POS** *Parts of Speech*

**PLN** Processamento de Linguagem Natural

**PV-DM** *Distributed Memory version of Paragraph Vector*

**ROC** *Receiver Operating Characteristic*

**ROC-AUC** *Area Under the ROC Curve*

**SVM** *Support Vector Machines*

**VSM** *Vector Space Models*

## LISTA DE FIGURAS

## LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

A forma com que a sociedade consome informação e se comunica, mudou drasticamente com o surgimento da Web. Em sua fase inicial, os usuários da Web eram basicamente consumidores das informações disponíveis. Contudo, a Web vem passando por constantes mudanças. As páginas que eram estáticas tornaram-se dinâmicas, usuários que antes apenas consumiam conteúdo, hoje produzem a maior parte da informação que lá circula. Esse fenômeno é chamado de Web 2.0 (AGHAEI; NEMATBAKHSH; FARSANI, 2012).

As redes sociais são parte fundamental desta revolução, pois ampliam a capacidade e o alcance da comunicação e expressão de opiniões. Redes sociais são utilizadas em grande parte como canais coletivos de comunicação, nos quais os usuários podem interagir, compartilhando conteúdo e construindo comunidades com interesses em comum. A natureza descentralizada torna as redes sociais um local para o compartilhamento de ideias, informações e mídias. Entretanto, elas também têm sido utilizadas para fins prejudiciais, como por exemplo o discurso do ódio.

Nockleby (2000) define o discurso de ódio como qualquer comunicação que deprecie uma pessoa ou um grupo com base em alguma característica como raça, cor, etnia, gênero, orientação sexual, nacionalidade, religião ou outras características. Segundo Silva et al. (2011), o discurso de ódio é uma manifestação de segregação, onde o emissor do discurso, atinge ao outro baseado em uma suposta relação de superioridade. Esta manifestação é efetivada quando é transmitida para outro indivíduo seja ele a vitima ou terceiros, por qualquer que seja o meio, inclusive *online*.

Os mesmos aspectos que tornam a Web uma ferramenta de grande utilidade, também possibilitam a disseminação do discurso do ódio. A natureza democrática da Web dificulta a censura deste tipo de conteúdo, o direito à privacidade gera uma sensação de impunidade e o grande alcance sugere que pessoas que compartilham destes pensamentos se juntem em comunidades. Antes das redes sociais, todos estes grupos já existiam, mas com maior alcance, as consequências dessas ações também são amplificadas.

No Brasil, textos ofensivos são postados em redes sociais diariamente, embora todos sejam afetados por este problema, o alvo principal são as minorias: negros [1], gays [2],

[1] <https://glo.bo/2GdbDSB>
[2] <http://bit.ly/2UCJwzh>

mulheres[3]. A questão tem chamado a atenção inclusive da classe política [4] e em outros países tem sido tratado como caso de saúde pública[5].

O discurso de ódio não se limita às redes sociais. Em uma análise preliminar, realizada no dia 7 de junho de 2016 pelo autor deste trabalho, foi coletada uma amostra de 145 notícias publicadas em um único dia no site de notícias com maior quantidade de acessos no Brasil[6]. Foram analisados os comentários destas notícias, e foi constatado que 90% das notícias teve pelo menos um comentário ofensivo. Na maior parte dos casos, os usuários começam a discutir o conteúdo das notícias e acabam se envolvendo em discussões que usam linguagem abusiva. Embora as empresas tenham mecanismos para impedir a publicação de textos ofensivos, ainda não conseguem impedir casos como os já citados.

A empresa Agência Nova/SB [7] realizou um estudo sobre intolerância em redes sociais. Durante três meses, eles monitoraram diversos blogs, sites e redes sociais, foram registradas todas as ocorrências de postagens em que assuntos como racismo e homofobia eram mencionados, sendo coletadas um total de 542.781 menções. Suas descobertas relatadas em NoavaS/B (2016), afirmam que 84% dos comentários sobre esses assuntos são negativos, tendo como base ferramentas de análise de sentimento.

Empresas que administram as redes sociais estão sendo obrigadas a aumentar os esforços para combater a proliferação de conteúdo extremista e o discurso de ódio em suas plataformas, diante das novas legislações [8]. Com isso, várias empresas têm tomado a iniciativa no combate ao discurso de ódio, como é o caso do Google[9], Twiter[10], Facebook e Instagram[11], mas muitas das vezes sem sucesso [12].

A verificação de tudo que é publicado em redes sociais de forma manual, ou seja, por agentes humanos, é uma tarefa inviável, uma vez que o volume de publicações é muito maior do que qualquer disponibilidade de agentes. Em 2012, 2,5 bilhões de pessoas estavam conectadas à Internet, em 2017, esse número saltou para 3,8 bilhões. Por minuto, cerca de 500 mil postagens são realizas apenas no Twitter [13]. Diante deste quadro, são necessárias ferramentas que identifiquem automaticamente textos ofensivos em meio a

[3]<https://glo.bo/2SDZzLy>
[4]<http://bit.ly/2L88vWS>
[5]<http://bit.ly/2Eg6aYL>
[6]<www.g1.globo.com>
[7]<http://www.novasb.com.br>
[8]<https://reut.rs/2mTdtvF>
[9]<https://www.bbc.com/news/technology-39707642>
[10]<https://blog.twitter.com/official/en_us/topics/company/2017/safetycalendar.html>
[11]<https://bit.ly/2UA80t5>
[12]<https://reut.rs/2vKN0ov>
[13]<https://www.domo.com/learn/data-never-sleeps-6>

essa grande quantidade de dados.

A necessidade destas ferramentas é tão evidente que várias campanhas de avaliação estão sendo criadas, como é o caso da competição SemEval 2019 que conta com duas tarefas baseadas neste tema, a Task 5[14], que consiste em uma tarefa de detecção de discurso de ódio no Twitter, caracterizada por dois alvos específicos (imigrantes e mulheres), e em uma perspectiva multilíngue (espanhol e inglês). Há também a Task 6[15], que propõe três sub-tarefas: a de identificação de linguagem ofensiva, categorização automática de tipos de ofensa e a identificação de alvo de ofensa. Outro exemplo é Toxic Comment Classification Challenge [16] onde os participantes são desafiados a desenvolver um modelo capaz de detectar diferentes tipos de toxicidade, como ameaças, obscenidades, insultos e ódio baseado em identidade.

Entretanto, realizar a identificação de forma automática, não é uma tarefa fácil, uma vez que os autores tendem a disfarçar palavras ofensivas inserindo asteriscos, espaços ou substituindo os caracteres por outros com sons semelhantes. Fazer a identificação somente com base na presença de termos que estão em uma lista pré-construída de ofensas (*dirty-word list*) deixaria de detectar muitos comentários. Outra questão importante é que muitas palavras só se tornam ofensivas dependendo do contexto em que estão inseridas, o que não seria contemplado pelo método citado acima.

Uma solução seria modelar esta tarefa como um problema de classificação de texto e gerar modelos que aprendessem automaticamente como identificar comentários ofensivos. Essa abordagem requer um conjunto de dados anotado com exemplos positivos e negativos (ou seja, comentários ofensivos e não ofensivos). Mesmo modelos mais robustos, estão sujeitos a impeditivos como: $(i)$ O idioma na Web é coloquial, abundante em gírias, abreviações e jargões específicos da Web; $(ii)$ A língua portuguesa possui poucos recursos linguísticos em comparação a outros idiomas; $(iii)$ Os métodos de detecção precisam ser rápidos, robustos e escaláveis para lidar com a grande quantidade de dados publicados diariamente; $(iv)$ O vocabulário da Web é muito dinâmico, portanto, os métodos de detecção precisam se adaptar continuamente; e $(v)$ Existe um limite impreciso entre a filtragem do conteúdo ofensivo e a interferência na liberdade de expressão das pessoas.

Diante do estudo realizado, constatou-se que não existia um *dataset* anotado com conteúdo ofensivo em português, este trabalho então elaborou os *datasets* OFFCOMBR-

[14]<https://competitions.codalab.org/competitions/19935>
[15]<https://competitions.codalab.org/competitions/20011>
[16]<https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge>

2 e o OFFCOMBR-3, que são descritos no Capítulo 4. A principal contribuição deste trabalho é o método `Hate2Vec`, que tem como objetivo detectar conteúdo ofensivo em comentários e postagens na Web. O método é baseado em um conjunto de classificadores no qual o meta-classificador se alimenta da saída dos outros três classificadores, combinando *embeddings* de palavras e documentos além de classificadores baseados nos termos dos comentários.

Nos experimentos realizados usando conjuntos de dados em português e inglês, a medida F variou entre 0,90 e 0,97, desempenho comparável ao estado da arte em detecção de texto ofensivo. Como os *datasets* em geral são desbalanceados, a obtenção de uma baixa taxa de falsos negativos que chegou a 0,04 também é um forte indicador da robustez do método.

O trabalho está organizado da seguinte forma: no Capítulo 2 as tecnologias utilizadas no desenvolvimento do método são introduzidas. No Capítulo 3, são discutidos os trabalhos relacionados. No Capítulo 4, é abordada a construção do *dataset*. No Capítulo 5, é apresentado o método identificação de comentários ofensivos desenvolvido. No Capítulo 6, são listados os resultados alcançados e as configurações dos experimentos. For fim, o Capítulo 7 traz uma reflexão sobre as contribuições e apresenta possibilidades de trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

Neste Capítulo são abordados os conceitos utilizados nos métodos desenvolvidos que são necessários para o entendimento deste trabalho. O primeiro conceito exposto é o de algoritmos de classificação que são utilizados para classificar os textos dos comentários como sendo ou não ofensivo. Outro conceito utilizado é o de *Word Embeddings*, que trata da representação vetorial do sentido semântico palavras.

## 2.1 Algoritmos de classificação

Algoritmos de classificação são técnicas de aprendizado de máquina, nas quais um programa de computador passa por um processo de aprendizado a partir de uma entrada de dados previamente rotulada. Posteriormente, esse aprendizado é utilizado para classificar novos dados (não rotulados). Este processo ocorre por meio do reconhecimento de padrões dos atributos (características) de cada instância, sendo as novas instâncias rotuladas com as classes correspondentes as padrões de seus atributos.

Algoritmos utilizados na classificação texto podem ser definidos como uma função $F$ onde, dado um conjunto de documentos $D$ e um conjunto de classes $C$, esta função atribui uma classe de $C$ para cada documento em $D$. Por exemplo, o conjunto $D$ pode ser formado por *tweets* coletados da Web e o conjunto $C$ pode possuir duas classes *positiva* ou *negativa*, que descrevem se o *tweet* possui polaridade positiva ou negativa em relação ao objeto de estudo.

Nestes classificadores de texto, os atributos utilizados costumam ser as palavras dos documentos. A abordagem *Bag of Words* (BOW), na qual o texto é representado pelo conjunto de suas palavras, sem considerar a ordem, é bastante utilizada. Outro método utilizado para representar palavras como atributos é o $n$-grama, que é uma maneira de combinar, em subconjuntos, um sequência de $n$ palavras ou caracteres do texto. Quando utilizado a nível de palavra, é possível analisar termos compostos, ou mesmo palavras que tem seu sentido alterado dependendo de onde e como é utilizada. Quando a abordagem BOW é utilizada a nível de caractere, permite analisar e agrupar palavras que foram alteradas por prefixos e/ou sufixos, mas que possuem o mesmo sentido.

Existe na literatura uma ampla variedade de algoritmos de classificação que podem ser aplicados em diversos fins, abaixo destacamos três algoritmos que são utilizados neste trabalho além de um método de combinação de classificadores.

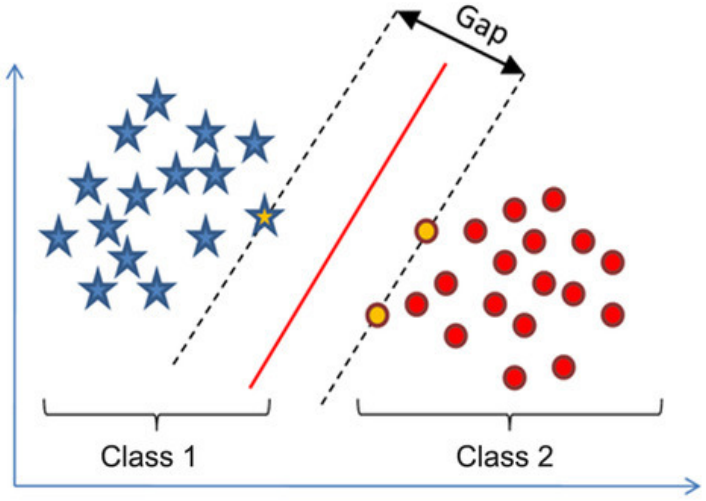
### 2.1.1 Naïve Bayes

Russell e Norvig (2016) apresentam Naïve Bayes como uma técnica de classificação baseada no Teorema de Bayes com uma suposição de independência entre atributos. Um classificador Naïve Bayes assume que a presença de um atributo particular em uma classe não está relacionada à presença de qualquer outro atributo. Desta forma, mesmo que um atributo dependa de outro, todos contribuem de igual forma para a probabilidade de uma instância pertencer a uma determinada classe. Pode ser aplicado em vários domínios, sendo um dos principais a classificação de texto.

### 2.1.2 SVM

O algoritmo *Support Vector Machines* (SVM) é um método de aprendizado supervisionado que pode ser usado para tarefas de classificação ou regressão. O treinamento baseia-se em representar cada instância da entrada como um ponto em um espaço $n$-dimensional, onde $n$ é o número de atributos, com o valor de cada atributo sendo o valor de uma determinada coordenada. Então a classificação é realizada encontrando o hiperplano que diferencia as classes com a maior margem possível, como descrito por Boser, Guyon e Vapnik (1992).

Figura 2.1: Máquina de Vetores de Suporte




Fonte: https://www.quora.com/Could-someone-explain-this-joke-What-did-one-support-vector-say-to-another-I-feel-so-marginalized

A Figura 2.1 mostra uma representação do algoritmo SVM realizando uma clas-

sificação binária, na qual primeiramente são definidos os pontos pertencentes a cada uma das classes, em seguida as margens são maximizadas. Esta margem, que é a distância entre o primeiro ponto da classe e o hiperplano é chamada de *gap*. Caso exista um *outlier* (que são pontos que estão distantes do do agrupamento de pontos pertencentes a sua classe), o SVM realiza um tentativa adaptação dos vetores, mas caso não seja possível o *outlier* é desconsiderado.

