UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

MARCELLO DÓRIO QUEIROZ

APRENDIZADO DE MÁQUINA PARA DETECÇÃO AUTOMÁTICA DE DISCURSO DE ÓDIO: UMA COMPARAÇÃO ENTRE OS CLASSIFICADORES SVM E CNN

VITÓRIA
2021

MARCELLO DÓRIO QUEIROZ

APRENDIZADO DE MÁQUINA PARA DETECÇÃO AUTOMÁTICA DE DISCURSO DE ÓDIO: UMA COMPARAÇÃO ENTRE OS CLASSIFICADORES SVM E CNN

Parte manuscrita do Projeto de Graduação do aluno **Marcello Dório Queiroz**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof. Dr. Patrick Marques Ciarelli

VITÓRIA
2021

MARCELLO DÓRIO QUEIROZ

APRENDIZADO DE MÁQUINA PARA DETECÇÃO AUTOMÁTICA DE DISCURSO DE ÓDIO: UMA COMPARAÇÃO ENTRE OS CLASSIFICADORES SVM E CNN

Parte manuscrita do Projeto de Graduação do aluno **Marcello Dório Queiroz**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 14 de maio de 2021

COMISSÃO EXAMINADORA:

Prof. Dr. Patrick Marques Ciarelli
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. Evandro Ottoni Teatini Salles
Universidade Federal do Espírito Santo
Examinador

Prof. Dr. Jorge Leonid Aching Samatelo
Universidade Federal do Espírito Santo
Examinador

AGRADECIMENTOS

Agradeço a minha mãe por ter dedicado a vida à educação dos filhos.

A meu irmão pela ajuda e incentivo sem os quais este trabalho não teria sido feito.

A minha irmã e sobrinhos pelo apoio e carinho.

A todos os amigos e amores que fizeram parte da minha vida ao longo desses anos.

A todos os professores que me acompanharam de perto durante a graduação.

Ao PET Elétrica por ter sido um lar dentro da universidade.

A meus orientadores e tutores, todos eles, pelos ensinamentos e pela paciência.

Por fim, à Ufes, por ter me formado, além de engenheiro, cidadão.

RESUMO

A popularização e o crescente uso de redes sociais traz consigo as complicações advindas da confluência de pessoas e ideias diferentes numa mesma plataforma. No *Twitter*, por exemplo, é notória a proliferação de discursos de ódio e linguagem ofensiva. Identificá-los é importante para manter as redes sociais como ambientes saudáveis para os usuários e diminuir a necessidade de trabalhadores que moderam esses espaços. A detecção de discursos de ódio se configura como um problema de classificação de textos, pertencente ao ramo da Inteligência Artificial chamado de Processamento de Linguagem Natural. Diferentes técnicas têm sido testadas e este trabalho apresenta uma pesquisa que tem como objetivo avaliar o desempenho de redes neurais convolucionais (CNN, do inglês *convolutional neural networks*) na identificação de discurso de ódio e compará-lo com o desempenho de uma técnica tradicional, aqui representada pelas máquinas de vetor suporte (SVM, do inglês *support vector machines*). O banco de dados é composto por tuítes em língua inglesa rotulados em 3 classes: discurso de ódio, linguagem ofensiva e neutro. A extração de características é feita com *n-grams* e o método que calcula a frequência do termo–inverso da frequência nos documentos (TF-IDF, do inglês *term frequency-inverse document frequency*) para o SVM e com o modelo pré-treinado *GloVe Twitter* de *word embeddings* para as CNN. A arquitetura de CNN escolhida já foi utilizada na tarefa de detecção de discurso de ódio com bons resultados. Observou-se no experimento desempenhos superiores de precisão, *recall* e *F-measure* obtidos pela CNN conforme sugerido pela literatura. Entretanto, houve dificuldade em ambos os métodos de classificar corretamente discuso de ódio. Os achados evidenciam a complexidade de distinguir discurso de ódio de linguagem ofensiva no *Twitter* por meio de algoritmos de inteligência artificial, uma vez que o vocabulário utilizado é semelhante em ambas as modalidades.

Palavras-chave: Discurso de ódio. *Twitter*. Redes neurais convolucionais. *Support vector machine*.

ABSTRACT

The popularization and growing use of social networks brings in itself the complications that arise when different people and ideas find each other in the same platform. On Twitter, for exemple, hate speech and offensive language have found a place to proliferate. Identify them is necessary to keep social networks as healthy environments and to diminish the need of moderator workers. Hate speech detection is aproached as a text classification problem belonging to Natural Language Processing field within the realm of Artificial Intelligence. Different techniques have been tested and this monography presents a research aiming to compare Convolutional Neural Networks (CNN) with a baseline method, represented here by Support Vector Machines (SVM). The database is formed by collected tweets labled in 3 classes: hate speech, offensive language and none. Feature extraction for the SVM algorithm is done by finding unigrams and bigrams and applying term frequency-inverse document frequency (TF-IDF). The CNN is fed with GloVe Twitter Embeddings. The chosen architecture has been used in text classification tasks on detecting hatespeech with good results. In our expertiment, the CNN gave higher scores on precision, recall and F-measure as previous literature suggested although both methods have struggled to identify tweets containing hate speech correctly. Our findings make evident the complexity in the task of distinguishing hate speech from offensive language on Twitter since both showcase similar vocabulary.

Keywords: Hate speech. Twitter. Convolutional neural networks. Support vector machines.

LISTA DE FIGURAS

Figura 1 – Etapas do processamento do texto fonte 20
Figura 2 – Visualização em duas dimensões de *word embeddings* de termos associados ao lar 24
Figura 3 – Visualização em duas dimensões das relações vetoriais entre palavras num espaço criado por *word embeddings* 24
Figura 4 – Exemplo de visualização em duas dimensões do funcionamento do algoritmo SVM (a) reta criada pelo SVM na etapa de treinamento (b) classificação realizada pelo SVM sobre o conjunto de teste 25
Figura 5 – Exemplo de visualização em duas dimensões da classificação realizada pelo SVM sobre o conjunto de dados com 3 classes e abordagem "um contra todos". 26
Figura 6 – Modelo matemático de um neurônio 27
Figura 7 – Ilustração da operação realizada pelo neurônio 28
Figura 8 – Modelo de rede neural 28
Figura 9 – Arquitetura de CNN reproduzida no projeto 29
Figura 10 – Ilustração de uma rede com neurônios desconectados durante o processo de *dropout* 30
Figura 11 – Exemplo de filtro e resultado obtido por ele na etapa de convolução . . 31
Figura 12 – Exemplo de filtro que opera sobre *embeddings* e resultado obtido por ele na etapa de convolução (a) filtro com comprimento igual a 2 (b) filtro com comprimento igual a 3 31
Figura 13 – Ilustração das diferentes operações de *pooling* (a) *max pooling* numa janela de tamanho 2 (b) *average pooling* numa janela de tamanho 2 (c) *global max pooling* 32
Figura 14 – Ilustração da atuação da função *softmax* na classificação 32
Figura 15 – Matriz de confusão da classificação realizada pelo SVM normalizada pelas colunas 43
Figura 16 – Matriz de confusão da classificação realizada pelo SVM normalizada pelas linhas 44
Figura 17 – Evolução das funções de perda e precisão e *recall* com os pesos do Quadro 8 em 50 épocas de treinamento 45
Figura 18 – Evolução da função de perda para os conjuntos de treino e validação em 10 épocas com os novos pesos do Quadro 11 46
Figura 19 – Matriz de confusão da classificação realizada pela CNN normalizada pelas colunas 46
Figura 20 – Matriz de confusão da classificação realizada pela CNN normalizada pelas linhas 47

LISTA DE QUADROS

Quadro 1 – Matriz de confusão teórica 33
Quadro 2 – Matriz de confusão com os valores do exemplo da Figura 4(b) 34
Quadro 3 – Matriz de confusão para um conjunto de dados com 3 classes 35
Quadro 4 – Matriz de confusão com os valores do exemplo da Figura 5 35
Quadro 5 – Matriz de confusão do Quadro 4 normalizada pelas colunas 36
Quadro 6 – Matriz de confusão do Quadro 4 normalizada pelas linhas 36
Quadro 7 – Parâmetros e valores testados para otimização do SVM 40
Quadro 8 – Parâmetros e valores testados para otimização da classificação pela rede 41
Quadro 9 – Resultados obtidos nas etapas de pré-processamento 42
Quadro 10 – Parâmetros e valores testados para otimização do SVM 43
Quadro 11 – Parâmetros e valores testados para otimização da classificação pela CNN 45

LISTA DE TABELAS

Tabela 1 – Matriz documento-termo para representação em BOW dos tuítes exemplos 22
Tabela 2 – Matriz documento-termo para representação em BOW após calculado o TF-IDF 23
Tabela 3 – Quantidade e proporção de cada classe no corpus do banco de dados . 38
Tabela 4 – Resultados de precisão, *recall* e *F-measure* obtidos pelos classificadores SVM e CNN para as classes DO, LO, NE e a junção de DO e LO . . . 48