### 2.1.3 Regressão Logística

Jr, Lemeshow e Sturdivant (2013) coloca que o a regressão logística tem por objetivo estimar a probabilidade de uma variável dependente assumir um determinado valor em função dos valores conhecidos de outras variáveis. Caso o problema de classificação seja binário, ou seja, a variável $Y$ assumir apenas dois possíveis estados, e haver um conjunto de $p$ variáveis independentes $X1$, $X2$, ..., $Xp$, onde *g(x)= B0 + B1X1 + ... + BpXp*, o modelo de regressão logística pode ser escrito da seguinte forma:

$$P(Y = 1) = \frac{1}{1 + e^{-g(x)}}$$

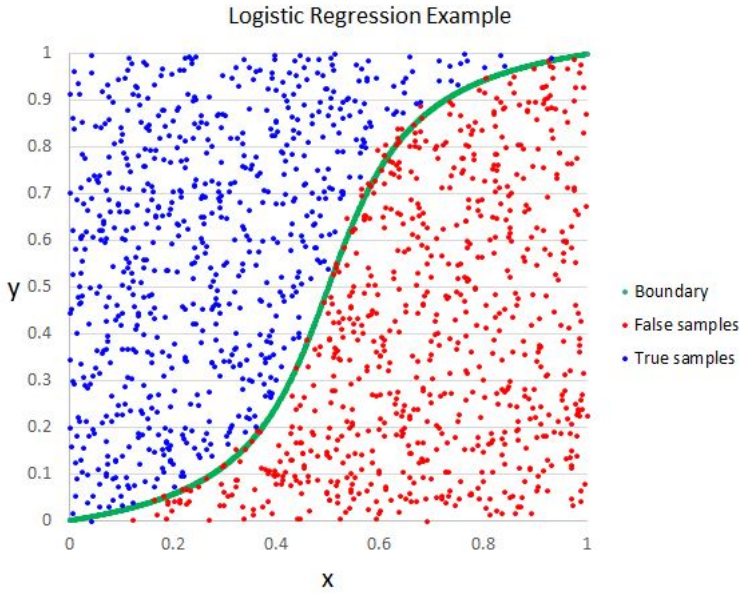
A função logística quando expressa graficamente se parece com um S, transformando qualquer valor no intervalo de 0 a 1. Isso é útil porque é possível aplicar uma regra à saída da função logística para ajustar valores para 0 e 1. Por exemplo, se o valor da saída for menor que 0,5 ela pertence a classe $X$. A Figura 2.2 apresenta um exemplo de regressão logística onde a curva verde representa a fronteira encontrada, e embora ela não consiga realizar uma separação perfeita das classes, obtém uma grande taxa de acertos.

### 2.1.4 Empilhamento de classificadores

Para alcançar melhores resultados nas tarefas de classificação, os classificadores podem ser combinados em um *meta-classificador*. A ideia é que aprender com múltiplos classificadores, combinando suas previsões, reduz a variância. Como consequência, os resultados são menos dependentes das peculiaridades de um único conjunto de treinamento ou algoritmo. Uma combinação de classificadores pode aprender uma classe conceitual mais expressiva do que um único classificador.

Uma forma de obter um *meta-classificador* é construindo um empilhamento de

Figura 2.2: Regressão Logística.




Fonte: https://helloacm.com/a-short-introduction-logistic-regression-algorithm/

classificadores, também de chamado *stacking*. No *stacking* vários *classificadores de base* são treinados e suas previsões são usadas como entrada para um *meta-classificador* que toma a decisão final sobre qual classe deve ser atribuída às instâncias (WOLPERT, 1992).

## 2.2 *Word Embeddings*

Algoritmos de PLN geralmente utilizam símbolos discretos para representar palavras. Por exemplo, a palavra `carro` pode ser representada como `id001`, `moto` como `id002`, *etc.* Essas codificações são arbitrárias e não fornecem informações úteis ao algoritmo sobre os relacionamentos que podem existir entre os símbolos individuais. Além disso, representar palavras como identificadores únicos e discretos leva à dispersão de dados. Utilizando representações vetoriais (*embedding*) de palavras é possível superar alguns desses obstáculos, pois os *Vector Space Models* (VSM) representam palavras em um espaço vetorial contínuo, onde palavras semanticamente semelhantes são mapeadas para pontos próximos.

A utilização de VSM em PLN permaneceu inviável por muito tempo, devido ao custo computacional, uma vez que alta dimensionalidade dos modelos propostos, exigia uma performance não alcançável pelo hardware existente. O avanço na capacidade de processamento e também a utilização de Unidades de Processamento Gráfico (GPUs) em cálculos matriciais, permitiram os primeiros experimentos utilizando VSM em grandes

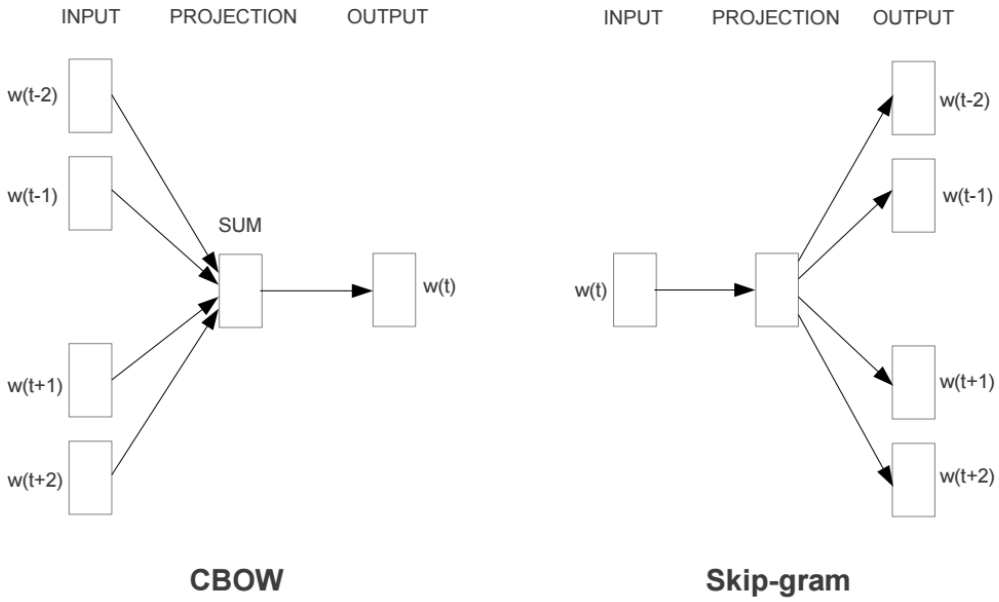
quantidades de texto, mas foi a redução de dimensionalidade aplicado a esta tarefa, proposta por Mikolov et al. (2013), que popularizou sua utilização por meio de uma técnica chamada Word2vec, que é um modelo de *embedding* de palavras a partir de texto bruto.

### 2.2.1 Word2vec

Sendo a alta dimensionalidade algo que inviabilizada o uso de *word embeddings*, Bengio et al. (2003) propõe aprender uma representação distribuída para palavras, para reduzir a dimensionalidade. Esta abordagem permite que cada sentença de treinamento informe ao modelo o número sentenças semanticamente vizinhas. Baseado nesse principio, (MIKOLOV et al., 2013) propôs o Word2vec como modelo eficiente para representação vetorial de palavras, tornando-se rapidamente muito utilizado pela comunidade.

Para realizar o mapeamento das palavras, o Word2Vec utiliza uma rede neural com uma camada intermediaria. A rede pode ser treinada de duas maneiras diferentes: *Continuous Bag of Words* (CBOW) ou Skip-Gram, apresentados na Figura 2.3. Os dois modelos são bastante similares, a diferença é que o CBOW prevê palavras-alvo a partir das palavras de contexto de origem, enquanto o Skip-Gram faz o inverso e prediz palavras de contexto de origem a partir das palavras-alvo.

Figura 2.3: Comparação entre o treinamento baseado em CBOW e Skip-Gram



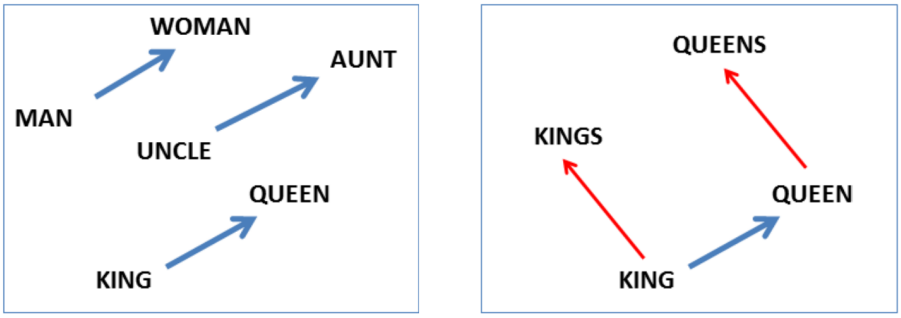

Fonte: (MIKOLOV et al., 2013)

O contexto é definido com uma janela de $w$ palavras, ou seja, o contexto de uma

palavra $x$ são as $w$ palavras que a antecedem e as $w$ palavras subsequentes. O tamanho de $w$ pode variar dependendo do domínio da aplicação, visto que as sentenças utilizadas no treinamento podem variar de tamanho.

O processo de treinamento parte de um arquivo de texto bruto, contendo uma grande quantidade de sentenças separadas por por algum caractere predefinido, geralmente \ $n$. É extraído um vocabulário com todas as palavras que aparecem no texto, e a cada palavra é atribuído um vetor que a representa. Para o cálculo deste vetor, são realizadas várias iterações em um rede neural.

Figura 2.4: Exemplos de relações em um espaço vetorial treinado pelo Word2Vec




Fonte: (MIKOLOV; YIH; ZWEIG, 2013)

A cada iteração na rede neural, é produzida uma camada intermediária que é a projeção da palavra no espaço vetorial, sua posição é calculada com base no seu contexto, ou seja, as palavras que a cercam. O resultado deste processo é apresentado na Figura 2.4 que mostra que palavras semanticamente relacionadas tem menor distância vetorial.

Estes vetores podem ser utilizados no cálculo da similaridade entre palavras utilizando distância de euclidiana ou de cosseno. A função de similaridade mais comumente encontrada nas bibliotecas é a de semelhança de cosseno, e é constituída pelo cálculo da distância dos vetores baseado no ângulo formado entre eles. Para encontrar essa distância é necessário aplicar a equação de multiplicação de pontos dos vetores, multiplicando dois vetores para produzir um único valor escalar como representado na fórmula abaixo (LEVY; GOLDBERG, 2014).

$$\cos(u, v) = \frac{u \cdot v}{\|u\| \, \|v\|}$$

Ao utilizar a semelhança de cosseno, quando um ângulo de ângulo de 90 graus é formado entre dois vetores de palavras, nenhuma semelhança é expressa, enquanto a similaridade total de 1 é um ângulo de 0 graus, ou seja, quando ocorre a sobreposição

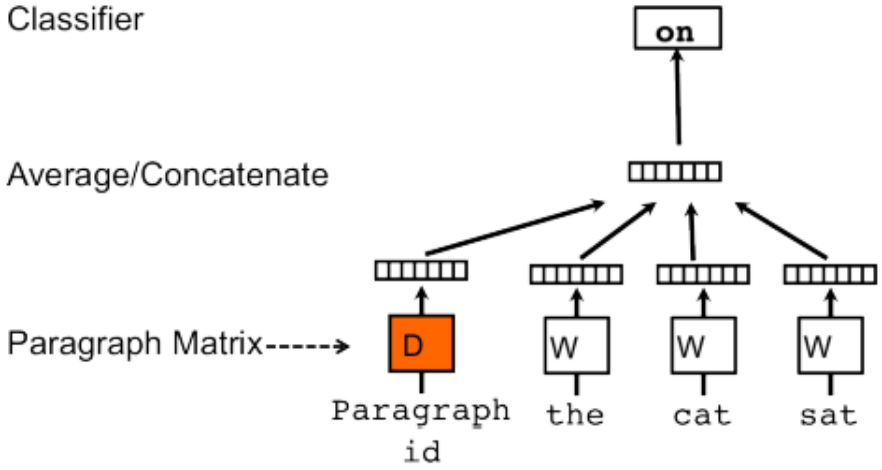
completa. Esta função dá origem a duas formas de obter alta similaridade: se as palavras aparecem com frequentemente juntas e principalmente se duas palavras podem ser usadas no mesmo lugar sem alterar o sentido da frase

### 2.2.2 Doc2Vec

O trabalho de Le e Mikolov (2014) propôs uma extensão ao Word2vec chamada de *Paragraph Vectors* (ou Doc2Vec) que é uma abordagem de aprendizado de VSM para representar sentenças de texto (parágrafos ou trechos de documentos). Para representar sentenças de vários tamanhos de forma numérica e uniforme, foi utilizado o mesmo modelo do Word2Vec adicionando um novo vetor, chamado *Paragraph ID* como mostra a Figura 2.5.

Este modelo é extensão do CBOW, mas em vez de usar apenas palavras para prever a próxima palavra, também utiliza o vetor *Paragraph ID*, que é exclusivo do documento. Assim, ao treinar os vetores de palavras, o vetor do documento também é treinado e, ao final do treinamento, contém uma representação vetorial do documento. Este modelo é chamado de *Distributed Memory version of Paragraph Vector* (PV-DM).

Figura 2.5: Representação do modelo de treinamento do Doc2Vec




Fonte: (LE; MIKOLOV, 2014)

Assim como no treinamento do Word2Vec, o treinamento Doc2Vec também é baseado em uma rede neural, que a cada iteração promove a atualização dos vetores. Este treinamento pode ainda ser enriquecido com um corpus de de palavras já treinado, obtendo assim uma melhor representação de palavras e consequentemente do um melhor representação de documentos.

## 2.3 Métricas de avaliação

A avaliação do desempenho de um modelo de aprendizado de máquina é uma etapa essencial em um pipeline de modelagem preditiva. Geralmente o modelo é avaliado por sua precisão estatística, ou seja, se as premissas estatísticas estão corretas, o desempenho do modelo é avaliado positivamente. Existem várias métricas de avaliação, cada uma aplicada a diferentes domínios.

As primeiras métricas a serem extraídas dos resultados geralmente são TP, TN, FP, e FN, que referem-se a verdadeiro positivo, verdadeiro negativo, falso positivo, e falso negativo, respectivamente. Estas métricas formam a chamada matriz de confusão, da qual é possível extrair medidas mais abrangentes como: precisão (p), revocação (r) e medida F (POWERS, 2011).