LISTA DE ABREVIATURAS E SIGLAS

1D	Unidimensional
2D	Bidimensional
AM	Aprendizado de Máquina
API	*Application Programming Interface*
BOW	*Bag-of-words*
CNN	*Convolutional Neural Networks*
DO	Discurso de ódio
FN	Falso-negativo
FP	Falso-positivo
GB	*Gradient Boosting*
IA	Inteligência Artificial
LO	Linguagem ofensiva
NE	Neutro
NLTK	*Natural Language Toolkit*
ONU	Organização das Nações Unidas
PLN	Processamento de Linguagem Natural
ReLU	*Rectified Linear Unit*
SVM	*Support Vector Machine*
$\mathbf{t}$	Vetor "*one-hot*" que representa a rótulo de um documento
TF-IDF	*Term Frequency - Inverse Document Frequency*
VN	Verdadeiro-positivo
VP	Verdadeiro-negativo

LISTA DE SÍMBOLOS

C	Número de classes num problema de classificação
d_i	i-ésimo documento do corpus
t_j	j-ésimo termo de um documento
w_{d_i,t_j}	Peso dado ao termo j do documento i na matriz documento-termo
tf_{d_i,t_j}	Frequência do termo j no documento i
N	Quantidade de documentos presente no corpus analisado
df_{t_j}	Quantidade de documentos em que o termo j aparece
x	Valor de entrada de um neurônio ou rede neural
w	Peso de um neurônio
$\mathbf{x}$	Vetor de entradas de um neurônio ou rede neural
$\mathbf{w}$	Vetor de pesos de um neurônio
b	Valor de *bias* de um neurônio
z	Valor resultado da operação realizada na entrada do neurônio
$R(z)$	Função de ativação ReLU
$\mathbf{s}$	Vetor de pontuações obtida na camada de saída
va	Valor absoluto numa matriz de confusão
vnc	Valor normalizado pela coluna numa matriz de confusão
vnl	Valor normalizado pela linha numa matriz de confusão
y	Saída de um neurônio
$\mathbf{y}$	Vetor de saídas de uma rede neural

SUMÁRIO

1 INTRODUÇÃO 12
1.1 Apresentação 12
1.2 Justificativas 13
1.3 Objetivos 15
1.4 Estrutura do Trabalho 15
2 EMBASAMENTO TEÓRICO 16
2.1 O *Twitter* 16
2.2 O Discurso de Ódio 17
2.3 Processamento de Linguagem Natural (PLN) 19
2.3.1 Pré-processamento 21
2.3.2 Modelagem e extração de características 21
2.3.2.1 *Bag-of-words*, *n-grams* e TF-IDF 21
2.3.2.2 *Word embeddings* 23
2.3.3 Máquinas de vetores suporte (SVM) 25
2.3.4 Redes neurais convolucionais (CNN) 27
2.3.5 Métricas de avaliação 33
3 METODOLOGIA E ETAPAS DE DESENVOLVIMENTO 38
3.1 Base de Dados 38
3.2 Metodologia para o SVM 39
3.3 Metodologia para a CNN 40
4 RESULTADOS E DISCUSSÃO 42
5 CONCLUSÃO 49
REFERÊNCIAS BIBLIOGRÁFICAS 50

1 INTRODUÇÃO

1.1 Apresentação

Com a popularização das redes sociais ao redor do mundo, elas têm angariado um número cada vez maior de usuários. Estima-se que em 2020 esse número atinja 3,08 bilhões (CLEMENT, 2020c) e que os usuários passam em média 2,4 horas por dia nas redes sociais (KEMP, 2020).

Umas das redes sociais mais proeminentes é o *Twitter*. Nele, as pessoas são capazes de compartilhar, em tempo real, ideias, sentimentos e opiniões com outras de diversas origens, incluindo personalidades políticas e influentes. O volume de dados gerados na rede e sua fácil obtenção por meio de intaerfaces de programação de aplicações (API, do inglês *application programming interface*) torna o *Twitter* uma rede muito visada por pesquisadores de várias áreas para o estudo de fenômenos sociais que emergem dela.

O encontro de ideias divergentes numa mesma plataforma por vezes culmina, no entanto, na publicação de conteúdo ofensivo e de ódio (BURNAP; WILLIAMS, 2016), que é o ataque ou qualquer forma de comunicação que use linguagem pejorativa ou discriminatória a uma pessoa ou grupo de pessoas com base em fatores identitários como raça, gênero e origem, por exemplo (ORGANIZAÇÃO DAS NAÇÕES UNIDAS, 2019).

Em escala global, o discurso de ódio é uma ameaça a valores democráticos e estabilidade social (ORGANIZAÇÃO DAS NAÇÕES UNIDAS, 2019; UZOCHUKWU; OKAFOR, 2019). Individualmente, afeta a saúde mental de suas vítimas, aumentando índices de estresse (SAHA; CHANDRASEKHARAN; CHOUDHURY, 2019) e números de ocorrências de suicídio (MULLEN, 2004). Dessa forma, torna-se importante proteger usuários das redes sociais desse tipo de conteúdo. Burnap e Williams (2016) encontraram que 67% de adolescentes entre 15 e 18 anos já haviam sido expostos à linguagem ofensiva em redes sociais, dentre os quais 21% afirmavam terem sido alvo dos perpetradores, sendo que essa faixa etária constitui uma parcela considerável do público do *Twitter* (CLEMENT, 2020a). Diante desse cenário, tem se tornado crescente entre os usuários o desejo de evasão das redes sociais, chegando a 32% em setembro de 2020 (CLEMENT, 2020d).

Como resposta ao problema do discurso de ódio nas redes, entidades governamentais têm pressionado as empresas a combater sua disseminação. Um exemplo é o acordo firmado entre a União Europeia e as grandes empresas do setor para melhoria em índices, como velocidade de resposta às denúncias feitas nas plataformas dos sites (EUROPEAN COMISSION, 2020a). Como resultado, esses índices têm melhorado nos últimos anos

(EUROPEAN COMISSION, 2020b), porém, essa melhoria tem ocorrido às custas da saúde de um grande número de moderadores empregados com a responsabilidade de impedir que conteúdo nocivo seja publicado (DWOSKIN; WHALEN; CABATO, 2019).

Outro grupo que tem exercido pressão é o das empresas que promovem seus anúncios publicitários nas redes sociais, as quais também são pressionadas pelos consumidores a demonstrarem o compromisso na luta contra o discurso de ódio nas redes (FUNG, 2020). Cientes disso, redes como o *Twitter* têm tomado medidas para promover um ambiente de discussão mais saudável (HICKS; GASCA, 2019), por exemplo, expandindo o número de grupos protegidos pelas regras e políticas da plataforma (TWITTER, c2020a; TWITTER SAFETY, 2020).

Combinando os fatores apresentados, surge então a necessidade do desenvolvimento de técnicas para detecção automática e eficaz do discurso de ódio nas redes sociais. A exploração da inteligência artificial (IA) no estudo do discurso de ódio na internet pode ser um meio de estudar esse fenômeno para propor soluções aos problemas por ela causados: melhorar a experiência do usuário, diminuir a necessidade de pessoal dedicado à moderação de conteúdo (GÓMEZ, 2019) e ajudar a definir políticas públicas em países cada vez mais preocupados com as ações de seus cidadãos nas redes (UNITED NATIONS CONFERENCE ON TRADE AND DEPELOPMENT, 2020). Sendo assim, este trabalho busca saber qual capacidade algoritmos de IA têm para identificar discurso de ódio em textos publicados no *Twitter*.

1.2 Justificativas

A identificação de discurso de ódio em texto pode ser interpretado como um problema de Processamento de Linguagem Natural (PLN), um ramo da Inteligência Artificial dedicado a analisar a linguagem humana, seja em texto ou áudio. Para solucionar esse problema, diversas técnicas de Aprendizado de Máquina (AM) já foram utilizadas (PITSILIS; RAMAMPIARO; LANGSETH, 2018). Entre elas estão as *Deep Neural Networks* (DNN), redes neurais que se diferem por terem múltiplas camadas entre as camadas de entrada e saída da rede. A principal característica delas é a capacidade de encontrar padrões complexos em conjuntos de dados e têm sido aplicadas em diversas áreas com excelentes resultados (ROSA *et al.*, 2018).

Badjatiya e outros (2017) usaram DNN na detecção de discurso de ódio em tuítes e tiveram seu melhor resultado usando uma arquitetura de *Recursive Neural Network* (RNN), um tipo de DNN. Pitsilis, Ramampiaro e Langseth (2018) colaboraram com esse

resultado realizando diferentes testes também com RNN. Park e Fung (2017), por sua vez, compararam os resultados de classificação em uma base de dados usando um outro tipo de DNN, chamadas *Convolutional Neural Networks* (CNN), em dois métodos. Em um deles, os dados eram classificados em três classes (neutro, misoginia e racismo) a partir de uma única rede. Em outro, primeiro os tuítes eram classificados entre sendo conteúdo neutro ou não para em seguida, numa segunda rede, os que haviam sido classificados como ódio, eram depois classificados entre misoginia e racismo. Melhores resultados foram obtidos com a classificação em uma etapa, mas resultados promissores também foram obtidos com a classificação em duas etapas.

Uma outra técnica é o *Support Vector Machine* (SVM). Ele é um método de AM que objetiva encontrar na fase de treinamento a função no espaço que melhor separa um subconjunto dos dados já classificados, assim dividindo o espaço em uma região para cada classificação. Em seguida, na fase de teste, ao se projetar o restante dos dados no espaço, encontra-se a classificação de cada um deles. É uma técnica já consolidada no problema de classificação de textos pois tem a capacidade de generalizar bons modelos mesmo com o número alto de dimensões necessários para representar as características de um texto (JOACHIMS, 1998). Oriola e Kotzé (2020) fizeram um comparativo entre SVM, Regressão Logística, *Gradient Boosting* (GB) e *Random Forest* na tarefa de na identificação de discurso de ódio em tuítes e obtiveram melhores resultados com SVM e GB. Rosa e outros (2018), por sua vez, compararam SVM e diferentes arquiteturas de CNN e encontraram taxas de precisão de 84,6% e 85,0%, respectivamente, na detecção de *cyberbullying*. Essa pequena diferença indica haver abertura na literatura para comprovar a real eficácia das CNN. Já o SVM, com base nos argumentos apresentados, foi escolhido como método *baseline* de comparação com as CNN.

Apesar de as CNN serem computacionalmente mais custosas durante o treinamento, técnicas mais rudimentares são incapazes de distinguir contexto e nuance. Gomes, Antonialli e Oliva (2019) analisaram uma ferramenta que mede o nível de toxicidade de um texto na Internet e identificaram que a ferramenta considerava tuítes discriminatórios de supremacistas brancos menos tóxicos que tuítes de *drag queens* por nestes existirem palavras consideradas ofensivas se lidas fora de contexto. Assim, se forem capazes de fornecer resultados mais precisos, CNN estarão protegendo a liberdade de expressão dos usuários de redes sociais, impedindo que a ferramenta sirva para silenciar grupos já marginalizados e alvos de discurso de ódio nas redes.

1.3 Objetivos

Este projeto tem como objetivo geral explorar as potencialidades das redes neurais, em especial as CNN, em tarefas envolvendo processamento de linguagem natural. Considerando o levantamento bibliográfico realizado ao longo deste trabalho, que indica maior eficácia das CNN em comparação a técnicas tradicionais de AM (como o SVM) ao realizar tarefas que são semelhantes ao proposto neste trabalho, levanta-se a hipótese de que os resultados a serem obtidos com tal técnica serão superiores.

Os objetivos específicos do projeto são:

- Aplicar o SVM e uma arquitetura CNN na detecção de discurso de ódio em textos retirados do *Twitter*;
- Comparar o desempenho das técnicas usando métricas estatísticas.

Para atingir esses objetivos, será utilizada uma base de dados pública já rotulada, elaborada por Davidson e outros (2017).

1.4 Estrutura do Trabalho

Este trabalho se divide em 5 seções. Na seção 2 serão expostos os conceitos e explicadas as técnicas a serem utilizadas no desenvolvimento do projeto. Na seção 3 tem-se a metodologia aplicada para atingir seu objetivo. Na seção 4 discutem-se os resultados obtidos e na seção 5 estão as conclusões. Por fim, na seção pós-textual se encontram as referências bibliográficas que embasam o trabalho.

2 EMBASAMENTO TEÓRICO

2.1 O *Twitter*

O *Twitter* é uma rede social criada em 2006 e hoje possui uma estimativa de alcance de 316 milhões de pessoas em todo o mundo (CLEMENT, 2020b). É usada por pessoas de todas as idades, entre elas anônimos, artistas, celebridades e autoridades políticas. Ao longo dos anos catalisou manifestações sociais (LOZANO *et al.*, 2017) e serviu de palco para crises políticas (AGOSTINE, 2020).

O *Twitter* pede ao usuário para responder a pergunta "O que está acontecendo?", o que incita os usuários a fazerem publicações curtas e frequentes compartilhando ideias, pensamentos, sentimentos e situações cotidianas. Por isso, classifica-se como parte do conjunto dos microblogs, redes sociais que se caracterizam pelo grande número de publicações por dia e pela natureza instantânea das interações (ARCENEAUX; WEISS, 2010). Suas funções mais básicas estão listadas a seguir:

- Tuíte (do inglês *tweet*): uma "mensagem publicada no *Twitter* que contém texto, fotos, um GIF e/ou um vídeo" (TWITTER, c2020b). O texto é limitado em 280 caracteres;
- Seguir (do inglês *follow*): ato de se inscrever para ver o conteúdo publicado por outro perfil, tornando-se seguidor dele. No *Twitter*, em contraste com plataformas como *Facebook*, o relacionamento entre perfis é unidirecional, ou seja, A pode seguir B sem que B siga A;
- Favoritar: ato que exprime apreciação pelo tuíte;
- Retuitar (do inglês *retweet*): compartilhar com seus seguidores o tuíte de outro perfil;
- Menção: um tuíte contendo o nome de usuário de outro perfil precedido do símbolo arroba (@);
- Resposta (do inglês *reply*): um tuíte que começa com o nome de usuário de outro perfil em resposta a um de seus tuítes;
- Bloquear: impedir que um determinado perfil veja suas publicações e também deixar de ver o que esse perfil publica.

O conteúdo compartilhado pelos usuários é público e pode ser visto por um número maior de pessoas se forem usadas *hashtags*. Elas são capitaneadas pelo símbolo cerquilha (#)

e servem para concentrar e sinalizar os tuítes que se referem a um mesmo assunto ou movimento. Exemplos de *hashtags* recentes e relevantes são *#metoo* e *#BlackLivesMatter*. No primeiro, mulheres denunciaram situações em que sofreram assédio e, no segundo, pessoas negras denunciaram a violência policial a que estão submetidas. Pelo seu poder agregador, pesquisadores costumam usar as *hashtags* como parâmetro para coleta de tuítes para formar bancos de dados em pesquisas. Malini (2016) usou os termos #VemPraRua e #passelivre para estudar a relação entre as manifestações de junho de 2013 e o conteúdo publicado no *Twitter* sobre o tema. Lozano e outros (2017) coletou tuítes contendo #hillary2016, *#imwithher*, #trump2016 e *#makeamericagreatagain* num estudo que tentou identificar racismo no contexto político das eleições estadunidenses de 2016.

O *Twitter* não exige que os usuários forneçam seu nome real ao criar uma conta, permitindo pseudônimos. Peddinti, Ross e Cappos (2017) tentaram estimar a quantidade de usuários identificáveis e verificar se esses usuários eram mais prováveis de consumir ou publicar conteúdo sensível, como islamofobia, pornografia e supremacia branca. Foi encontrado que pelo menos 26% dos usuários não são identificáveis e que 21,6% daqueles que seguem perfis que publicam conteúdo sensível eram anônimos. Além disso, Peddinti, Ross e Cappos (2017) verificaram que perfis de notícias contêm também 6,6% de seguidores não identificáveis, o que pode indicar que o anonimato seja uma prática generalizada entre os usuários da plataforma e que os mesmos perfis que consomem e publicam conteúdo sensível, também consomem conteúdo não sensível.

Outros pesquisadores analisam o *Twitter* por uma abordagem diferente e o remetem a um conceito da psicologia social, a desindividuação, um processo de perda de autoconsciência em grupos que causa comportamento desinibido e antinormativo. Burnap e Williams (2015) usam esse conceito para afirmar que em redes sociais é mais fácil os usuários expressarem opiniões de forma mais emotiva. Lee (2007) estudou o efeito desse fenômeno em comunicações mediadas por computador e relatou que os participantes do estudo eram mais propensos a polarizar suas opiniões numa discussão sobre um tema controverso e que tendiam a manter o anonimato ao opinarem.

Sabendo-se a forma como o *Twitter* funciona e tendo-se em mente as ideias apresentadas, pode-se inferir que a plataforma é um ambiente com potencial para a manifestação de ideias e sentimentos nocivos aos usuários, como o discurso de ódio.

2.2 O Discurso de Ódio

O discurso de ódio é entendido como

> [...] qualquer forma de comunicação, em fala, escrita ou comportamento, que ataque ou use linguagem pejorativa ou discriminatória que se refere a uma pessoa ou grupo com base em quem são, ou seja, em sua religião, etnia, nacionalidade, raça, cor, ascendência, gênero ou outro fator identitário (ORGANIZAÇÃO DAS NAÇÕES UNIDAS, 2019, p. 2, tradução nossa).

No Brasil, apesar de não definido explicitamente como tal, o discurso de ódio está vedado pelo Art. 20 da Lei no. 1116/89 que proíbe: "Praticar, induzir ou incitar a discriminação ou preconceito de raça, cor, etnia, religião ou procedência nacional" (BRASIL, 1989, online).

Em sua página que trata das políticas sobre conduta odiosa, o *Twitter* estabelece que "não é permitido promover violência, atacar diretamente ou ameaçar outras pessoas com base em raça, etnia, nacionalidade, orientação sexual, sexo, identidade de gênero, religião, idade, deficiência ou doença grave" (TWITTER, c2020a, online, tradução nossa). Afirma também que não permite "contas cuja finalidade principal seja incitar lesões a outros com base nessas categorias" (TWITTER, c2020a, online, tradução nossa).

No âmbito da detecção automática, alguns pesquisadores diferenciam discurso de ódio de linguagem ofensiva em suas classificações. Oriola e Kotzé (2020) consideram ódio quando além de palavras discriminatórias, o tuíte incita a violência e cita e dá apoio a organizações preconceituosas. Watanabe, Bouazizi e Ohtsuki (2018), por sua vez, distinguem ódio de ofensa quando há expressões racistas e segregacionistas.