A precisão é a fração de instâncias recuperadas que pertencem à classe esperada, já a revocação é a fração de instâncias pertencentes à classe esperada, que são recuperadas. Essas métricas são micro-médias que comparam os resultados obtidos contra os resultados esperados e todas consideram que as instâncias têm pesos iguais, ambas as são descritas abaixo:

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

$$r = \frac{TP}{TP + FN}$$

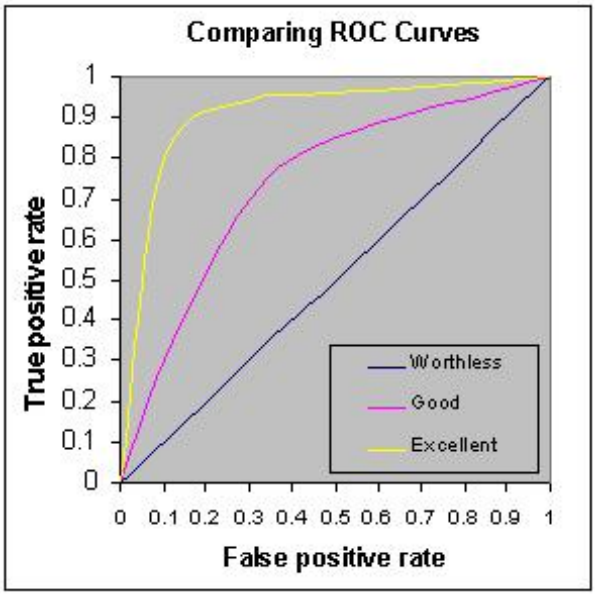
Goutte e Gaussier (2005) descrevem a medida F (F) como uma métrica de avaliação que leva em consideração a precisão (p) e a revocação (r), em que uma medida F alcança seu melhor valor em 1 e o pior escore em 0. O peso da precisão e da revocação na medida F é exatamente o mesmo, sendo assim considerada uma média harmônica entre as duas métricas. A fórmula para a o cálculo da medida F é:

$$F = 2 \cdot \frac{p \cdot r}{p + r}$$

ROC é um gráfico que mostra o desempenho de um modelo de classificação em todos os limites de classificação. Esta curva mostra dois parâmetros: Taxa Positiva Verdadeira, Taxa positiva falsa, apresentadas na Figura 2.6. ROC-AUC mede toda a área bidimensional por baixo de toda a curva ROC de (0,0) a (1,1), fornecendo assim uma métrica agregada de desempenho em todos os possíveis limiares de classificação Bradley

(1997).

Figura 2.6: Comparação de curvas ROC




Fonte: <http://gim.unmc.edu/dxtests/roc3.htm>

## 2.4 Sumário do Capítulo

Neste Capítulo foram apresentados algoritmos de classificação e bem como técnicas de *word embeddings* utilizadas neste trabalho. Em termos gerais, *word embeddings* tem apresentado melhores resultados em relação aos algoritmos de classificação de texto para variadas tarefas. No entanto, um combinação destas técnicas pode produzir resultados mais robustos, sendo o método proposto, uma tentativa de validar esta hipótese. Os fundamentos descritos neste Capítulo promovem o entendimento de métodos do estado da arte e do trabalho desenvolvido. No Capítulo que se segue, são apresentados trabalhos com objetivos e/ou técnicas similares aos propostos por este trabalho.

# 3 TRABALHOS RELACIONADOS

O aumento nos casos de discurso de ódio citado na Capítulo 1, tem motivado o surgimento de trabalhos que objetivam identificar automaticamente estas ocorrências nas redes sociais, blogs, espaços de comentários em sites *etc.* Nas Seções a seguir, são apresentados os principais trabalhos da literatura que tratam de criar *datasets* com conteúdo ofensivo e/ou que classificam este tipo de conteúdo.

## 3.1 Coleta e anotação dos dados

Em qualquer tarefa de classificação de texto, é essencial a utilização de dados devidamente rotulados, para realizar a avaliação dos métodos, e no caso de modelos de aprendizado supervisionado, estes dados também são utilizados na fase de treinamento. Entretanto, ainda não há um corpus de referência comumente aceito para a tarefa, sendo assim, geralmente cada autor coleta e rotula seus próprios dados.

### 3.1.1 Dados utilizados em *datasets*

A maior parte do esforço na detecção de textos ofensivos está sendo realizada para língua inglesa. No entanto, recentemente alguns trabalhos se propuseram a realizar esta tarefa em outros idiomas, como Alemão (BRETSCHNEIDER; PETERS, 2017; BURNAP PETEAND WILLIAMS, 2016; ROSS et al., 2016), Holandês (TULKENS et al., 2016; HEE et al., 2015), Italiano (VIGNA et al., 2017), Esloveno (FIŠER; ERJAVEC; LJUBEŠIĆ, 2017).

Várias fontes de dados são utilizadas, em sua maioria redes sociais, destacando-se o **Twitter**[1] por fornecer fácil acesso aos dados (MAGU; JOSHI; LUO, 2017; KENNEDY et al., 2017; HASANUZZAMAN; DIAS; WAY, 2017; GAO; KUPPERSMITH; HUANG, 2017; DUCHARME, 2017; DAVIDSON et al., 2017; WASEEM; HOVY, 2016; ROSS et al., 2016; BURNAP PETEAND WILLIAMS, 2016; AL-GARADI; VARATHAN; RAVANA, 2016; PETE; L., 2015; BURNAP; WILLIAMS, 2014; BRETSCHNEIDER; WÖHNER; PETERS, 2014; KWOK; WANG, 2013; XU et al., 2012; XIANG et al., 2012).

[1] <https://twitter.com/>

Outras redes sociais possuem uma política de dados mais restritiva, tornando a coleta mais trabalhosa e específica, como nos casos do **Facebook**[2] (BRETSCHNEIDER; PETERS, 2017; VIGNA et al., 2017; TING et al., 2013a), **Formspring**[3] (DINAKAR et al., 2012; BIGELOW; (KONTOSTATHIS); EDWARDS, 2016; KONTOSTATHIS et al., 2013), **Instagram**[4] (ZHONG et al., 2016; HOSSEINMARDI et al., 2015) e **ask.fm**[5] (SAMGHABADI et al., 2017; HEE et al., 2015),

Além de redes sociais, também são fontes de dados sites de vídeos como o **YouTube**[6] (DUCHARME, 2017; DADVAR; TRIESCHNIGG; JONG, 2013; CHEN et al., 2012; DINAKAR et al., 2012; XU; ZHU, 2010), notícias como o **Yahoo!**[7] (PELLE; MOREIRA, 2017; NOBATA et al., 2016; DJURIC et al., 2015; WARNER; HIRSCHBERG, 2012; SOOD; CHURCHILL; ANTIN, 2012), e enciclopédia online como a **Wikipédia**[8] (PAVLOPOULOS; MALAKASIOTIS; ANDROUTSOPOULOS, 2017; WULCZYN; THAIN; DIXON, 2017).

Após a coleta, é comum a realização do pré-processamento, como conversão de todas as letras para minúsculo (DAVIDSON et al., 2017; SALEEM et al., 2016; BURNAP; WILLIAMS, 2014), remoção de *stopwords* e caracteres não alfabéticos (MAGU; JOSHI; LUO, 2017; SALEEM et al., 2016; DJURIC et al., 2015). Entretanto, com a utilização de grandes quantidades de dados para o treinamento de *word embeddings*, esta etapa não tem sido priorizada, uma vez que quanto maior a diversidade dados de treinamento, mais genérico será o modelo gerado.

### 3.1.2 Métodos de anotação dos dados

Ting et al. (2013b) realiza uma anotação baseada na extração de *keywords* mais relevantes do texto por TF-IDF, Chen et al. (2012) faz uso da WordNet [9] e Hasanuzzaman, Dias e Way (2017) utiliza uma lista de palavras ofensivas para construir o *dataset*. Com exceção dos trabalhos citados acima, todos os outros *datasets* analisados neste trabalho são anotados por humanos.

A maior parte dos trabalhos realizam a anotação dos dados com a ajuda de juízes

[2] <https://www.facebook.com/>
[3] <https://spring.me/>
[4] <https://www.instagram.com/>
[5] <https://ask.fm/>
[6] <https://www.youtube.com/>
[7] <https://www.yahoo.com/news/>
[8] <https://www.wikipedia.org/>
[9] <https://wordnet.princeton.edu/>

humanos escolhidos para esta tarefa, estes juízes avaliam as sentenças coletadas, atribuindo a elas a classe ofensiva ou não ofensiva (BRETSCHNEIDER; PETERS, 2017; VIGNA et al., 2017; PELLE; MOREIRA, 2017; SU et al., 2017; DUCHARME, 2017; ROSS et al., 2016; GAO; KUPPERSMITH; HUANG, 2017; KENNEDY et al., 2017; WASEEM; HOVY, 2016; WULCZYN; THAIN; DIXON, 2017; ZHONG et al., 2016; DADVAR; TRIESCHNIGG; JONG, 2013; NAHAR; LI; PANG, 2013; KWOK; WANG, 2013; DADVAR; JONG, 2012; WARNER; HIRSCHBERG, 2012).

Vários trabalhos utilizam *crowdsourcing* como forma de maximizar a quantidade de instâncias nos *datasets*, através de plataformas colaborativas mais anotadores podem participar do processo de anotação dos dados. As plataformas utilizadas são: CrowdFlower [10] (CHATZAKOU et al., 2017; DAVIDSON et al., 2017; BURNAP PETE-AND WILLIAMS, 2016; HOSSEINMARDI et al., 2015), Amazon Mechanical Turk [11] (BIGELOW; (KONTOSTATHIS); EDWARDS, 2016; KONTOSTATHIS et al., 2013; REYNOLDS; KONTOSTATHIS; EDWARDS, 2011) e Brat Rapid [12] (HEE et al., 2015)

Entretanto, os maiores *datasets* produzidos, são obtidos utilizando os comentários e postagens reportados por usuários em diversas plataformas, uma vez que não requer um esforço específico para a tarefa, conseguem acumular uma quantidade de instâncias significativamente maior do que as outras formas de anotação (PAVLOPOULOS; MALAKASIOTIS; ANDROUTSOPOULOS, 2017; SALEEM et al., 2016; NOBATA et al., 2016; DJURIC et al., 2015).

## 3.2 Métodos de classificação

Nesta Seção são listados diferentes métodos de classificação utilizados pelos trabalhos analisados. Sendo os dados de cada conjunto originados em diferentes sites de diferentes domínios, é natural que estes *datasets* tenham especificidades que não permitam uma comparação direta entre modelos de classificação.

[10] <https://crowdflower.com/>
[11] <https://www.mturk.com/>
[12] <http://brat.nlplab.org>

### 3.2.1 Recursos de PLN

Alguns estudos mostram que é possível identificar palavras ofensivas que foram disfarçadas através de erros ortográficos intencionais, e esta pode ser usada para complementar métodos baseadas em dicionário. Conforme apontado (WARNER; HIRSCHBERG, 2012), geralmente é realizada a substituição de um único caractere, por exemplo "nagger". O número mínimo de edições necessárias, conhecida com a distância de Levenshtein, pode ser utilizada na resolução teste problema (NANDHINI; SHEEBA, 2015).

As ferramentas de *Parts of Speech* (POS) foram umas das primeiras abordagens utilizadas em problemas de detecção de discurso de ódio (GREEVY; SMEATON, 2004). Dinakar et al. (2012) mapeou bigramas frequentes em textos ofensivos a partir de ferramentas de POS, já (BURNAP; WILLIAMS, 2014) conseguiu detectar *substrings* frases como *"should be hung"* (deve ser enforcado). No entanto, quando utilizadas como um recurso para outros classificadores, POS não obteve bons resultados (VIGNA et al., 2017; WARNER; HIRSCHBERG, 2012),

A análise de sentimento é geralmente empregada como um recurso nos modelos de classificação para definir a polaridade do texto. Considerando que o discurso do ódio tem uma polaridade negativa, autores tem utilizado esta abordagem como classificador ou um dos recursos do classificador (DAVIDSON et al., 2017; VIGNA et al., 2017; KENNEDY et al., 2017; AGARWAL; SUREKA, 2017; GITARI et al., 2015; SCHMIDT; WIEGAND, 2017). Liu e Forss (2014) realizam uma combinação de análise de similaridade baseada em $n$-gramas com análise de sentimento na tarefa de classificação.

### 3.2.2 Aprendizado de máquina supervisionado

O problema de detectar conteúdo ofensivo tem sido comumente tratado como uma tarefa de classificação de texto, e os métodos propostos frequentemente usam algoritmos de aprendizado de máquina supervisionado. Uma variedade de classificadores é empregada em diferentes trabalhos na literatura: SVM (DAVIDSON et al., 2017; VIGNA et al., 2017; PELLE; MOREIRA, 2017; SALEEM et al., 2016; BURNAP PETEAND WILLIAMS, 2016; HEE et al., 2015) e Naïve Bayes (DAVIDSON et al., 2017; PELLE; MOREIRA, 2017; SALEEM et al., 2016), Regressão Logística (DAVIDSON et al., 2017; SALEEM et al., 2016; WULCZYN; THAIN; DIXON, 2017), Árvores de Decisão e *Random Forests* (DAVIDSON et al., 2017), além de empilhamento destes classificadores (PETE;

L., 2015).

Os atributos utilizadas nos classificadores desempenham um papel importante no resultado final dos modelos de classificação. A maioria dos recursos é baseada em análise de texto, onde $n$-gramas de palavras e caracteres têm sido amplamente empregados como atributos extraídos dos textos (com $n$ geralmente variando de 1 a 5) (DAVIDSON et al., 2017; VIGNA et al., 2017; KENNEDY et al., 2017; SALEEM et al., 2016). Outros elementos de texto também são utilizados como atributos, que frequentemente são considerados indicadores binários ou de contagem: *hashtags* (DAVIDSON et al., 2017), *emoticons* (VIGNA et al., 2017; BURNAP; WILLIAMS, 2014), *Named Entity Recognition* (NER) (CORTIS; HANDSCHUH, 2015), URLs (DAVIDSON et al., 2017; NOBATA et al., 2016), Extração de tópicos (LIU; FORSS, 2014), letras maiúsculas e minúsculas (CHATZAKOU et al., 2017; CHEN et al., 2012; DADVAR; JONG, 2012), comprimento das palavras e frases (NOBATA et al., 2016; DADVAR; TRIESCHNIGG; JONG, 2013) e menção de outros usuários (DAVIDSON et al., 2017),

Quando o autor dos comentários pode ser identificado de alguma forma, recursos sobre eles também podem ser usados. Exemplos de tais características são a idade do usuário (HASANUZZAMAN; DIAS; WAY, 2017; DADVAR; TRIESCHNIGG; JONG, 2013), gênero (HASANUZZAMAN; DIAS; WAY, 2017; WASEEM; HOVY, 2016), localização (HASANUZZAMAN; DIAS; WAY, 2017; WASEEM; HOVY, 2016), popularidade (quantos seguidores o usuário possui), número de postagens, quanto tempo a pessoa é um usuário dessa plataforma, se a conta é verificada, o número de assinaturas e o intervalo entre as postagens/comentários (CHATZAKOU et al., 2017).