Para além do problema da definição, surgem os problemas para a identificação. A linguagem utilizada pelos usuários do *Twitter* pode apresentar muitos desafios. É comum encontrar erros ortográficos, abreviações e expressões informais. Além disso, os usuários aprenderam a trocar caracteres ou fazer anagramas de forma que a palavra continue legível, mas incapacitando que um programa reconheça aquela palavra sem que tenha sido preparado para fazê-lo. Um exemplo dessa estratégia são as diversas formas que o nome Bolsonaro é escrito para evitar que o termo suba no ranking de assuntos mais falados: "bonoro", "biroliro" e "b1roliro", por exemplo.

Outro desafio é o reconhecimento do contexto, não só onde a frase está escrita mas também da relação entre o emissor e o receptor da mensagem. A frase "to com raiva de vc tu me deixou no vácuo v1ado" não deveria ser interpretada como discurso de ódio porque, apesar de ter uma palavra com tom pejorativo, nesse contexto indica proximidade entre os interlocutores. Esse tuíte também ilustra como as palavras são modificadas no contexto de coloquialidade existente no *Twitter*. Há tuítes, no entanto, que não aparentam ser ofensivos, mas têm seu conteúdo ofensivo revelado pelo contexto, como na frase "*mulher

no singular"[1]. Ela é resposta a um tuíte de divulgação de uma chapa para a prefeitura de São Paulo onde duas mulheres aparecem numa foto, uma cisgênero e a outra, transgênero. Nele, havia a frase: "É a vez das mulheres!". Logo, o tuíte com a frase "*mulher no singular" pretende corrigir a afirmação de haver duas mulheres na foto, revelando seu caráter transfóbico quando conhecido o contexto.

Há tuítes em que a mensagem é claramente odiosa, mas uma máquina deve ser sofisticada o suficiente para compreender as relações sintáticas e semânticas entre as palavras. A frase "Homofobia não é crime, é inevitável" é uma delas. Nesse caso, a máquina deveria entender os conceitos de homofobia e crime e o significado de inevitável, para então compreender "homofobia" como sujeito da frase, "crime" como predicativo do sujeito, e "é" como verbo de ligação negado pelo advérbio de negação "não". Em seguida, associar o predicativo do sujeito "inevitável" ao sujeito "homofobia" oculto pela elipse. Pode-se ainda encontrar uma camada cultural no conteúdo da fala porque ela comenta sobre o debate na sociedade civil sobre a criminalização da homofobia.

Todos esses exemplos em seus contextos e suas linguagens são interpretados instantaneamente por um humano. Uma máquina, porém, precisa de técnicas que consigam extrair essas informações numa linguagem que seja capaz de ser entendida e com a qual se possa tomar decisões. A Inteligência Artificial tenta fazer essa mediação e as técnicas que serão usadas neste projeto serão apresentadas na seção seguinte.

2.3 Processamento de Linguagem Natural (PLN)

O PLN é uma vertente de aplicações da Inteigência Artifical (IA) que se dedica a compreender a linguagem humana, tanto escrita quanto falada. Os principais objetivos pelos quais se deseja criar técnicas que deem essas habilidades a computadores são dois: possibilitar a comunicação entre humanos e máquinas e adquirir informações de fontes produzidas por humanos (RUSSELL; NORVIG, 2010). Este projeto se encontra no escopo do segundo objetivo, como um problema de classificação de texto. Dado um texto, o programa deve decidir a qual classe ele pertence a partir de um conjunto pré-determinado de classes (RUSSELL; NORVIG, 2010).

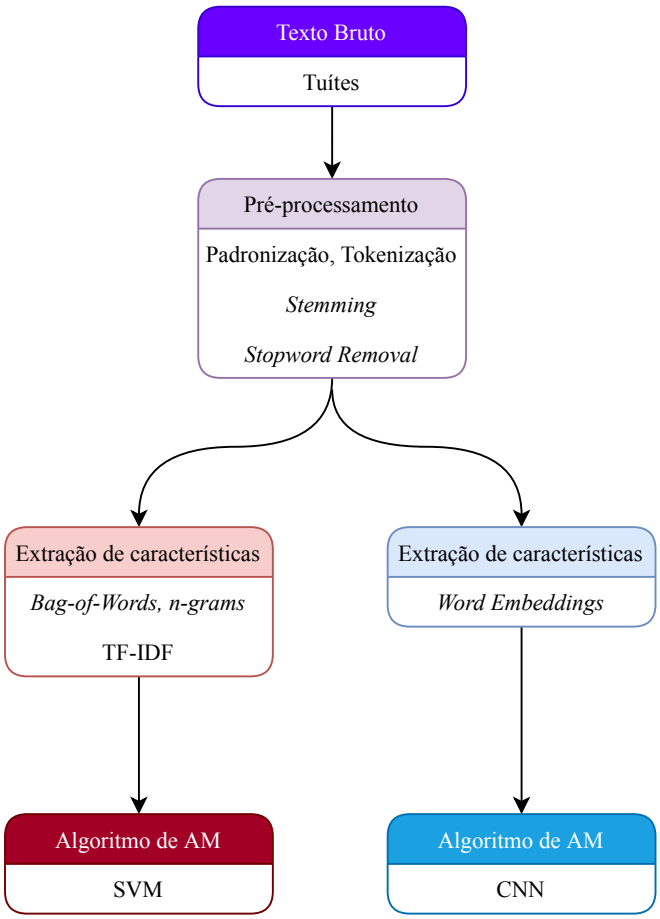
Inicialmente, numa pesquisa de PLN, define-se o corpus a ser estudado. Corpus é o conjunto de documentos analisados e cada documento é composto por um texto. Desde o texto bruto extraído de uma fonte até um nível mais alto de abstração da língua, o texto passa por algumas etapas de processamento e existem diversas técnicas aplicadas. Neste

[1] O símbolo asterisco (*) indica que a palavra ou frase que o segue pretende ser uma correção a algo escrito anteriormente.

projeto, serão aplicadas duas técnicas de Aprendizado de Máquina (AM) para tratar o problema de classificação de discurso de ódio em tuítes e elas exigem que o texto passe por processamentos diferentes.

A etapa de pré-processamento é comum tanto ao SVM quanto às CNN. O texto obtido precisa ser modelado numa linguagem inteligível ao algoritmo de aprendizado, numa etapa chamada de extração de características. No caso do SVM podem ser usadas as *Bag-of-Words* (BOW), *n-grams* ou *char n-grams*, por exemplo. Uma outra técnica pode ser aplicada em conjunto com a anterior, a *Term Frequency - Inverse Document Frequency* (TF-IDF). O produto gerado é então alimentado à SVM. No caso das CNN, a etapa de extração de características envolve técnicas de *Word Embedding*. São métodos que tentam representar palavras numericamente em um espaço vetorial a partir de suas semelhanças e usos em comum com outras palavras. A Figura 1 ilustra o caminho que o dado percorre.

Figura 1 – Etapas do processamento do texto fonte



Fonte: produzido pelo próprio autor.

Nas seções seguintes, cada uma dessas etapas e as técnicas usadas serão explicadas em mais detalhes.

2.3.1 Pré-processamento

A primeira etapa consiste no pré-processamento do texto, transformando-o em sequências bem definidas de unidades linguisticamente significativas (INDURKHYA; DAMERAU, 2010). O texto passa por processos chamados de padronização, filtragem e segmentação. A padronização consiste em deixar todas as letras minúsculas e a filtragem em retirar sinais de pontuação e caracteres especiais. Já a segmentação de texto é a identificação de cada palavra ou frase que constitui o texto. A segmentação de palavras é mais comumente chamada de tokenização e é a mais usada no pré-processamento de tuítes, já que não costumam ser constituídos de muitas frases por estarem limitados a 280 caracteres.

Outras técnicas que podem ser usadas no pré-processamento são a remoção de *stopwords* e o *stemming*. A remoção de *stopwords* retira palavras que, apesar de serem muito frequentes, não adicionam muito valor semântico ao texto, como artigos, preposições e alguns pronomes, podendo variar de aplicação para aplicação. O *stemming* elimina as desinências das palavras, deixando apenas o radical, ou seja, a raiz semântica da palavra. Porém, pode ser que termo restante não seja uma palavra válida da língua.

Para exemplificar essas técnicas de pré-processamento, considere o tuíte d_1: "Essa dupla é uma piada. Se não fosse essa, seria a gorda e o pastor trav*co."[2] Após o pré-processamento com padronização, filtragem, tokenização, remoção de artigos e *stemming*, o vetor de termos (ou *tokens*) obtidos seria: [dupla], [piada], [gord], [pastor], [trav*co]. Um outro exemplo é o tuíte d_2: "Gente, hoje eu não tô bem. Hoje o dia tá incrível!" Após o pré-processamento: [gente], [hoje], [bem], [hoje], [dia], [incrível]. E o tuíte d_3, "Bandidagem X*ng-l*ng não é de hoje. Essa gente não tem escrúpulos" se torna [bandid], [x*ng-l*ng], [hoje], [gente], [escrúpulo]. Cada tuíte, então, termina o pré-processamento como um vetor constituído dos *tokens* extraídos do texto bruto e pronto para a próxima etapa.

2.3.2 Modelagem e extração de características

2.3.2.1 *Bag-of-words*, *n-grams* e TF-IDF

Após o pré-processamento, tem-se vetores de *tokens* de cada documento, nesse caso, tuítes, e é necessário escolher uma forma de representar os documentos numericamente para que se possa utilizá-los nos algoritmos de AM. Duas técnicas são mais comuns: *bag-of-words* (BOW) e *n-grams*, e podem ser usadas sozinhas ou em conjunto.