### 3.2.3 *Embeddings* e *Deep Learning*

Após a publicação do trabalho de Mikolov et al. (2013) e o surgimento de várias ferramentas que trabalham com *embeddings* de palavras e sentenças, vários modelos de classificação de texto ofensivo foram propostos baseados na utilização destas abordagens.

Djuric et al. (2015) realizam a identificação de discurso de ódio nos comentários de notícias, usando o modelo de word2vec com CBOW para treinar representações de palavras e comentários em "*embeddings* de parágrafo" (chamado paragraph2vec). Após a produção das representações vetoriais é utilizado regressão logística para a classificação obtendo 0,82 de medida F. Nobata et al. (2016) utilizam os mesmo dados porém utilizando uma média dos vetores das palavras da frase, para obter uma única representação da frase

como um todo.

Yuan, Wu e Xiang (2016) também utilizam um classificador de regressão logística para identificar ocorrências de discriminação. Um modelo *Long Short Term Memory* (LSTM) pré-treinado é utilizado como entrada deste classificador, este modelo contém *embeddings* de palavras para a representações semânticas dos tweets.

Badjatiya et al. (2017) utilizam *Convolutional Neural Networks* (CNN) e LSTM, combinando vários recursos como os *embeddings*, TF-IDF e BOW. O foco principal foi detecção de racismo e o sexismo, sendo que o classificador baseado em LSTM, obteve um desempenho melhor do que os métodos de *baseline*.

Bashar et al. (2018) propõem um método para detectar tweets misóginos. Utilizando uma CNN e tendo como entrada vetores de palavra pré-treinados em no domínio específico da tarefa, foi alcançado resultado de 0,93 na medida F, que é um resultado bastante relevante, uma vez que o *dataset* possui apenas 5000 instâncias.

### 3.2.4 Avaliação dos resultados dos trabalhos relacionado

Na Tabela 3.1, os resultados dos trabalhos relacionados são apresentados em ordem decrescente do ano de publicação. Foi selecionado o melhor resultado dentre os testes realizados por cada estudo. A primeira coluna possui número do artigo, especificado na legenda abaixo da tabela; a segunda coluna apresenta o melhor resultado e sua métrica; a terceira coluna reporta os algoritmos utilizados; e a última coluna os recursos utilizados. Não é possível concluir quais abordagens têm melhor desempenho, uma vez que os testes são realizados em diferentes *datasets*. Entretanto, nota-se que várias tecnologias utilizadas pelos autores são técnicas recentes, como é o caso das *CNN*.

Tabela 3.1: Resultados alcançados pelos trabalhos relacionados, utilizando as métricas: medida F (F), ROC-AUC (A), Revocação (R), Acurácia (Ac)

| Artigo* | Resultado | Algoritmos | Recursos |
|---|---|---|---|
| [1] | 0,89 (A) | CNN | texto e metadados baseados no usuário, rede e texto |
| [2] | 0,93 (F) | CNN | pré-treinamento com tweets abusivos e dados rotulados |
| [3] | 0,92 (F) | Regressão Logística | word2vec, $n$-gramas de caracteres, unigramas de palavras e TF-IDF |
| [4] | 0,90 (F) | Regressão Logística e SVM | TF-IDF, POS, análise de sentimento, hashtags, menções, retweets, URLs, número de caracteres, palavras e sílabas |
| [5] | 0,85 (F) | SVM e LSTM | POS, análise de sentimento, word2vec, CBOW, $n$-grams, recursos de texto |
| [6] | 0,86 (R) | One-class Classifiers, Random Forest, Naive Bayes, Árvores de Decisão | Modelagem de tópicos, análise de sentimentos, análise de tom, análise semântica, metadados contextuais |
| [7] | 0,83 (F) | Skip-bigram Model | $n$-gramas, comprimento, pontuação, POS |
| [8] | 0,77 (F) | Random Forest, Decision Tree, SVM | BOW, dicionário, dependências digitadas |
| [9] | 0,73 (F) | Regressão Logística | Atributos do usuário |
| [10] | 0,63 (A) | SVM | Dicionários |
| [11] | 0,91 (Ac) | Deep Learning | word2vec |
| [12] | 0,80 (A) | Regressão Logística | paragraph2vec |

* Artigos: [1] (FOUNTA et al., 2018), [2] (BASHAR et al., 2018), [3] (KSHIRSAGAR et al., 2018), [4] (DAVIDSON et al., 2017), [5] (VIGNA et al., 2017), [6] (AGARWAL; SUREKA, 2017), [7] (NOBATA et al., 2016), [8] (BURNAP PETEAND WILLIAMS, 2016), [9] (WASEEM; HOVY, 2016), [10] (TULKENS et al., 2016), [11] (YUAN; WU; XIANG, 2016), [12] (DJURIC et al., 2015),

## 3.3 Sumário do Capítulo

Neste Capítulo foi construída uma visão geral sobre a detecção automática de discurso de ódio. Realizar uma comparação direta de diferentes características e métodos, se torna muito difícil sem um *dataset* de *benchmark*. Entretanto, foram expostas as principais abordagens utilizadas, tanto na geração quanto na classificação de dados.

Uma das principais dificuldades relatadas pelos trabalhos analisados é que o discurso de ódio depende fortemente do contexto cultural em que está inserido, sendo que uma mesma sentença pode ser considerada ofensiva em uma comunidade e não ofensiva em outra. Esta subjetividade também dificulta a construção de *datasets*, que é agravada pela necessidade de se classificar muitas instâncias para conseguir uma amostra relevante.

O aprendizado supervisionado mostrou ser o método mais utilizado por trabalhos que atacam esta tarefa, sendo que as abordagens em nível de caractere entregaram melhor desempenho em relação as abordagens em nível de palavra. Entretanto, os melhores resultados são apresentados em trabalhos que utilizam *word embeddings*.

Neste trabalho, os esforços foram concentrados extrair automaticamente recursos do texto, não utilizando ferramentas e recursos especializados, como *parsers* e *POS-taggers*. A ideia é que `Hate2Vec` deve ser capaz de funcionar mesmo para idiomas que não possuem esse tipo de recurso, sendo o único recurso necessário é uma lista de palavras ofensivas e uma grande quantidade de texto contextualizado que pode ser coletado automaticamente da Web.

# 4 *DATASET* COM COMENTÁRIOS OFENSIVOS EM PORTUGUÊS

Este Capítulo é dedicado ao detalhamento da construção do OFFCOMBR-2 e OFFCOMBR-3, que ao nosso conhecimento são os primeiros conjuntos de dados com a temática de comentários ofensivos em Português do Brasil (PT-BR). Primeiramente é apresentada a origem dos dados e o método utilizado em sua obtenção. Além disso, são fornecidos os resultados de classificação obtidos por algoritmos de classificação padrão nesses conjuntos de dados, que podem ser utilizados como *baseline* para futuros trabalhos sobre esse tópico.

## 4.1 Coleta de dados

Como fonte de dados foi escolhido o site <g1.globo.com> por ser a página de notícias mais acessada do Brasil [1] e como resultado disso, o que recebe mais comentários. Embora os comentários neste site passem por moderação, foi encontrado um número considerável de textos ofensivos. Após uma análise preliminar sobre todas as notícias postadas em um dia neste site, foi constatado que cerca de 90% delas possuíam pelo menos um comentário ofensivo. Além disso, foi verificado que as categorias de notícias com a maior quantidade de comentários ofensivos são **política** e **esportes**. Com o objetivo de obter o maior número de comentários ofensivos, a coleta de dados foi limitada a essas seções.

Para obter estes comentários, foi implementado um *Webscraper* utilizando as bibliotecas Python requests[2] e LXML[3]. Este *script* envia requisições para o site nas seções escolhidas, sendo então descobertos os links para as páginas com notícias, que por sua vez são baixadas em formato HTML e analisadas para extrair os atributos das notícias como texto, título e data além da URL para a API onde todos os comentários para uma determinada notícia podem ser obtidos no formato JSON.

Os comentários extraídos destes arquivos passam por uma etapa de pré-processamento, onde os que são compostos apenas por emojis ou outros caracteres não alfabéticos são descartados. Acentuação, caracteres especiais e formatação do texto foram preservadas a fim de serem reaproveitadas como possíveis *features*. Todo o código

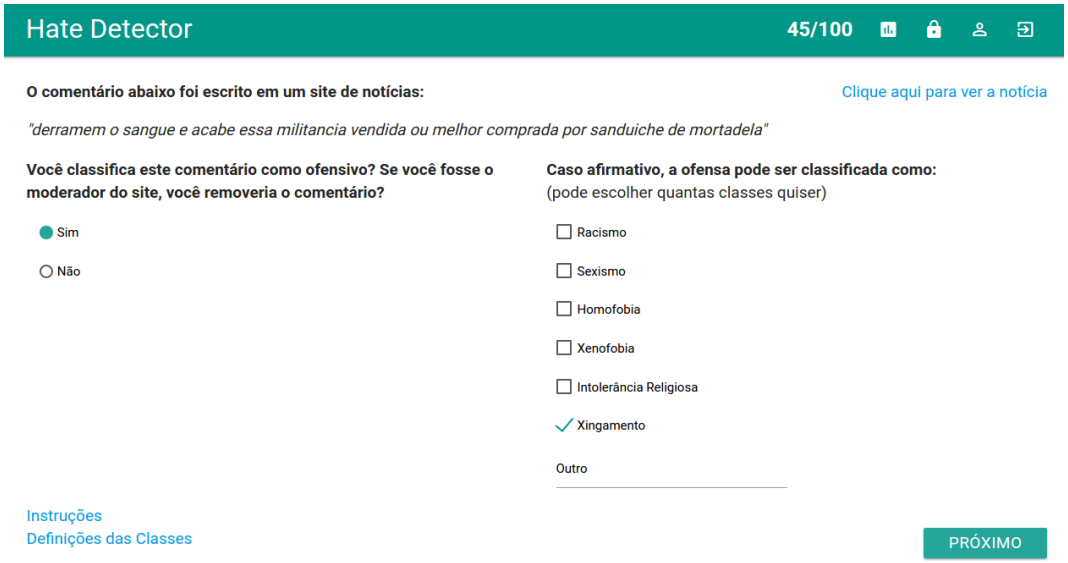
[1] <http://www.alexa.com/topsites/countries/BR>
[2] <http://docs.python-requests.org>
[3] <http://lxml.de>

fonte da ferramenta de captura de dados está disponível em um repositório no github [4]. Para preservar o anonimato dos autores, foram removidos todos os sobrenomes, quando os usuários eram citados. Ao final deste processo, foram obtidos 10.336 comentários postados para 115 notícias.

## 4.2 Ferramenta de anotação

Um tópico crítico na criação de *datasets* é o processo de anotação, pois ele é o que demanda mais tempo e depende da disponibilidade de juízes humanos. Sendo inviável fazer a rotulação de todos os 10.336 comentários, uma amostra de 1.250 comentários foi aleatoriamente selecionada. Para estruturar o processo de anotação, foi desenvolvida uma ferramenta chamada de Hate Detector, que também possui código fonte disponível[5].

Figura 4.1: Hate Detector: ferramenta para anotação de dados




Fonte: Elaborado pelo autor

A ferramenta Hate Detector é um sistema Web, no qual os anotadores são previamente cadastrados e as instâncias a serem anotadas são importadas para o banco de dados. A Figura 4.1 apresenta a interface de anotação, onde cada anotador visualiza as instâncias que ele deve rotular que são escolhidas de forma aleatória no banco de dados. Ao finalizar, o anotador pode aceitar receber um novo lote. Para este trabalho decidiu-se por utilizar três classificações por instância, somando um total de 3750 classificações divididas por sete anotadores.

[4] <https://github.com/rogersdepelle/hatedetector/blob/master/comments/scraper.py>
[5] <https://github.com/rogersdepelle/hatedetector/>

Cada comentário foi classificado por três juízes que foram apresentados à seguinte pergunta: **Você classifica este comentário como ofensivo? Se você fosse o moderador do site, você removeria o comentário?**. No caso de uma resposta afirmativa, o anotador também foi solicitado a categorizar a ofensa: **Caso afirmativo, a ofensa pode ser classificada como: racismo, sexismo, homofobia, xenofobia, intolerância religiosa ou xingamento**.

O Hate Detector possui a funcionalidade de exportar os dados classificados nos formatos CSV e ARFF, selecionando também o nível de concordância dos anotadores. No nível 2, a instância pertence à classe atribuída por pelo menos dois. No nível 3, constam apenas as instâncias cuja classe foi escolhida de forma unânime. Esta segunda opção pode ocasionar a diminuição na quantidade de instâncias, pois são desconsideradas as que tiveram divergência entre seus anotadores.

## 4.3 OFFCOMBR-2 e OFFCOMBR-3

A partir deste processo de anotação foram gerados dois conjuntos de dados, que também se encontram disponíveis <https://github.com/rogersdepelle/OffComBR>, o primeiro chamado de OFFCOMBR-2, possui 1250 instâncias sendo que a classe atribuída a cada comentário foi a escolhida por *pelo menos dois* anotadores. O segundo é um conjunto de dados mais restrito, chamado OFFCOMBR-3, este conjunto é composto apenas dos comentários para os quais *três juízes concordaram* com o fato de o comentário ser ou não ofensivo.

Para medir o nível de concordância entre os juízes, foi calculada a medida de Fleiss Kappa (FLEISS; COHEN, 1973). Para o OFFCOMBR-2, o valor foi de 0,71, este valor está dentro do intervalo de concordância encontrado em outros trabalhos que também realizaram anotações (0,63 foi reportado por Warner e Hirschberg (2012), 0,73 por Chen et al. (2012) e 0,84 por Nobata et al. (2016)). Como o OFFCOMBR-3 só contém instâncias para as quais a mesma classe foi atribuída pelos três juízes, não se fez necessário calcular medida de Fleiss Kappa.

No OFFCOMBR-2, 419 (de 1.250) comentários foram considerados ofensivos por pelo menos dois juízes, representando 32,5 % do total. Um aspecto observado é que nenhum comentário foi considerado ofensivo por apenas um juiz. Para OFFCOMBR-3, existem 202 comentários ofensivos (de 1.033), totalizando 19,5 % dos casos. Em ambos os *datasets*, existe um desequilíbrio entre o número de instâncias nas classe, sendo o

número instâncias negativas maior do número de positivas.

Tabela 4.1: Representação de cada categoria no dados anotados

| # Anotadores | Xenofobia | Homofobia | Sexismo | Racismo | Xingamento | Intolerância Religiosa |
|---|---|---|---|---|---|---|
| 1 | 13 (1,0%) | 35 (2.8%) | 14 (1,1%) | 19 (1.5%) | 375 (30.0%) | 1 (0.1%) |
| 2 | 12 (1.0%) | 14 (1,1%) | 8 (0.6%) | 18 (1.4%) | 286 (22.9%) | 1 (0.1%) |
| 3 | 5 (0.5%) | 9 (0.9%) | 4 (0.4%) | 1 (0.1%) | 175 (16,9%) | 0 (0.0%) |

Em relação às categorias (racismo, sexismo, homofobia, xenofobia, intolerância religiosa ou xingamentos), as estatísticas de cada conjunto de dados são mostradas na Tabela 4.1. A categoria mais comum é a de xingamento (375 de 419), outras categorias tiveram poucos comentários. A intolerância religiosa foi encontrada em apenas um comentário e não foi unânime. Desta forma, não serão realizados experimentos visando a detecção de conteúdo ofensivo nas categorias específica, pois não existem instâncias suficientes.

## 4.4 Resultado de algoritmos de classificação convencionais

Com o objetivo de fornecer um *baseline* para OFFCOMBR-2 e OFFCOMBR-3, eles foram submetidos aos classificadores SVM e Naive Bayes (*classifier*). Alguns recursos foram aplicados a fim de testar seu ganho, são eles: utilizar $n$-gramas de palavras como atributos, onde $n$ variou de 1 a 3 palavras (*tockenizer*). Utilizar formatação original ou converter todos os caracteres para minúsculo (*lower*). A seleção de atributos (*FS*) a serem usados pelo classificador, mantendo apenas os que apresentavam *Information Gain* (InfoGainAttributeEval [6]) maior do que zero. A métrica escolhida é a média ponderada da medida F. Para testar se os diferentes desempenhos eram estatisticamente significativos, foi utilizado o teste T pareado com a medida F das execuções com o limiar padrão de significância estatística de $\alpha$ = 0,05.

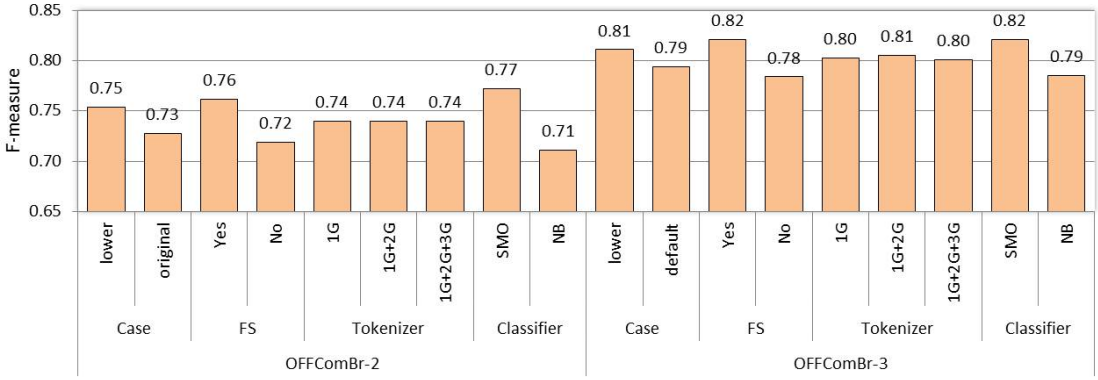
As colunas da Figura 4.2 refletem a média da medida F em todas as execuções para um determinado recurso apresentados acima. Por exemplo, a primeira coluna mostra a média da medida F (CHEN; KUO; MERKEL, 2004) ponderada em todas as execuções nas quais o texto dos comentários foi convertido para minúsculas, considerando ambos os algoritmos de classificação.

A configuração com o melhor resultado, em OFFCOMBR-2 e OFFCOMBR-3, foi obtida pelo SVM sobre os arquivos nos quais os comentários foram convertidos em letras

[6] <http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html>

minúsculas e foi aplicada a seleção de atributos. Os valores absolutos para a medida F variaram de 0,69 a 0,85. Outros trabalhos que também utilizaram SVM com recursos de $n$-gramas apresentaram resultados um pouco abaixo do patamar alcançado, destacando-se: Pete e L. (2015) entre 0,28 e 0,77, Vigna et al. (2017) entre 0,25 e 0,75, Badjatiya et al. (2017) entre 0,78 e 0,81, todos em medida F.

Figura 4.2: Resultados *baseline* para OFFCOMBR-2 e OFFCOMBR-3




Fonte: Elaborado pelo autor

## 4.5 Sumário do Capítulo

Neste Capítulo foi apresentado o método utilizado na construção dos *datasets* OFFCOMBR-2 e OFFCOMBR-3, passando pelas etapas de coleta de dados, anotação e análise. O processo de obter os comentários decorreu sem maiores dificuldades, assim como o pré-processamento, na fase de anotação não foi identificada uma ferramenta que atendesse exatamente o processo almejado, sendo desenvolvida uma específica para este fim. Assim como em outros *datasets* do estado da arte, foi constatado um desbalanceamento entre as classes, o que pode representar uma dificuldade para os algoritmos de classificação. Apesar disso, nas melhores configurações, foi alcançado 0,85 de medida F para utilização como *baseline*.

# 5 HATE2VEC: MÉTODO IDENTIFICAÇÃO DE COMENTÁRIOS OFENSIVOS

Este Capítulo detalha a principal contribuição deste trabalho: a proposta de um método de classificação de comentários ofensivos, baseada na combinação de classificadores. Inicialmente este Capítulo fornece uma visão geral do método e, a seguir, cada fase é abordada com mais profundidade, detalhando cada etapa do processo cada uma das fases.

Embora existam outros trabalhos já utilizaram representação vetorial de palavras e documentos para detectar textos ofensivos, a forma como foram combinadas essas técnicas (utilizando um conjunto de classificadores) é original para este tipo de tarefa. Além disso, o método não depende de recursos como etiquetadores morfossintáticos (*part-of-speech tagers*), dicionários de sinônimos e outros recursos de PLN, que podem não estar disponíveis em alguns idiomas. Os únicos recursos externos utilizados são uma listas de "sementes"com algumas palavras ofensivas e um *dataset* anotado.

## 5.1 Definição do Problema e Visão Geral do Método

O problema que abordamos pode ser resumido como: dado um comentário (representado sob a forma de um texto curto), devemos atribuí-lo a uma classe que pode ser *yes* (ou 1) para indicar que o comentário é ofensivo ou *no* (ou 0) para não ofensivo. Esta descrição faz com que o nosso problema possa ser modelado como uma classificação binária de textos.

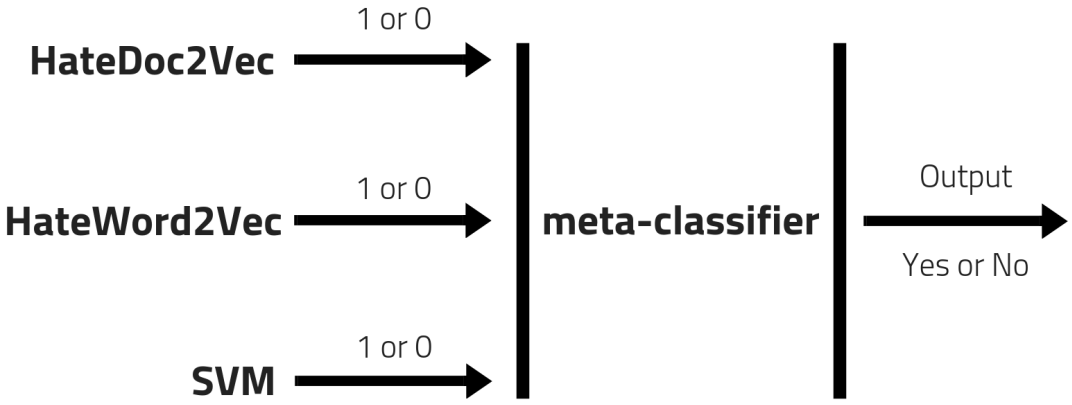
Ao estudar o estado da arte na área de detecção de textos ofensivos, muitos trabalhos têm utilizado representação vetorial de palavras de alguma forma, seja em nível de palavra ou em nível de sentença (DJURIC et al., 2015; NOBATA et al., 2016; ALMEIDA et al., 2017). A abordagem proposta, chamada de `Hate2Vec`, utiliza representações vetoriais nos níveis de palavra e de sentença.

O `Hate2Vec` realiza a classificação em duas fases. Na primeira fase, o objetivo é ter as previsões dos classificadores base:
($i$) HateWord2Vec, um classificador baseado em léxico que utiliza uma lista de palavras ofensivas, que é automaticamente expandida usando a similaridade entre os vetores que representam estas palavras;
($ii$) HateDoc2Vec, um classificador de regressão logística dos vetores que representam as sentenças; e

($iii$) SVM, um classificador padrão baseado em uni-gramas

Na segunda fase, um meta-classificador é construído a partir das previsões fornecidas pela etapa anterior.

Figura 5.1: Visão geral do método: As previsões dos três classificadores de base são utilizadas para alimentar um meta-classificador que faz a previsão final




Fonte: Elaborado pelo autor

A Figura 5.1 apresenta o visão geral do método com seus componentes. O meta-classificador realiza suas predições com base nos resultados dos três classificadores-base.

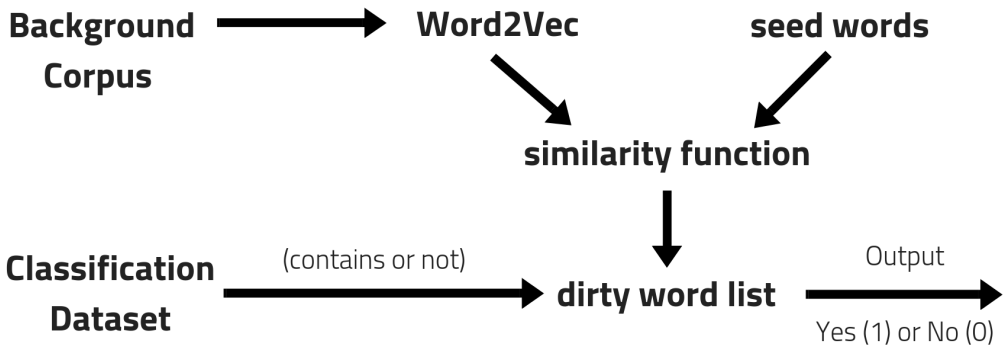
## 5.2 HateWord2Vec

Nesta etapa, foi construído um classificador baseado em um léxico e em uma lista de palavras ofensivas, expandida pelo uso de *word embeddings*. A Figura 5.2 demonstra as etapas da classificação, que utiliza dois recursos: um *background corpus* e uma lista de palavras-semente (*seed words*). Foram produzidas versões em inglês e português para estes dois recursos, visto que esses serão os idiomas dos datasets a serem usados na avaliação experimental. O conjunto e dados a ser classificado é denominado *Classification Dataset* na Figura 5.2.

Foi observado que existe uma forte correlação entre os textos ofensivos e a presença de palavrões. Embora palavrões possam ser usados até mesmo em elogios, nossos esforços foram concentrados em identificar a sua presença como uma indicação de ofensividade. Em outras palavras, embora às vezes não relacionados, ofensas e palavrões tendem a ter uma alta correlação. A ideia é que, mesmo que o conteúdo não seja uma ofensa, os pais podem não gostar que as crianças acessem textos com palavrões, por exemplo.

Autores de comentários e postagens em páginas da Web, frequentemente disfar-

Figura 5.2: Visão geral do classificador HateWord2Vec.




Fonte: Elaborado pelo autor

çam palavras ofensivas para evitar que seus comentários sejam removidos pelos moderadores (*e.g., f*ck*). Para encontrar essas variações, foi utilizada a representação vetorial de palavras (*i.e.,* word embeddings). Uma vez que ela se concentra no significado, e não na forma com que é escrita. Desta forma, foi possível encontrar maneiras variadas de escrever a mesma palavra como as *"sht"* e *"mothafucka"*, que não estavam originalmente na lista de palavras-semente.

Em geral, os conjuntos de dados utilizados para treinar um modelo de *word embeddings* possuem ao menos 1 GB de tamanho e vocabulários entre 30 e 50 mil palavras. Eles são baseados em extrações de dados da Wikipédia[1] (PENNINGTON; SOCHER; MANNING, 2014) e de páginas de notícias indexadas pelo por buscadores [2]. A forma de escrita e o vocabulário utilizado nestes textos é muito diferente daquela utilizada em comentários postados na Internet. Estes contêm muitas gírias, erros de digitação e jargões específicos da Web. Sendo assim, a utilização de modelos pré-treinados em dados da Wikipedia não seria adequada, uma vez que a intersecção entre o conjunto palavras ofensivas e o vocabulário desses corpora é muito pequena.

Para realizar o treinamento dos modelos compatíveis com o domínio, foi realizada uma coleta de comentários de páginas da Web e *tweets*. Com o intuito de garantir que o vocabulário ofensivo estivesse presente nestes dados, a lista de palavras-semente foi utilizada na busca de *tweets* utilizando a API do Twitter[3]. Estes *tweets* foram coletados juntamente com suas respostas, baseado-se no princípio que ofensas são geralmente respondidas com mais ofensas em ambientes virtuais.

[1] <https://dumps.wikimedia.org/>
[2] <https://github.com/Kyubyong/wordvectors>
[3] <https://developer.twitter.com/>

Comentários de sites de notícias também foram coletados, somando mais de 1 milhão de *tweets* e comentários com e sem textos ofensivos. Este conjunto de dados foi chamado de *background corpus*. Observe que esse corpus é montado independentemente do conjunto de dados a ser classificado. Além disso, as instâncias a serem classificadas não fazem parte deste conjunto. Como essas sentenças geralmente são curtas, os modelos gerados não são muito grandes, mas contemplam o vocabulário desejado.

O único pré-processamento realizado foi converter todo o texto do *background corpus* em minúsculo e remover caracteres de pontuação, tendo a intenção de preservar ao máximo a estrutura original do texto. O treinamento foi realizado com diferentes hiperparâmetros, a fim de definir o melhor modelo para a realização da tarefa, que é encontrar palavras semelhantes as palavras-semente.

A lista de palavras-semente [4] foi manualmente construída contendo 50 palavras para português e 56 palavras para inglês. A lista em português foi baseada em uma lista de xingamentos [5] e o termos presentes no hatebase [6], já para o inglês foi utilizada uma lista criada pela Carnegie Mellon University [7].

A partir desta lista foi realizada uma expansão, dando origem a *dirty word list*. O processo expansão passa por extrair o vocabulário do conjunto de dados de classificação (ou seja, as palavras distintas) e, para cada palavra desse vocabulário, o nível de similaridade com todas as palavras na lista de palavras-semente foi medido pelo modelo pré-treinado com o *background corpus*.