[2] As palavras ofensivas estão censuradas pelo autor com um asterisco (*).

Na técnica BOW, uma matriz chamada de matriz documento-termo é gerada. Nela, cada linha representa um documento e cada coluna apresenta um termo presente em todo o corpus. Cada posição da matriz mostra a quantidade de vezes que cada termo aparece em cada documento do corpus estudado. Considere os tuítes exemplos da seção 2.3.1. A Tabela 1 mostra o resultado que seria obtido caso esse corpus fosse representado por BOW.

Tabela 1 – Matriz documento-termo para representação em BOW dos tuítes exemplos

Documentos	Termos												
	bem	bandid	dia	dupla	escrúpulo	gente	gord	hoje	incrível	pastor	piada	trav*co	x*ng-l*ng
d_1	0	0	0	1	0	0	1	0	0	1	1	1	0
d_2	1	0	1	0	0	1	0	2	1	0	0	0	0
d_3	0	1	0	0	1	1	0	1	0	0	0	0	1

Fonte: produzido pelo próprio autor.

Uma desvantagem da BOW é o aumento da sua dimensionalidade quanto maior for o corpus modelado. Outra característica é a esparsidade da matriz gerada, já que muitas palavras não se repetem de um tuíte para outro, o que prejudica o desempenho do algoritmo (CAMELO, 2017).

A técnica *n-grams* leva em consideração a sequência e proximidade das palavras num texto. A técnica consiste em formar sequências de n termos para cada termo do documento. Um exemplo para $n = 3$, ou *trigram*, de d_1 é o vetor [dupla piada gord], [piada gord pasrtor], [gord pastor trav*co]. Uma matriz documento-termo também é gerada, mas nesse caso, cada trio de palavras é considerado um termo.

Com a obtenção da matriz documento-termo, uma segunda técnica pode ser aplicada, a *Term Frequency-Inverse Document Frequency* (TF-IDF). Badjatiya e outros (2017) testaram diferentes formas de extração de características para identificação de discurso de ódio em tuítes e obteve melhores resultados quando TF-IDF era utilizado.

TF-IDF é um método utilizado no campo de Recuperação de Informação e interpretado na Teoria da Informação como uma forma de encontrar a quantidade de informação contida em cada termo de um corpus e graduá-la com a probabilidade de ocorrência desse termo (AIZAWA, 2003). Essa quantificação é estimada a partir da frequência de um termo em um documento e da quantidade de documentos do corpus que apresentam esse termo. Sua formulação é detalhada na equação (1).

$$w_{d_i,t_j} = tf_{d_i,t_j} \cdot \log \frac{N}{df_{t_j}} \tag{1}$$

Na matriz documento-termo, cada célula recebe o peso w_{d_i,t_j}, em que d_i indica o documento e t_j indica o termo. Esse peso é dado pelo produto entre a frequência do termo (TF,

do inglês *term frequency*) e o inverso da frequência nos documentos (IDF, do inglês *inverse document frequency*). O TF é a quantidade de vezes que o termo t_j aparece no documento d_i, denotado na equação (1) por tf_{d_i,t_j}. O IDF é o logaritmo da razão entre o número de documentos do corpus (N) e o *document frequency* (df_{t_j}), que é a quantidade de documentos em que o termo t_j aparece (RUSSELL; NORVIG, 2010, p. 459). De forma geral, quanto mais um termo aparece num documento e quanto menos documentos apresentam esse termo, mais relevante ele é.

Foi calculado o TF-IDF para a Tabela 1 e o resultado pode ser visto na Tabela 2.

Tabela 2 – Matriz documento-termo para representação em BOW após calculado o TF-IDF

Documentos	Termos												
	bem	bandid	dia	dupla	escrúpulo	gente	gord	hoje	incrível	pastor	piada	trav*co	x*ng-l*ng
d_1	0	0	0	0,48	0	0	0,48	0	0	0,48	0,48	0,48	0
d_2	0,48	0	0,48	0	0	0,18	0	0,35	0,48	0	0	0	0
d_3	0	0,48	0	0	0,48	0,18	0	0,18	0	0	0	0	0,48

Fonte: produzido pelo próprio autor.

Comparando as Tabelas 1 e 2, pode-se perceber que as células que tinham valor igual zero permaneceram iguais a zero, afinal o tf_{d_i,t_j} era zero. O interessante é notar que termos que aparecem em mais de um documento recebem pesos menores com o TF-IDF. Isso indica que suas presenças nos documentos têm menos relevância para a extração de significado e, portanto, para a classificação se um discurso é de ódio ou não.

O vetor formado pelo cálculo de TF-IDF de cada documento é normalizado de acordo com a equação (2) (RUSSELL; NORVIG, 2010, p. 462) para então ser aplicado no algoritmo de AM que fará a classificação. No caso deste projeto, será utilizado o SVM.

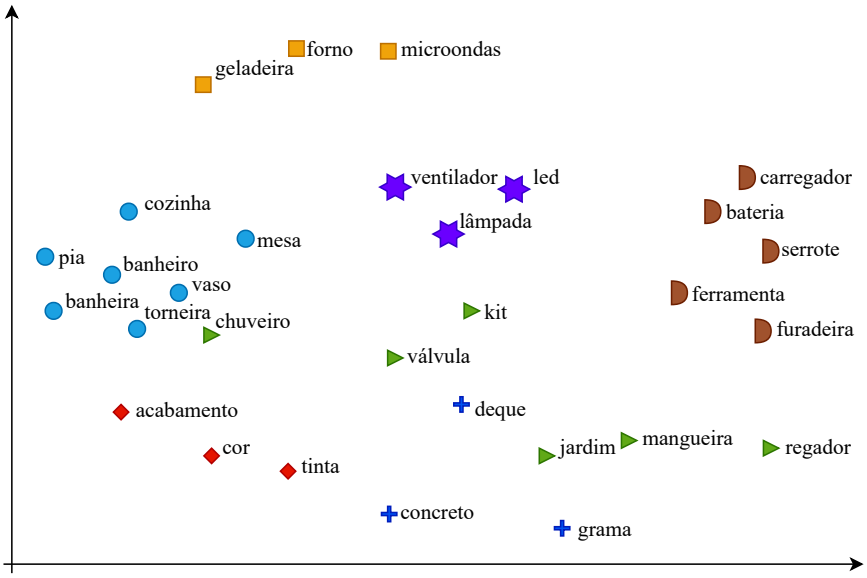
$$w_{d_i,t_j} = \frac{tf_{d_i,t_j} \cdot idf_{t_j}}{\sqrt{\sum_{k=1}^{v} (tf_{d_i,t_k} \cdot idf_{t_k})^2}} \tag{2}$$

2.3.2.2 *Word embeddings*

Em algoritmos que contêm arquiteturas de CNN são usadas outras técnicas para extração de características. As *word embeddings* são um conjunto de técnicas que conseguem representar palavras num espaço vetorial com base em suas semelhanças semânticas. O processo de formação desses modelos consiste em analisar sequências de BOW para compreender quais palavras são usadas em contextos e significados parecidos. Dessa forma, cada palavra tem um vetor associado e palavras com significado semelhante, ou que estão relacionadas ao

mesmo tema, tendem a ter seus vetores mais próximos uns dos outros. A Figura 2 ilustra esse fenômeno. A palavra “banheiro”, indicada em ciano, está mais próxima de “vaso sanitário” que de “furadeira”, por exemplo.

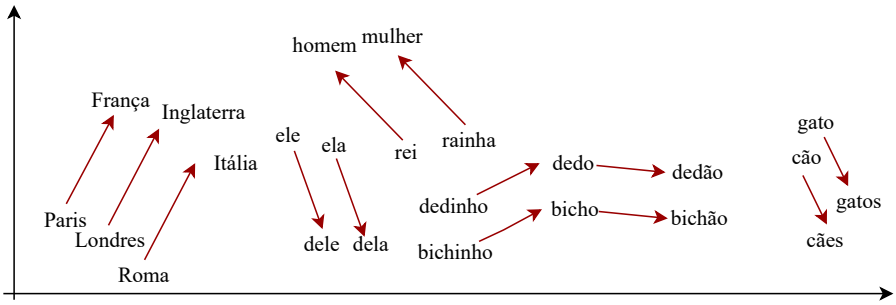
Figura 2 – Visualização em duas dimensões de *word embeddings* de termos associados ao lar



Fonte: Lynn (c2020).
Nota: traduzido e adaptado pelo autor.

Outra propriedade das *word embeddings* é a capacidade de realizar operações vetoriais sobre conceitos semânticos e morfológicos das palavras. Gato está para gatos assim como cão está para cães, como pode ser visto na Figura 3. Ou seja, é como se o plural se tornasse um vetor que pode ser usado em operações com palavras. Outro exemplo é a relação entre os pares “Paris” e “França”, “Londres” e “Inglaterra” e “Roma” e “Itália”. As capitais estão agrupadas juntas, assim como os países, e estão relacionadas pelo vetor que representa a ideia “é capital de”.

Figura 3 – Visualização em duas dimensões das relações vetoriais entre palavras num espaço criado por *word embeddings*



Fonte: Pal (2019).
Nota: traduzido e adaptado pelo autor.

Para compor frases, o vetor de cada palavra é concatenado de maneira que o resultado

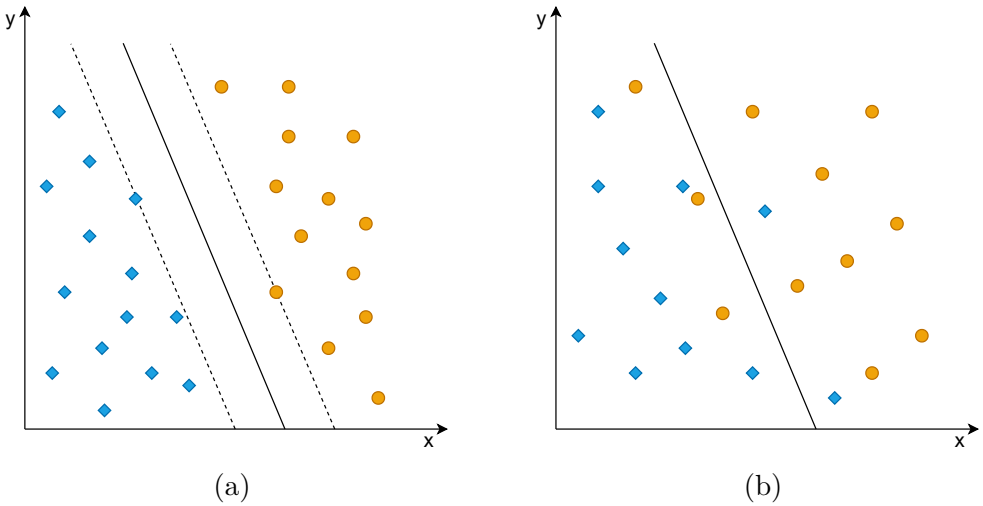
é uma matriz, que é interpretada como uma imagem pelas CNN. Essa matriz é então fornecida como entrada das CNN, que serão tratadas em mais detalhes na seção 2.3.4.

2.3.3 Máquinas de vetores suporte (SVM)

Máquinas de vetores suporte (SVM, do inglês *Support vector machines*) são uma técnica supervisionada de AM que é utilizada em classificações binárias. Aprendizado supervisionado quer dizer que a máquina passa por um processo em que é ensinada a realizar uma tarefa na qual já é conhecido o resultado esperado. No caso da detecção de discurso de ódio, na fase de treinamento, um subconjunto dos dados, já rotulados, é fornecido à máquina para que ela aprenda a distinguir as duas classes de texto: ódio e neutro, por exemplo. Então, numa segunda etapa, dados não rotulados são fornecidos para que a máquina faça a classificação a partir do modelo de decisão criado na fase de treinamento.

No caso do SVM, a classificação é feita dividindo-se o espaço vetorial em dois. O algoritmo define o hiperplano que melhor separa as duas classes do conjunto de dados de treinamento. Considere o exemplo na Figura 4(a). Os círculos em amarelo simbolizam tuítes que continham discurso de ódio e os quadrados em azul, aqueles que continham conteúdo neutro. As retas tracejadas contém, cada uma, o ponto mais próximo da reta, chamado de vetor suporte. A área entre essas retas é chamada de margem e a reta que melhor separa os dois conjuntos de dados é aquela equidistante às retas de suporte. No centro do gráfico, em preto, encontra-se essa reta para o exemplo.

Figura 4 – Exemplo de visualização em duas dimensões do funcionamento do algoritmo SVM (a) reta criada pelo SVM na etapa de treinamento (b) classificação realizada pelo SVM sobre o conjunto de teste



Fonte: produzido pelo próprio autor.

Finalizado o treinamento, o algoritmo pode ser testado quanto à sua efetividade na tarefa

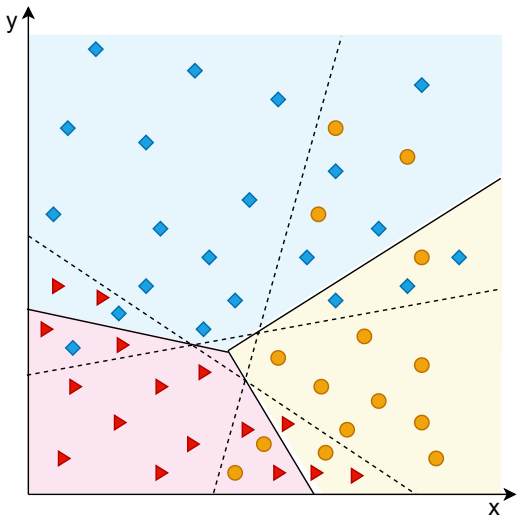
para a qual foi treinada. O conjunto de dados de teste então é projetado no espaço vetorial do problema e a classificação de cada documento é determinada pelo lado do hiperplano onde foi projetado.

Prosseguindo com o exemplo, o conjunto de teste é classificado, como mostra a Figura 4(b), pela reta gerada na etapa de treinamento. Nota-se que alguns círculos em amarelo e alguns quadrados em azul se encontram "do lado errado" da reta. Isso pode acontecer porque a reta foi calculada a partir dos dados de treinamento, um subconjunto diferente dos dados.

Neste exemplo, o classificador parece ter conseguido uma alta taxa de sucesso já que poucos pontos foram classificados de forma incorreta. As métricas usadas para avaliar o desempenho de algoritmos de classificação que serão usadas neste projeto serão tratadas em detalhes na seção 2.3.5.

Tradicionalmente, O SVM é aplicado em problemas com 2 classes de dados. Entretanto, são comuns problemas com 3 ou mais classes. Dessa forma, foram desenvolvidas estratégias para solucioná-los. A abordagem que será empregada neste trabalho é a "um contra todos". Ela define que num problema com n classes, n classificadores são necessários e cada um deles é responsável por distinguir uma determinada classe de todas as outras. A partir das n retas obtidas pelo algoritmo, são definidas as regiões de classificação de cada classe. O exemplo da Figura 5 ilustra um problema com 3 classes. Cada reta tracejada separa um conjunto de dados dos outros dois. As retas contínuas definem as fronteiras que delimitam as regiões de classificação para cada conjunto dos dados. Novamente, alguns pontos podem ser erroneamente classificados.

Figura 5 – Exemplo de visualização em duas dimensões da classificação realizada pelo SVM sobre o conjunto de dados com 3 classes e abordagem "um contra todos".

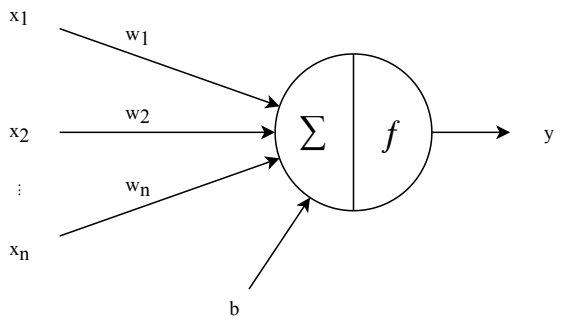


Fonte: produzido pelo próprio autor.

2.3.4 Redes neurais convolucionais (CNN)

Redes neurais artificiais são algoritmos de AM que são inspirados no funcionamento do cérebro humano. Têm como unidade básica o neurônio, ilustrado na Figura 6.

Figura 6 – Modelo matemático de um neurônio



Fonte: produzido pelo próprio autor.

O neurônio é uma abstração para uma operação matemática. Nele, cada vetor de entrada $\mathbf{x} = [x_1,\ x_2,\ \ldots,\ x_n]^T$ multiplica o vetor de pesos $\mathbf{w} = [w_1,\ w_2,\ \ldots,\ w_n]^T$. O produto é somado ao valor de *bias* b. A equação (3) descreve a operação.

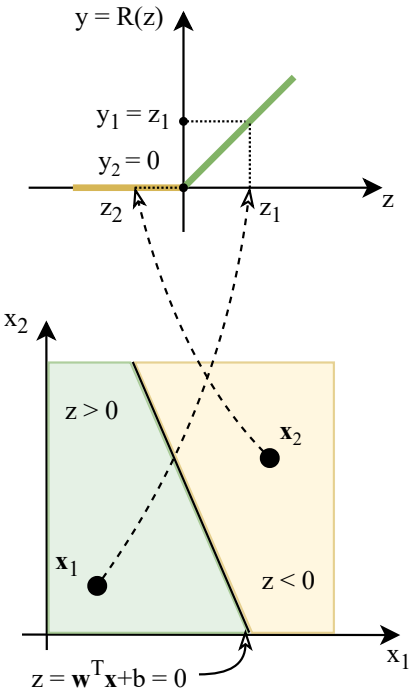
$$z = \mathbf{w}^T\mathbf{x} + b \tag{3}$$

Da geometria analítica, sabe-se que essa é a equação de um hiperplano. Os parâmetros $\mathbf{w}$ e b descreveriam, em duas dimensões, uma reta e, em três dimensões, um plano. Os valores de entrada $\mathbf{x}$, ao serem projetados no espaço, ficam de um lado ou de outro da função. Nota-se que o princípio de funcionamento é semelhante ao do SVM. O valor z é aplicado à chamada função de ativação, que pode ser de vários tipos. A unidade de retificação linear (ReLU, do inglês *rectified linear unit*) é comumente usada em CNN e é definida como na equação (4).

$$R(z) = \begin{cases} 0 & se\ z < 0 \\ z & se\ z \geq 0 \end{cases} \tag{4}$$

A Figura 7 ilustra o cálculo feito por um neurônio com dois parâmetros de entrada x_1 e x_2. Os dois parâmetros são coordenadas dos pontos $\mathbf{x}_1$ e $\mathbf{x}_2$, que são dois diferentes vetores de entrada.