Palavras com valores de similaridade maiores do que um limiar $t$ com pelo menos $w$ palavras da lista de palavras-semente, foram consideradas ofensivas e adicionadas à *dirty word list*. Os valores de $t$ e $w$ são determinados experimentalmente e podem ser encontrados na Seção 6.3.

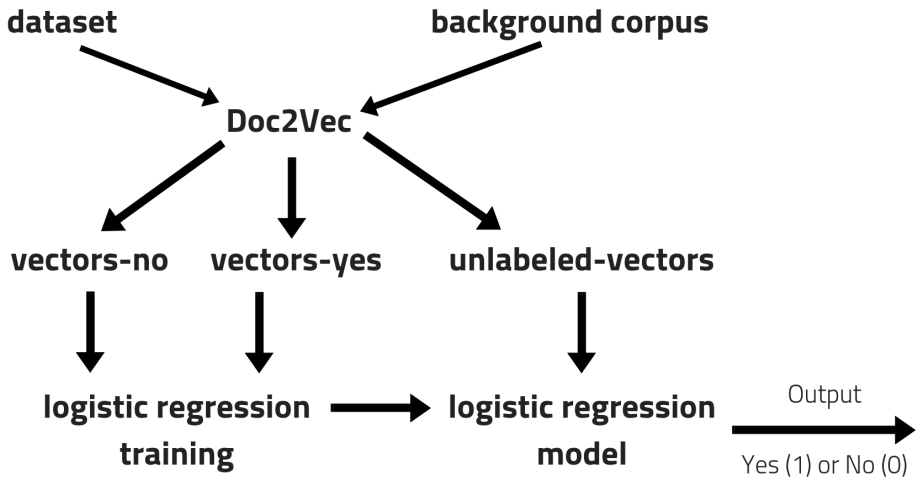
A classificação é baseada na presença de palavras ofensivas, portanto, todas as frases que tiverem pelo menos uma palavra encontrada na *dirty word list* são consideradas sentenças ofensivas. Embora este método se assemelhe à utilização de um simples dicionário, a hipótese é que ao utilizar uma lista expandida, que foi treinada para se adaptar ao contexto e vocabulário das sentenças, a chances de obter sucesso aumentam consideravelmente.

[4] <https://gist.github.com/rogersdepelle/d06c25844bcbe5d8c53299eaa795d1a2>
[5] <https://aprenderpalavras.com/lista-de-palavroes-xingamentos-e-girias>
[6] <https://www.hatebase.org/>
[7] <https://www.cs.cmu.edu/~biglou/resources/bad-words.txt>

### 5.3 HateDoc2Vec

O *HateDoc2Vec* é um dos classificadores-base que compõem o meta-classificador. Sua função é realizar a classificação das entradas a nível de sentença, utilizando o Doc2Vec (LE; MIKOLOV, 2014). Este processo ocorre em três etapas, são elas: treinamento CBOW da sentenças, treinamento do classificador de regressão logística e classificação das sentenças. Uma visão geral do processo é apresentada na Figura 5.3.

Figura 5.3: Visão geral do classificador HateDoc2Vec.




Fonte: Elaborado pelo autor

O classificador *HateWord2Vec* se baseia exclusivamente na presença e ou ausência de palavras ofensivas, mas existem comentários são igualmente ofensivos sem utilizar xingamentos. Outra possibilidade, é a utilização de palavras que dependem do contexto para se tornarem ofensivas. O *HateDoc2Vec* se propõe a realizar uma análise do comentário como um todo, e se baseia na hipótese de que comentários ofensivos são semanticamente semelhantes, possuindo ou não palavras ofensivas.

A primeira etapa consiste em um treinamento CBOW em nível de palavra e sentença que é executado simultaneamente. Semelhante ao Word2Vec, mas em vez de usar apenas palavras para prever a próxima palavra da sentença, ele também usa um vetor chamado *Paragraph ID*, que é exclusivo do documento. Ao final do treinamento dos vetores de palavras, existirá também um vetor de documento (*Paragraph ID*), que contém uma representação vetorial de cada sentença. Estes serão chamados de *doc vectors*.

Os *doc vectors* representam todos os comentários do *dataset* como vetores de baixa dimensionalidade (cerca de 200 dimensões). É importante destacar que estes veto-

res não são frutos de operações matemáticas sobre os vetores das palavras que compõem as frases, mas sim representações semânticas, uma vez que sentenças que não possuem palavras em comum, mas sentidos semelhantes provavelmente estão em locais próximos neste espaço vetorial. Palavras e frases semanticamente semelhantes também estão localizadas próximas.

Esta fase de treinamento foi enriquecida utilizando o modelo treinado na fase anterior, o *background corpus*. Esta prática melhora a qualidade do resultado, uma vez que os *datasets* são consideravelmente pequenos para realizar treinamento de *word embeddings*. Vale notar que este o treinamento deste modelo não foi influenciado pelo *dataset* uma vez que ele foi foi utilizado na construção dos vetores do modelo.

Na segunda etapa, é realizada classificação utilizando um algoritmo de regressão logística. Os *doc vectors* são separados em três conjuntos, os anotados como ofensivos (*vectors-yes*), não ofensivos (*vectors-no*), e não classificados (*unlabeled-vectors*). Os dois primeiros conjuntos são utilizados no treinamento do forma separada, a partir dos quais o algoritmo gera uma função logística que será utilizada na classificação. Esta função separa o espaço vetorial em duas regiões, baseando-se nas posições dos *doc vectors*.

Por fim, o terceiro conjunto (chamado de *unlabeled-vectors* na Figura 5.3), que possui os vetores dos comentários a serem classificados, é submetido a este classificador. A decisão de qual classe será atribuída à instância depende de qual das regiões o vetor se localiza. A saída deste classificador é binária, descrevendo se o comentário é ou não ofensivo.

## 5.4 Classificador Bag-of-Words

Com objetivo de formar uma combinação de classificadores, fez se necessário a utilização de um terceiro, estas combinações apresentam melhor desempenho quando utilizam uma quantidade ímpar de entradas. São então utilizadas as palavras dos comentários como atributos formando um classificador BOW convencional (conforme descrito na Seção 2.1). O classificador escolhido foi o SVM, pois tende a ter bom desempenho na classificação de textos, em especial, nas tarefas de análise de sentimento (MEDHAT; HASSAN; KORASHY, 2014). Além disso, obteve o melhor desempenho nos testes preliminares realizados sobre o os *datasets* OFFCOMBR-2 e OFFCOMBR-3 reportados no Capítulo 4.

Os vetores submetidos ao SVM para a classificação das instâncias são forma-

dos por todos os uni-gramas presentes nas mesmas utilizando pesos TF-IDF, que mede frequência do termo no documento em relação frequência do termo na coleção e tem o objetivo de indicar a importância de um termo de um documento em relação a uma coleção de documentos. Este processo de classificação gera uma saída binária para ser utilizada no meta-classificador, onde 1 representa um documento com conteúdo ofensivo e 0 um documento que não apresenta este tipo de conteúdo.

## 5.5 Meta-Classificador

A última fase da classificação é composta por um meta-classificador que se utiliza das saídas dos outros três classificadores base. A Figura 5.1 representa o formato da entrada e da saída do classificador. Cada instância da entrada agora é formada por um vetor de três posições que podem assumir valores de 0 ou 1. Cada posição é oriunda de um classificador base, sendo que se este classificou a instância como ofensiva, o valor é 1, caso contrário é 0.

- **HateWord2Vec**: Saída binária do classificador HateWord2Vec.
- **HateDoc2Vec**: Saída binária do classificador HateDoc2Vec.
- **SVM**: Saída binária do classificador SVM.

Vários classificadores foram testados na tarefa de meta-classificação tendo apresentado resultados semelhantes. O escolhido foi o Naïve Bayes, por apresentar resultado mais estável entre os *datasets*. Os parâmetros utilizados nas execuções, bem como os resultados alcançados são apresentados no Capítulo 6

## 5.6 Sumário do Capítulo

Neste Capítulo foi apresentado o método proposto para a classificação de comentários ofensivos na Web. O objetivo foi combinar diferentes abordagens que obtiverem bons resultados em diferentes tipos de instâncias, de modo que estas pudessem contribuir para um classificador mais genérico. O método realiza classificação das instâncias a nível de palavra e sentença, com a utilização de poucos recursos externos. O intuito é permitir sua aplicação em diferentes idiomas, mesmo os que não dispõem de ferramentas de PLN.

# 6 EXPERIMENTOS E RESULTADOS

Neste Capítulo são apresentados os experimentos realizados com o método descrito no Capítulo 5, incluindo os conjuntos de dados, parâmetros utilizados e métricas de avaliação escolhidas. A última Seção do Capítulo traz os resultados dos experimentos, juntamente com uma análise estatística sobre os mesmos, comparando e discutindo as causas das diferenças de desempenho entre os *datasets*, utilizando mesmo método.

## 6.1 Conjuntos de dados

A abordagem utilizada na escolha dos *datasets* seguiu a proposta de experimentar o método em diferentes domínios. Para tanto foram escolhidos conjuntos de dados em português e inglês, compostos por postagens em redes sociais e comentários extraídos de sites, com variação também no número de instâncias. Abaixo são descritos os três conjuntos de dados escolhidos, entretanto, um deles possui duas versões, sendo os testes aplicados em 4 *datasets*.

- `Tweets-EN` [1] (WASEEM; HOVY, 2016) é um conjunto de dados com aproximadamente 16 mil *tweets* em inglês anotados pela presença de discurso de ódio, sendo que a cada *tweet* foi atribuída, pelos anotadores, uma classe das três classes possíveis, são elas: racismo, sexismo ou nenhuma. Para que este *dataset* possuir a mesma configuração de classe dos outros, as classes racismo e sexismo foram fundidas, sendo que qualquer instâncias anotada como racismo ou sexismo passaram a pertencer a classe *yes*, e o restante a classe *no*.
- `Kaggle` [2] este *dataset* é composto por 6 mil *tweets* em inglês. Todas as instâncias foram anotadas como ofensivas e não ofensivas, que foram redesignadas como as classes *yes* e *no*, respectivamente. A instâncias se encontram divididas em dois conjuntos, um de treinamento `Kaggle-train` e `Kaggle-test` Originalmente este conjunto de dados foi disponibilizado para uma competição de classificadores no ano 2012, sendo que o vencedor da competição alcançou um resultado de 0,84248 [3] de área sob a curva (AUC).

[1] <https://github.com/zeerakw/hatespeech>
[2] <https://www.kaggle.com/c/detecting-insults-in-social-commentary/data>
[3] <https://www.kaggle.com/c/detecting-insults-in-social-commentary/leaderboard>

- `OffComBR` [4] (PELLE; MOREIRA, 2017) descrito no Capítulo 4.

A tabela 4.1 detalha a disposição das instâncias em cada conjunto de dados. Todos os *datasets* são desbalanceados, com a classe *no* contendo um número significativamente maios de instâncias que a classe *yes*.

Tabela 6.1: Datasets

| nome | instâncias | ofensivas (*yes*) | não ofensivas (*no*) | idioma |
|---|---|---|---|---|
| OffComBr2 | 1.250 | 419 | 831 | PT-BR |
| OffComBr3 | 1.033 | 201 | 832 | PT-BR |
| Tweets-EN | 15.930 | 5.113 | 10.817 | EN |
| Kaggle-train | 3.947 | 1.049 | 2.898 | EN |
| Kaggle-test | 2.647 | 693 | 1.954 | EN |

## 6.2 Ferramentas e parâmetros escolhidos

A linguagem de programação Python3 [5] foi utiliza para a implementação dos scripts de preparação dos dados e nos classificadores `HateWord2Vec`, `HateDoc2Vec`. Várias bibliotecas também foram utilizadas, sendo a Gensim [6] a principal delas, pois fornece métodos para modelagem de espaço vetorial para gerar as representações de texto, com implementações dos modelos Word2Vec, Doc2Vec entre outros. É importante destacar que todas as ferramentas utilizadas são *open-source*.

Para construção do `background corpus` em português, foram coletados *tweets* e comentários de sites de notícias, totalizando 1 milhão de instâncias. Estas foram pré-processadas e armazenadas em um banco de dados PostgreSQL [7]. O `background corpus` em inglês foi treinado com base em 1 milhão de *tweets* somados a um conjunto de dados anotados com 100 mil comentários de páginas do Wikipédia [8], que contém texto ofensivo [9].

Os parâmetro utilizados na expansão da *dirty word list* são $t$= 0,75 e $w$ = 3. Diferentes tamanhos de dimensão e foram testadas no processo de treinamento dos modelos

[4] <https://github.com/rogersdepelle/OffComBR>
[5] <https://www.python.org/>
[6] <https://radimrehurek.com/gensim/>
[7] <https://www.postgresql.org/>
[8] <https://www.wikimedia.org/>
[9] <https://meta.wikimedia.org/wiki/Research:Detox/Data_Release>

vetoriais, são eles: 50, 100, 200, e 500, Diante dos resultados apresentados, o que teve melhor desempenho nos dados testado foi o valor 100, e os outros parâmetros utilizados foram: $min - count$: 1, $window$: 10, $sample$: 1e-4, $negative$: 5.

Os algoritmos de classificação foram executados a partir da ferramenta de mineração de dados Weka (FRANK; HALL; WITTEN, 2016). O SVM (chamado de SMO no Weka) foi utilizado como classificador base, e realizou uma classificação utilizando uni-gramas. O classificador Naïve Bayes foi utilizado como meta-classificador, os valores padrão dos seus parâmetros foram mantidos. A divisão em conjunto de treinamento e teste se deu por meio de validação cruzada com 10 partições, com exceção do *dataset* kaggle, que já possui esta separação de conjuntos.

A análise dos resultados foi baseada na ROC-AUC e medida F, definidas na Seção 2.3. Para testar se os diferentes resultados foram estatisticamente significativos, foi utilizado o teste de Wilcoxon considerando o nível de significância padrão $\alpha$ = 0,05.

### 6.3 Resultados

Os resultados alcançados são apresentados por meio de tabelas 6.2 e 6.3 que mostram as pontuações para cada componente do conjunto de classificadores e também para o método que os combina, utilizando as métricas medida F e ROC-AUC.