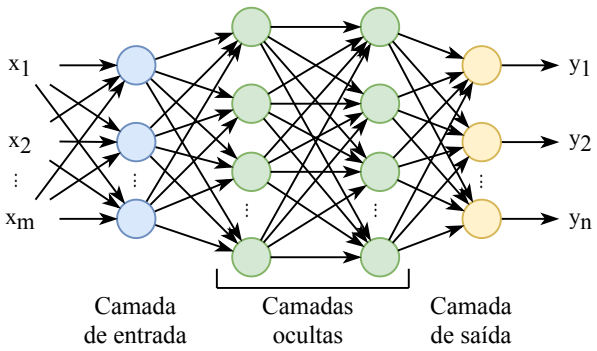
Figura 7 – Ilustração da operação realizada pelo neurônio



Fonte: produzido pelo próprio autor.

Até aqui foi mostrado o funcionamento de um neurônio isolado. Uma rede neural é composta de vários neurônios dispostos em camadas. O vetor de saída **y** de uma camada serve como entrada da camada seguinte, até que se chegue na camada de saída. A Figura 8 mostra uma rede com camadas de entrada, saída e camadas intermediárias, chamadas de camadas ocultas.

Figura 8 – Modelo de rede neural



Fonte: produzido pelo próprio autor.

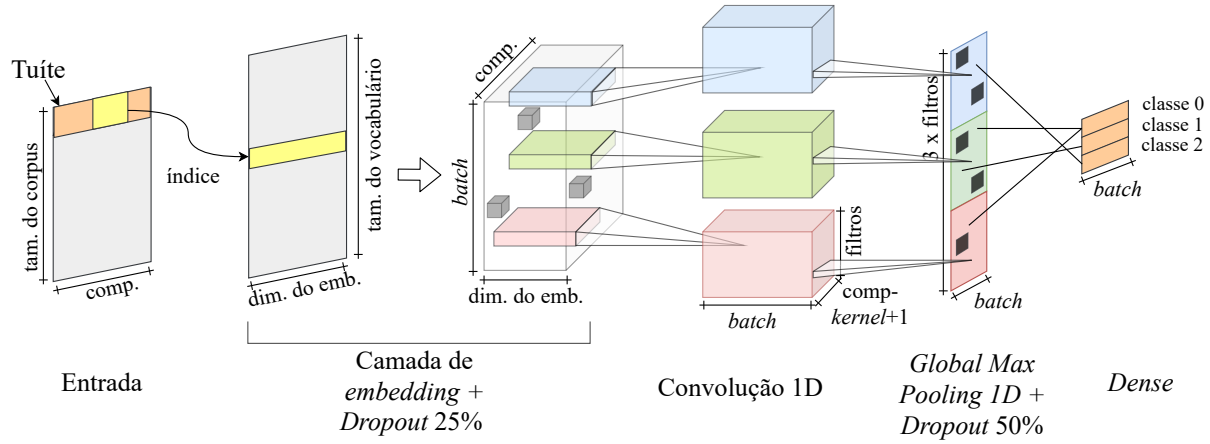
As redes neurais podem ter aprendizado supervisionado e ele se dá alterando os valores dos pesos **w** que transportam a saída de uma camada para a entrada da outra. Quando o aprendizado é supervisionado, um conjunto de dados rotulados é fornecido como entrada

da rede e os pesos são modificados até que a saída da rede corresponda ao rótulo real dos dados.

Um dos tipos de redes neurais são as CNN *do inglês Convolutional Neural Networks* as quais fazem parte do grupo das Redes Neurais Profundas, que têm esse nome por possuírem muitas camadas ocultas em sua arquitetura. Outra diferença é que as CNN recebem matrizes como entrada ao invés de vetores, como a rede da Figura 8.

A arquitetura da rede utilizada no experimento foi elaborada por Badjatiya e outros (2017) e está ilustrada na Figura 9. Ela foi escolhida pois nesse artigo ela apresentou *F-measure* acima de 0,8 na classificação de linguagem ofensiva e discurso de ódio de cunho sexista e racista.

Figura 9 – Arquitetura de CNN reproduzida no projeto



Fonte: produzido pelo próprio autor.

Cada bloco na Figura 9 é uma camada da rede, e sob cada uma está descrita a função da camada e as dimensões do resultado da operação realizada por ela. Essas funções também aparecem em outras arquiteturas de rede e são explicadas em detalhes a seguir.

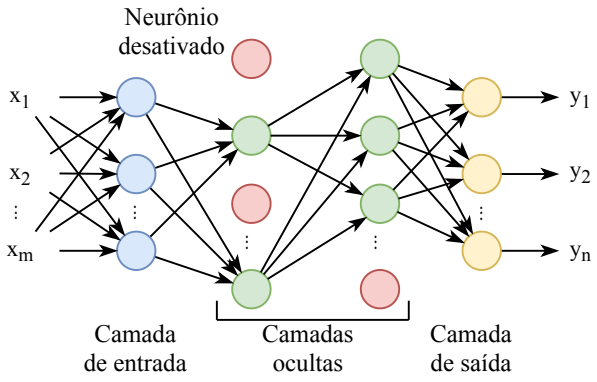
- Entrada: para a entrada da rede, forma-se uma matriz em que cada linha representa um documento do corpus. A quantidade de colunas vai ser o comprimento do maior documento. Cada *token* é substituído pelo seu índice no dicionário de *tokens* fornecido pelo modelo de *embeddings* aplicado. Quando um *token* não é encontrado no dicionário, ele recebe o rótulo designado de "fora de vocabulário". Documentos menores que o maior deles são completados com zero.

- Camada de *embedding*: quando se utiliza *word embeddings* para representação das palavras, a primeira camada da rede deve ser a de *embedding*. Ela é responsável por substituir cada *token* presente no corpus por sua respectiva representação vetorial. Ela usa os índices presentes na entrada para encontrar no dicionário sua

representação vetorial. Feita a substituição, a rede trabalha com volumes menores de dados, dividindo-os em *batches* (normalmente traduzidos como lotes). Essa divisão serve para diminuir a quantidade de memória necessária para o treinamento da rede. Entretanto, quanto menores forem, mais *batches* terão que passar pela rede, assim, há um *trade-off* que deve ser levado em consideração. O resultado da camada é o bloco que tem como dimensões o comprimento máximo do documento, o tamanho do *batch* e a dimensão do vetor de *embedding* aplicado.

- *Dropout*: é uma função que durante a fase de treinamento, a cada nova entrada, um conjunto aleatório de neurônios é desconectado da rede, de forma que eles não recebem entradas nem geram saídas (Figura 10). As porcentagens indicadas na Figura 9 (25% e 50%) definem a quantidade de neurônios que serão desligados. Isso é feito para evitar que a rede se especialize demais (processo chamado de *overfitting*), tornando-a capaz de reconhecer apenas os dados usados no treinamento.

Figura 10 – Ilustração de uma rede com neurônios desconectados durante o processo de *dropout*

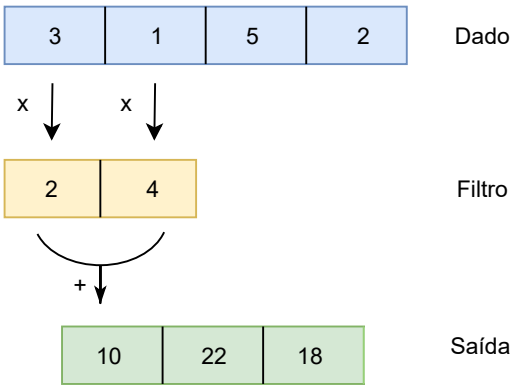


Fonte: produzido pelo próprio autor.

- Convolução 1D: a convolução é uma operação que serve para encontrar padrões nos dados de entrada. Como o tipo de dado estudado neste trabalho é o texto, se utiliza a convolução 1D, pois o texto é um dado unidimensional. Diferentes filtros são usados para encontrar esses padrões, chamados de *feature maps* (mapas de características, em português). A Figura 11 mostra um exemplo de filtro e o resultado obtido por ele. Na operação de convolução, cada peso do filtro multiplica os valores de uma janela de mesmo tamanho no vetor de dados. Os produtos são somados e compõem o vetor resultado.

  No caso deste trabalho, em que cada palavra é representada por um vetor, a operação fica um pouco diferente, como ilustrado na Figura 12. Os filtros também passam a ter como segunda dimensão o comprimento do vetor de *embedding*. Nota-se a partir das Figuras 11 e 12 que o comprimento da saída é menor, dado pelo comprimento do dado de entrada menos o comprimento do filtro mais 1.

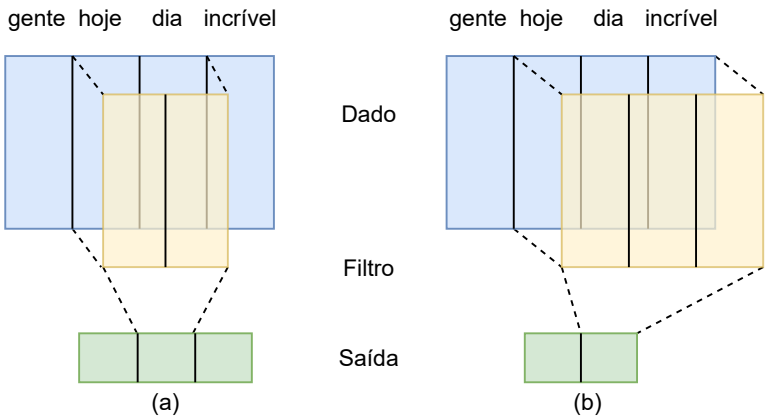
Figura 11 – Exemplo de filtro e resultado obtido por ele na etapa de convolução



Fonte: produzido pelo próprio autor.