Tabela 6.2: Resultados do classificadores base e do meta-classificador (`Hate2Vec`) utilizando medida F

| Dataset | SVM | HateWord2Vec | HateDoc2Vec | `Hate2Vec` |
|---|---|---|---|---|
| OffComBr2 | 0,76 | 0,94 | 0,95 | **0,97** |
| OffComBr3 | 0,80 | 0,88 | 0,87 | **0,94** |
| Tweets-EN | 0,81 | 0,86 | 0,88 | **0,90** |
| Kaggle | 0,83 | 0,87 | **0,91** | **0,91** |
| Average | 0,80 | 0,88 | 0,90 | **0,93** |

Tabela 6.3: Resultados do classificadores base e do meta-classificador (`Hate2Vec`) utilizando ROC-AUC

| Dataset | SVM | HateWord2Vec | HateDoc2Vec | `Hate2Vec` |
|---|---|---|---|---|
| OffComBr2 | 0,73 | 0,93 | 0,92 | **0,98** |
| OffComBr3 | 0,66 | 0,76 | 0,73 | **0,94** |
| Tweets-EN | 0,77 | 0,84 | 0,87 | **0,93** |
| Kaggle | 0,76 | 0,82 | **0,88** | **0,88** |
| Average | 0,73 | 0,83 | 0,85 | **0,93** |

Ao analisar um dos três componentes isoladamente, o melhor desempenho geral foi o HateDoc2Vec em ambas as métricas, sendo que para dataset Kaggle, o HateDoc2Vec sozinho conseguiu atingir a mesma pontuação que os três métodos combinados. Isso pode ser explicado pelo fato de poder capturar melhor a semântica do comentário inteiro para fazer a previsão. O classificador SVM usando recursos de unigram foi consistentemente o pior desempenho, o que levou a realização de um teste de Wilcoxon, que mostrou que as diferenças entre o `Hate2Vec` e o SVM são estatisticamente significativos em todos os conjuntos de dados.

Comparando os resultados de HateWord2Vec e SVM, notamos que o HateWord2Vec alcança melhores resultados em ambas as métricas. Isso mostra que um classificador baseado em léxico (*i.e.,* HateWord2Vec) pode superar um classificador baseado em BOW para detecção de comentários ofensivos. Esse é um aspecto positivo, pois os classificadores baseados em léxico podem ser empregados em casos onde não há conjuntos de dados anotados e não há necessidade de treinamento adicional.

Tabela 6.4: Avaliação do impacto da remoção de elementos do conjunto de classificadores utilizando medida F

| Dataset | w/o SVM | w/o HateDoc2Vec | w/o HateWord2Vec | All features |
|---|---|---|---|---|
| OffComBr2 | 0,94 | 0,94 | 0,95 | 0,97 |
| OffComBr3 | 0,94 | 0,88 | 0,87 | 0,94 |
| Tweets-EN | 0,90 | 0,86 | 0,88 | 0,90 |
| Kaggle | 0,91 | 0,87 | 0,91 | 0,91 |

A Tabela 6.4 mostra os resultados da medida F quando cada um dos componentes do conjunto de classificadores é removido, isto permite medir a contribuição de cada componente. Como esperado dos resultados das Tabelas 6.2 e 6.3, é notável que a maior

queda ocorreu na execução sem o HateDoc2Vec, já o classificador SVM usando unigramas

**Comparação com os resultados das publicações originais:** Os resultados alcançados também superaram os resultados relatados na literatura para esses conjuntos de dados. Pelle e Moreira (2017) reporta pontuações de medida F de 0,77 (OFFCOMBR-2) e 0,82 (OFFCOMBR-3), enquanto o presente trabalho alcança 0,97 e 0,94, respectivamente. Isso representa ganhos significativos. Para o conjunto de dados Kaggle, o melhor resultado publicado [10] em termos de ROC-AUC foi 0,84, que foi superado pelo 0,91 alcançado pelo método proposto. Uma comparação direta não é possível para o conjunto de dados Tweets-EN (WASEEM; HOVY, 2016) pois resultados são para uma tarefa de classificação de três classes. Considerando esta discrepância, o resultado aqui alçado é 0,9, já no artigo original é de 0,73 para medida F.

**Análise de erros:** A tabela 6.5 mostra métricas para erro de classificação. Em geral, os conjuntos de dados anotados para detecção de conteúdo ofensivo têm uma representação maior da classe de texto não ofensivo, o que gera uma grande quantidade de falsos negativos. Este foi o caso do SVM. `Hate2Vec` tem uma redução considerável dos falsos positivos e falsos negativos, com falsos negativos sendo mais comuns.

Tabela 6.5: Percentual de Falsos positivos e Falsos negativos

| | Falsos positivos | | Falsos negativos | |
|---|---|---|---|---|
| Dataset | SVM | `Hate2Vec` | SVM | `Hate2Vec` |
| OffComBr2 | 0,14 | 0,01 | 0,39 | 0,04 |
| OffComBr3 | 0,09 | 0,05 | 0,57 | 0,05 |
| Tweets-EN | 0,07 | 0,02 | 0,38 | 0,24 |
| Kaggle | 0,09 | 0,05 | 0,37 | 0,17 |
| Average | 0,09 | 0,03 | 0,42 | 0,12 |

Por exemplo, o comentário *"Who cares? The only issue is are they guilt or innocent. All the rest is race baiting by assholes like you."* presente no *dataset* Tweets-EN, foi classificado corretamente como ofensivo por `Hate2Vec`, mas não foi identificado pelo SVM. Para palavras que aparecem apenas uma vez no conjunto de dados, o SVM tem dificuldade em classificar corretamente os comentários que as contêm. Por outro lado, `Hate2Vec` tende a sofrer menos com este problema.

Uma constatação é que ajustes que causam o aumento de de verdadeiros positivos, gera um aumento significativo na taxa de falsos positivos. Em outras situações, o oposto

[10] <https://www.kaggle.com/c/detecting-insults-in-social-commentary/leaderboard>

pode ser preferível, isto é, é preferível obter uma taxa de falsos negativos mais alta. No método proposto isso pode ser alcançado ajustando os limites $t$ e $w$.

Analisando instâncias que foram classificadas erroneamente em `Hate2Vec`, o padrão geral é que elas contêm palavras que podem ser usadas em contextos ofensivos e não ofensivos. Por exemplo, a frase *"Achei que a macaca vivia apenas na floresta ou no zologico"* da OffComBR3 é claramente racista, mas não foi classificado como ofensivo.

## 6.4 Sumário do Capítulo

Neste Capítulo foram apresentados os resultados alçados bem como a análise dos erros e acertos do método proposto. O ganho de performance é bastante claro em todos os *datasets* utilizados, entretanto as comparações diretas nem sempre são válidas em função da configuração do experimento, como é o caso do *dataset* Tweets-EN. Por fim, a análise dos erros permitiu verificar quais aspectos ainda não são cobertos pelo método, e devem ser tratados como desafios na Seção de trabalhos futuros.

# 7 CONCLUSÃO

Este Capítulo apresenta uma síntese das atividades desenvolvidas no decorrer do trabalho, discutindo os resultados obtidos e as contribuições, relacionando a produção cientifica resultante. Por fim são discutidas possibilidades para trabalhos futuros.

## 7.1 Resumo das contribuições

Este trabalho propõe o `Hate2Vec`, um método para detectar comentários ofensivos na Web. Mais especificamente, dado um segmento de texto (que pode ser muito curto), o `Hate2Vec` classifica se este é ou não ofensivo. A proposta é baseada em um conjunto de classificadores no qual o meta-classificador toma como entrada as saídas dos outros três classificadores base, são eles: ($i$) `HateWord2Vec`: um classificador baseado em léxico que conta com uma lista de palavras sementes que expandimos usando *word embeddings*; ($ii$) `HateDoc2Vec`: um classificador de regressão logística que usa representações vetoriais dos comentários como entradas; e ($iii$) um classificador SVM que usa recursos de unigrama.

Foram realizados experimentos utilizando conjuntos de dados em inglês e em português e os resultados foram muito precisos com valores de medida F variando de 0,90 a 0,97. Esses resultados estão dentro do estado da arte em detecção de texto ofensivo. Os experimentos mostraram que o classificador base mais forte foi o HateDoc2Vec, enquanto o mais fraco foi o SVM treinado em unigramas. O `Hate2Vec` apresenta uma baixa taxa de erro, especialmente em termos de falsos positivos. Uma análise de erro identificou que a maioria das instâncias classificadas incorretamente não continham palavrões e eram compostas de palavras que podem ser usadas em contextos não ofensivos.

Este trabalho também descreve a criação de conjuntos de dados anotados sobre postagens ofensivas coletadas de comentários de notícias. Estes *datasets* são chamados OFFCOMBR-2 e OFFCOMBR-3, estão disponíveis para a comunidade. Esta também é uma importante contribuição, dado que os métodos para detecção de conteúdo ofensivo que dependem de aprendizado de máquina supervisionado exigem conjuntos de dados anotados, e embora vários estudos tenham sido conduzidos neste tópico, existem apenas alguns conjuntos de dados disponíveis, e em sua maioria em inglês. Para realizar o processo de anotação destes *datasets* foi desenvolvida um ferramente específica para este fim, também disponibilizada para a comunidade.

## 7.2 Produção científica

Durante o desenvolvimento deste trabalho foram elaborados dois artigos científicos relacionados ao tema da dissertação, são eles:

- A etapa de criação dos *datasets* OFFCOMBR-2 e OFFCOMBR-3 foi publicada e apresentada na edição de 2017 do Brazilian Workshop on Social Network Analysis and Mining (BraSNAM).
- O modelo proposto para detecção comentários ofensivos na web (`Hate2Vec`) foi publicado e apresentado na edição de 2018 do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia).
- O `Hate2Vec` também participou da *OffensEval: Identifying and Categorizing Offensive Language in Social Media*, que é uma das tarefas do International Workshop on Semantic Evaluation (SemEval-2019).

## 7.3 Limitações e trabalhos futuros

No decorrer do desenvolvimento deste trabalho, foram identificadas algumas possíveis melhorias para solução proposta, gerando assim possibilidades de trabalhos futuros. É possível destacar:

1. **Aumentar o tamanho dos *datasets* em PT-BR:** Como discutido no Capítulo 4, o tamanho e a qualidade do *dataset* é de extrema importância. Sendo assim faz-se necessário aumentar o numero de instâncias, entretanto ao se considerar o custo da anotação, uma alternativa viável é construir um *dataset* a partir de comentários e postagens em PT-BR, reportados por outros usuários em sites e redes sociais, como já foi feito para inglês.
2. **Mapear autores:** Um novo atributo que pode ser considerado é um identificador de usuário, que ainda que de modo anônimo, permita identificar outros comentários do mesmo autor. Isso permitirá descobrir a prevalência de comentários ofensivos por parte do usuário, podendo ser usado como evidência adicional por métodos automáticos para filtrar texto ofensivo.
3. **Testar o modelo proposto em outros *datasets*:** Realizar todos os testes em *datasets* em outros idiomas, verificando se o desempenho se matem. Utilizar instâncias maiores de texto tanto no treinamento quanto no teste, como é o caso de postagens

em blogs.

4. **Gerar saídas não binárias:** Cada classificador base gera uma saída binária, isto cria a necessidade de se utilizar três entradas no meta-classificador, uma possibilidade é gerar saídas entre o intervalo de 0 a 1, que representaria um percentual de ofensividade. Isto permitiria utilizar apenas `HateWord2Vec` e o `HateDoc2Vec`, uma vez o SVM pouco contribuiu para o resultado final.
5. **Aplicar aprendizado contínuo:** Assim como discutido na Capítulo 1, uma das dificuldades encontradas é que o vocabulário dos usuários da web é muito dinâmico, então ferramentas podem perder sua capacidade e se tornarem obsoletas muito rapidamente. Uma possível solução é o aprendizado contínuo de representação vetorial de palavras, como o como proposto por Xu et al. (2018), que torna possível incorporação do novo domínio, a sistema já outros domínios, explorando os domínios anteriores por meio de meta-aprendizagem. Neste caso, cada domínio seria u corte temporal das coletas de dados.
6. **Construir um *framework*:** Com o objetivo de tornar a solução proposta mais genérica e usável, é necessário o desenvolvimento de um *framework*, onde as sentenças possaem ser submetidas para serem classificadas, o que também pode ser usado para realimentar o classificador.

# REFERÊNCIAS

AGARWAL, S.; SUREKA, A. Characterizing linguistic attributes for automatic classification of intent based racist/radicalized posts on tumblr micro-blogging website. **arXiv preprint arXiv:1701.04931**, 2017.

AGHAEI, S.; NEMATBAKHSH, M. A.; FARSANI, H. K. Evolution of the world wide web: From web 1.0 to web 4.0. **International Journal of Web & Semantic Technology**, Academy & Industry Research Collaboration Center (AIRCC), v. 3, n. 1, p. 1, 2012.

AL-GARADI, M. A.; VARATHAN, K. D.; RAVANA, S. D. Cybercrime detection in online communications. **Comput. Hum. Behav.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 63, n. C, p. 433–443, oct 2016. ISSN 0747-5632.

ALMEIDA, T. G. et al. Detecting hate, offensive, and regular speech in short comments. In: **Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web**. New York, NY, USA: ACM, 2017. (WebMedia '17), p. 225–228. ISBN 978-1-4503-5096-9.

BADJATIYA, P. et al. Deep learning for hate speech detection in tweets. In: INTERNATIONAL WORLD WIDE WEB CONFERENCES STEERING COMMITTEE. **Proceedings of the 26th International Conference on World Wide Web Companion**. [S.l.], 2017. p. 759–760.

BASHAR, M. A. et al. Misogynistic tweet detection: Modelling cnn with small datasets. Springer, 2018.

BENGIO, Y. et al. A neural probabilistic language model. **J. Mach. Learn. Res.**, JMLR.org, v. 3, p. 1137–1155, mar. 2003. ISSN 1532-4435. Available from Internet: <http://dl.acm.org/citation.cfm?id=944919.944966>.

BIGELOW, J. L.; (KONTOSTATHIS), A. E.; EDWARDS, L. Detecting cyberbullying using latent semantic indexing. In: **Proceedings of the First International Workshop on Computational Methods for CyberSafety**. New York, NY, USA: ACM, 2016. (CyberSafety'16), p. 11–14. ISBN 978-1-4503-4650-4.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: ACM. **Proceedings of the fifth annual workshop on Computational learning theory**. [S.l.], 1992. p. 144–152.

BRADLEY, A. P. The use of the area under the roc curve in the evaluation of machine learning algorithms. **Pattern recognition**, Elsevier, v. 30, n. 7, p. 1145–1159, 1997.

BRETSCHNEIDER, U.; PETERS, R. Detecting offensive statements towards foreigners in social media. In: **50th Hawaii International Conference on System Sciences, HICSS 2017**. Hilton Waikoloa Village, Hawaii, USA: AIS Electronic Library (AISeL), 2017. p. 2213–2222.

BRETSCHNEIDER, U.; WÖHNER, T.; PETERS, R. Detecting online harassment in social networks. In: **Proceedings of the International Conference on Information Systems - Building a Better World through Information Systems, ICIS 2014**. Auckland, New Zealand: Association for Information Systems, 2014. p. 1–14.

BURNAP, P.; WILLIAMS, M. Hate Speech , Machine Classification and Statistical Modelling of Information Flows on Twitter : Interpretation and Communication for Policy Decision Making. **Internet, Policy & Politics**, v. 6, p. 1–18, 2014. ISSN 19442866.

BURNAP PETEAND WILLIAMS, M. L. Us and them: identifying cyber hate on twitter across multiple protected characteristics. **EPJ Data Science**, v. 5, n. 1, p. 11, Mar 2016. ISSN 2193-1127.

CHATZAKOU, D. et al. Mean birds: Detecting aggression and bullying on twitter. In: **Proceedings of the 2017 ACM on Web Science Conference**. New York, NY, USA: ACM, 2017. (WebSci '17), p. 13–22. ISBN 978-1-4503-4896-6.

CHEN, T. Y.; KUO, F.-C.; MERKEL, R. On the statistical properties of the f-measure. In: IEEE. **Quality Software, 2004. QSIC 2004. Proceedings. Fourth International Conference on**. [S.l.], 2004. p. 146–153.

CHEN, Y. et al. Detecting offensive language in social media to protect adolescent online safety. In: **Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust**. Washington, DC, USA: IEEE Computer Society, 2012. (SOCIALCOM-PASSAT '12), p. 71–80. ISBN 978-0-7695-4848-7.

CORTIS, K.; HANDSCHUH, S. Analysis of cyberbullying tweets in trending world events. In: **Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business**. New York, NY, USA: ACM, 2015. (i-KNOW '15), p. 7:1–7:8. ISBN 978-1-4503-3721-2. Available from Internet: <http://doi.acm.org/10.1145/2809563.2809605>.

DADVAR, M.; JONG, F. de. Cyberbullying detection: A step toward a safer internet yard. In: **Proceedings of the 21st International Conference on World Wide Web**. New York, NY, USA: ACM, 2012. (WWW '12 Companion), p. 121–126. ISBN 978-1-4503-1230-1.

DADVAR, M.; TRIESCHNIGG, D.; JONG, F. de. Expert knowledge for automatic detection of bullies in social networks. In: TU DELFT. **25th Benelux Conference on Artificial Intelligence, BNAIC 2013**. Delft, Netherlands: TU Delft, 2013. p. 57–64.

DAVIDSON, T. et al. Automated hate speech detection and the problem of offensive language. In: **Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017**. Montréal, Québec, Canada: AAAI Press, 2017. p. 512–515.

DINAKAR, K. et al. Common sense reasoning for detection, prevention, and mitigation of cyberbullying. **ACM Trans. Interact. Intell. Syst.**, ACM, New York, NY, USA, v. 2, n. 3, p. 18:1–18:30, sep. 2012. ISSN 2160-6455. Available from Internet: <http://doi.acm.org/10.1145/2362394.2362400>.

DJURIC, N. et al. Hate speech detection with comment embeddings. In: **Proceedings of the 24th International Conference on World Wide Web**. New York, NY, USA: ACM, 2015. (WWW '15 Companion), p. 29–30. ISBN 978-1-4503-3473-0.

DUCHARME, D. N. **Machine Learning for the Automated Identification of Cyberbullying and Cyberharassment**. Thesis (PhD) — University of Rhode Island, 2017.

FIŠER, D.; ERJAVEC, T.; LJUBEŠIĆ, N. Legal framework, dataset and annotation schema for socially unacceptable online discourse practices in slovene. In: **Proceedings of the First Workshop on Abusive Language Online**. Association for Computational Linguistics, 2017. p. 46–51. Available from Internet: <http://aclweb.org/anthology/W17-3007>.

FLEISS, J. L.; COHEN, J. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. **Educational and psychological measurement**, Sage Publications Sage CA: Thousand Oaks, CA, v. 33, n. 3, p. 613–619, 1973.

FOUNTA, A.-M. et al. A unified deep learning architecture for abuse detection. **arXiv preprint arXiv:1802.00385**, 2018.

FRANK, E.; HALL, M. A.; WITTEN, I. H. The weka workbench. online appendix for "data mining: Practical machine learning tools and techniques",fourth edition. In: **Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2016. Available from Internet: <https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf>.

GAO, L.; KUPPERSMITH, A.; HUANG, R. Recognizing explicit and implicit hate speech using a weakly supervised two-path bootstrapping approach. In: **Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. Taipei, Taiwan: Asian Federation of Natural Language Processing, 2017. p. 774–782.

GITARI, N. D. et al. A lexicon-based approach for hate speech detection. **International Journal of Multimedia and Ubiquitous Engineering**, v. 10, n. 4, p. 215–230, 2015.

GOUTTE, C.; GAUSSIER, E. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: SPRINGER. **European Conference on Information Retrieval**. [S.l.], 2005. p. 345–359.

GREEVY, E.; SMEATON, A. F. Classifying racist texts using a support vector machine. In: **Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: ACM, 2004. (SIGIR '04), p. 468–469. ISBN 1-58113-881-4.

HASANUZZAMAN, M.; DIAS, G.; WAY, A. Demographic word embeddings for racism detection on twitter. In: **Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. Taipei, Taiwan: Asian Federation of Natural Language Processing, 2017. p. 926–936.

HEE, C. V. et al. Detection and fine-grained classification of cyberbullying events. In: **Proceedings of the International Conference Recent Advances in Natural Language Processing**. Hissar, Bulgaria: INCOMA Ltd. Shoumen, BULGARIA, 2015. p. 672–680.

HOSSEINMARDI, H. et al. Analyzing labeled cyberbullying incidents on the instagram social network. In: LIU, T.-Y.; SCOLLON, C. N.; ZHU, W. (Ed.). **Social Informatics**. Cham: Springer International Publishing, 2015. p. 49–66. ISBN 978-3-319-27433-1.

JR, D. W. H.; LEMESHOW, S.; STURDIVANT, R. X. **Applied logistic regression**. [S.l.]: John Wiley & Sons, 2013.

KENNEDY, G. et al. Technology solutions to combat online harassment. In: **Proceedings of the First Workshop on Abusive Language Online**. Vancouver, BC, Canada: Association for Computational Linguistics, 2017. p. 73–77.

KONTOSTATHIS, A. et al. Detecting cyberbullying: Query terms and techniques. In: **Proceedings of the 5th Annual ACM Web Science Conference**. New York, NY, USA: ACM, 2013. (WebSci '13), p. 195–204. ISBN 978-1-4503-1889-1.

KSHIRSAGAR, R. et al. Predictive embeddings for hate speech detection on twitter. **CoRR**, abs/1809.10644, 2018. Available from Internet: <http://arxiv.org/abs/1809.10644>.

KWOK, I.; WANG, Y. Locate the hate: Detecting tweets against blacks. In: **Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence**. [S.l.]: AAAI Press, 2013. (AAAI'13), p. 1621–1622.

LE, Q.; MIKOLOV, T. Distributed representations of sentences and documents. In: **Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32**. Beijing, China: JMLR.org, 2014. (ICML'14), p. II–1188–II–1196.

LEVY, O.; GOLDBERG, Y. Linguistic regularities in sparse and explicit word representations. In: **Proceedings of the eighteenth conference on computational natural language learning**. [S.l.: s.n.], 2014. p. 171–180.

LIU, S.; FORSS, T. Combining n-gram based similarity analysis with sentiment analysis in web content classification. In: SCITEPRESS-SCIENCE AND TECHNOLOGY PUBLICATIONS, LDA. **Proceedings of the International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management-Volume 1**. [S.l.], 2014. p. 530–537.

MAGU, R.; JOSHI, K.; LUO, J. Detecting the hate code on social media. In: **Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017**. Montréal, Québec, Canada: AAAI Press, 2017. p. 608–611.

MEDHAT, W.; HASSAN, A.; KORASHY, H. Sentiment analysis algorithms and applications: A survey. **Ain Shams Engineering Journal**, Elsevier, v. 5, n. 4, p. 1093–1113, 2014.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **CoRR**, abs/1301.3781, p. 1–12, 2013.

MIKOLOV, T.; YIH, W.-t.; ZWEIG, G. Linguistic regularities in continuous space word representations. In: **Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.: s.n.], 2013. p. 746–751.

NAHAR, V.; LI, X.; PANG, C. An effective approach for cyberbullying detection. **Communications in Information Science and Management Engineering**, World Academic Publishing LTD, v. 3, n. 5, p. 238–247, 2013.

NANDHINI, B.; SHEEBA, J. Cyberbullying detection and classification using information retrieval algorithm. In: ACM. **Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015)**. [S.l.], 2015. p. 20.

NOAVAS/B. **The Webcertain Global Search & Social Report**. 2016. Available from Internet: <http://www.comunicaquemuda.com.br/dossie/quando-intolerancia-chega-as-redes/>.

NOBATA, C. et al. Abusive language detection in online user content. In: **Proceedings of the 25th International Conference on World Wide Web**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2016. (WWW '16), p. 145–153. ISBN 978-1-4503-4143-1.

NOCKLEBY, J. T. Hate speech. In: **Encyclopedia of the American Constitution (2nd ed.,edited by Leonard W. Levy, Kenneth L. Karst et al., New York: Macmillan, 2000)**. [S.l.: s.n.], 2000. p. 1277–1279.

PAVLOPOULOS, J.; MALAKASIOTIS, P.; ANDROUTSOPOULOS, I. Deep learning for user comment moderation. **CoRR**, abs/1705.09993, p. 1–11, 2017.

PELLE, R. P. de; MOREIRA, V. P. Offensive comments in the brazilian web: a dataset and baseline results. In: **Brazilian Workshop on Social Network Analysis and Mining (BraSNAM) in conjunction witth Congresso da Sociedade Brasileira de Computação-CSBC**. São Paulo, SP, Brazil: Sociedade Brasileira de Computação, 2017. p. 510–519.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**. [S.l.: s.n.], 2014. p. 1532–1543.

PETE, B.; L., W. M. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. **Policy & Internet**, v. 7, n. 2, p. 223–242, 2015.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. Bioinfo Publications, 2011.

REYNOLDS, K.; KONTOSTATHIS, A.; EDWARDS, L. Using machine learning to detect cyberbullying. In: **Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops - Volume 02**. Washington, DC, USA: IEEE Computer Society, 2011. (ICMLA '11), p. 241–244. ISBN 978-0-7695-4607-0.

ROSS, B. et al. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. In: **Proceedings of the 3rd Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC III)**. Bochum, Germany: Bochumer linguistische Arbeitsberichte, 2016. p. 6–9.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence: a modern approach**. [S.l.]: Malaysia; Pearson Education Limited,, 2016.

SALEEM, H. M. et al. A web of hate: Tackling hateful speech in online social spaces. In: **Proceedings of the LREC 2016 Workshop on Text Analytics for Cybersecurity and Online Safety (TA-COS 2016)**. Portorož, Slovenia: European Language Resources Association (ELRA), 2016. p. 1–10.

SAMGHABADI, N. S. et al. Detecting nastiness in social media. In: **Proceedings of the First Workshop on Abusive Language Online**. Vancouver, BC, Canada: Association for Computational Linguistics, 2017. p. 63–72.

SCHMIDT, A.; WIEGAND, M. A survey on hate speech detection using natural language processing. In: **Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media**. Valencia, Spain: Association for Computational Linguistics, 2017. p. 1–10.

SILVA, R. L. da et al. Discurso de ódio em redes sociais: jurisprudência brasileira. **Revista direito GV**, SciELO Brasil, v. 7, n. 2, p. 445–467, 2011.

SOOD, S. O.; CHURCHILL, E. F.; ANTIN, J. Automatic identification of personal insults on social news sites. **JASIST**, v. 63, n. 2, p. 270–285, 2012.

SU, H.-P. et al. Rephrasing profanity in chinese text. In: **Proceedings of the First Workshop on Abusive Language Online**. Vancouver, BC, Canada: Association for Computational Linguistics, 2017. p. 18–24.

TING, I.-H. et al. An approach for hate groups detection in facebook. In: UDEN, L. et al. (Ed.). **The 3rd International Workshop on Intelligent Data Analysis and Management**. Dordrecht: Springer Netherlands, 2013. p. 101–106. ISBN 978-94-007-7293-9.

TING, I. H. et al. Content matters: A study of hate groups detection based on social networks analysis and web mining. In: **2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)**. [S.l.: s.n.], 2013. p. 1196–1201.

TULKENS, S. et al. The automated detection of racist discourse in dutch social media. **Computational Linguistics in the Netherlands Journal**, v. 6, p. 3–20, 12/2016 2016. ISSN 2211-4009.

VIGNA, F. D. et al. Hate me, hate me not: Hate speech detection on facebook. In: **Proceedings of the First Italian Conference on Cybersecurity (ITASEC17)**. Venice, Italy: CEUR-WS.org, 2017. p. 86–95.

WARNER, W.; HIRSCHBERG, J. Detecting hate speech on the world wide web. In: **Proceedings of the Second Workshop on Language in Social Media**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012. (LSM '12), p. 19–26.

WASEEM, Z.; HOVY, D. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In: **Proceedings of the NAACL Student Research Workshop**. San Diego, California: Association for Computational Linguistics, 2016. p. 88–93.

WOLPERT, D. H. Stacked generalization. **Neural Networks**, v. 5, p. 241–259, 1992.

WULCZYN, E.; THAIN, N.; DIXON, L. Ex machina: Personal attacks seen at scale. In: **Proceedings of the 26th International Conference on World Wide Web**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2017. (WWW '17), p. 1391–1399. ISBN 978-1-4503-4913-0.

XIANG, G. et al. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In: **Proceedings of the 21st ACM International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2012. (CIKM '12), p. 1980–1984. ISBN 978-1-4503-1156-4.

XU, H. et al. Lifelong domain word embedding via meta-learning. In: **Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18**. [S.l.: s.n.], 2018. p. 4510–4516.

XU, J.-M. et al. Learning from bullying traces in social media. In: **Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012. (NAACL HLT '12), p. 656–666. ISBN 978-1-937284-20-6. Available from Internet: <http://dl.acm.org/citation.cfm?id=2382029.2382139>.

XU, Z.; ZHU, S. Filtering offensive language in online communities using grammatical relations. In: **Proceedings of the Seventh Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference, CEAS 2010**. Redmond, Washington, USA: CEAS, 2010. p. 20–29.

YUAN, S.; WU, X.; XIANG, Y. A two phase deep learning model for identifying discrimination from tweets. In: **EDBT**. [S.l.: s.n.], 2016. p. 696–697.

ZHONG, H. et al. Content-driven detection of cyberbullying on the instagram social network. In: **Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence**. New York, New York, USA: AAAI Press, 2016. (IJCAI'16), p. 3952–3958. ISBN 978-1-57735-770-4.