Na arquitetura da Figura 9 há 3 camadas de convolução. Cada uma delas tem filtros de comprimentos diferentes. Na proposição dos autores, esses filtros teriam comprimentos iguais a 3, 4 e 5. Além disso, são implementados 100 filtros por camada. Numa convolução 1D, os filtros, em seus diferentes comprimentos, agem como *n-grams*, pois agrupam termos adjacentes ao serem filtrados.

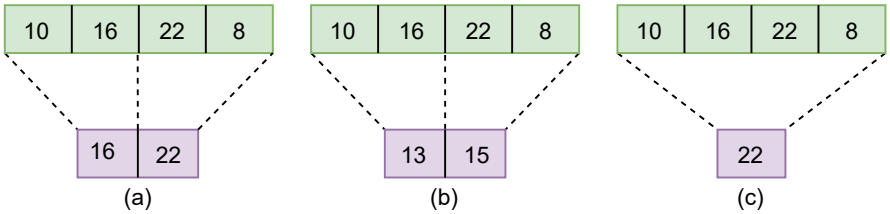
Figura 12 – Exemplo de filtro que opera sobre *embeddings* e resultado obtido por ele na etapa de convolução (a) filtro com comprimento igual a 2 (b) filtro com comprimento igual a 3



Fonte: produzido pelo próprio autor.

- *Pooling*: é uma etapa que reduz a dimensão dos dados para facilitar o cômputo e eliminar possíveis ruídos. Pode ser feito extraindo-se o maior valor (*max pooling*) ou a média (*average pooling*) dos valores em uma janela da matriz. Pode ainda capturar apenas o maior valor de toda a matriz, procedimento chamado de *global max pooling*. A Figura 13 ilustra as diferentes operações de *pooling* num vetor exemplo. Na rede da Figura 9, os resultados da operação de *pooling* para cada camada de convoulção são concatenados. Em seguida, passam por uma nova camada de *dropout*, dessa vez com taxa de 50%.

Figura 13 – Ilustração das diferentes operações de *pooling* (a) *max pooling* numa janela de tamanho 2 (b) *average pooling* numa janela de tamanho 2 (c) *global max pooling*



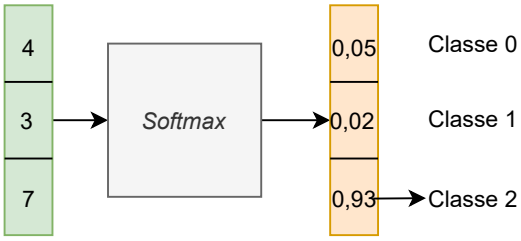
Fonte: produzido pelo próprio autor.

- *Dense*: como mostra a Figura 9, após a camada de *global max pooling*, o vetor ainda é muito maior que a quantidade de categorias na camada de saída. Então essas camadas finais, em que todos os neurônios de uma camada para a outra estão conectados, servem para fazer a redução final do número de neurônios até a camada de saída. A função de ativação nessa camada é a *soft max*. Ela ativa a saída da classe que foi avaliada com maior probabilidade de representar o dado de entrada. Ela é descrita pela equação (5), em que p_i é a pontuação dada para cada classe, z_i é a entrada para a função de ativação e C é o número de classes.

$$s_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{5}$$

A Figura 14 ilustra o processo de classificação final a partir da função *softmax*.

Figura 14 – Ilustração da atuação da função *softmax* na classificação



Fonte: produzido pelo próprio autor.

No processo de treinamento de uma rede, tenta-se minimizar o resultado de uma equação chamada de função de perda. Ela é responsável por comparar a classificação real do dado com a classificação que está sendo obtida pela rede. Assim, a rede é treinada até que se obtenha um valor que indique que a rede aprendeu a generalizar o problema a partir dos dados de treinamento. Num problema com 3 ou mais classes pode ser utilizada a função *categorical crossentropy* (CCE), que calcula a perda a partir da saída obtida pela rede. Para isso, os rótulos de cada documento devem estar representados na forma de um vetor de tamanho C com o valor 1 indicando a qual classe aquele dado pertence. A função CCE

está descrita na equação (6), em que $\mathbf{t}$ é o vetor com o rótulo de cada documento e $\mathbf{s}$ é o vetor com as pontuações obtidas na camada de saída.

$$CCE = -\mathbf{t} \cdot log(\mathbf{s}) \tag{6}$$

Essa equação indica que quanto menor a pontuação dada à classe correta, maior será o erro atribuído a essa classificação.

2.3.5 Métricas de avaliação

Dado um conjunto de documentos usados num teste de detecção de discurso de ódio, por exemplo, parte dele contém de fato discurso de ódio e a outra parte não. Diz-se que essa é a condição real do dado. Após ser processado pelo algoritmo, alguns deles terão sido classificados como discurso de ódio e os outros não. Nesse caso, tem-se a condição predita para os textos. Os algoritmos, no entanto, não são perfeitos e nem sempre a classificação é feita corretamente. Em outras palavras, nem sempre a condição predita de um dado corresponde à sua condição real. Essa discordância é representada na chamada matriz de confusão e pode ser vista no Quadro 1.

Quadro 1 – Matriz de confusão teórica

Condição real	Condição predita	
	Positivo	Negativo
Positivo	VP	FN
Negativo	FP	VN

Fonte: produzido pelo próprio autor.

Quando a predição coincide com a condição real, é dito que há um verdadeiro positivo (VP) ou um verdadeiro negativo (VN). Quando a condição real não coincide com a predição, há novamente dois casos: os falsos-positivos (FP), em que a ausência da condição foi erroneamente classificada como presença, e os falsos-negativos (FN), em que a presença da condição foi erroneamente classificada como ausência. Voltando ao exemplo na seção 2.3.3, vê-se na Figura 4(b) que alguns pontos foram classificados erroneamente. A matriz de confusão para esse caso se encontra no Quadro 2.

Para mensurar a qualidade de um algoritmo de classificação são usadas algumas métricas estatísticas calculadas a partir dos valores de VP, VN, FP e FN. São elas: a precisão, o *recall* e a *F-measure*. A precisão objetiva responder a pergunta: "quantas das classificações

Quadro 2 – Matriz de confusão com os valores do exemplo da Figura 4(b)

Condição real	Condição predita	
	Positivo	Negativo
Positivo	8	3
Negativo	2	9

Fonte: produzido pelo próprio autor.

positivas estão corretas?". Matematicamente, ela pode ser descrita pela equação (7).

$$precisão = \frac{VP}{VP + FP} \times 100\% \tag{7}$$

O *recall*, por sua vez, objetiva responder a pergunta: "quantos dos dados positivos foram classificados corretamente?". A equação (8) define sua formulação.

$$recall = \frac{VP}{VP + FN} \times 100\% \tag{8}$$

A *F-measure* é a média harmônica entre a precisão e o *recall* e pode ser dada pela equação (9).

$$F{-}measure = \frac{2 \cdot precisão \cdot recall}{precisão + recall} \times 100\% \tag{9}$$

Para o exemplo da Figura 4(b), então, $precisão = 80,0\%$, $recall = 72,73\%$ e $F{-}measure = 76,19\%$.

Em problemas com mais de 2 classes de dados, tem-se que analisar cada classe de interesse por vez. Ou seja, para calcular a precisão e o *recall*, toma-se uma classe como positiva e as outras como negativas. O Quadro 3 mostra um exemplo de matriz de confusão para um problema com 3 classes. Nele, a primeira classe é tomada como positiva.

Passa-se a calcular a precisão e o *recall* das formas mostradas nas equações (10) e (11). O *F-measure* continua sendo calculado da mesma forma que na equação (9).

$$precisão = \frac{VP}{VP + \sum FP} \times 100\% \tag{10}$$

Quadro 3 – Matriz de confusão para um conjunto de dados com 3 classes

Condição real	Condição predita		
	Positivo	Negativo	Negativo
Positivo	VP	FN	FN
Negativo	FP	VN	VN
Negativo	FP	VN	VN

Fonte: produzido pelo próprio autor.

$$recall = \frac{VP}{VP + \sum FN} \times 100\% \tag{11}$$

Os termos $\sum FP$ e $\sum FN$ denotam todos os valores de FP e FN nas mesmas coluna e linha de VP, respectivamente. Pode-se fazer uma matriz de confusão para o exemplo da Figura 5. O resultado é exibido no Quadro 4.

Quadro 4 – Matriz de confusão com os valores do exemplo da Figura 5

Condição real	Condição predita		
	Classe 1	Classe 2	Classe 3
Classe 1 (▷)	11	3	2
Classe 2 (○)	2	10	3
Classe 3 (◇)	1	3	16

Fonte: produzido pelo próprio autor.

Em problemas multiclasse, para calcular a precisão e o *recall* gerais, ou seja, do classificador como um todo, deve-se somar os elementos da diagonal principal e dividi-los pela soma de todos os valores da matriz, como definido na equação (12). $va_{i,j}$ denota o valor absoluto de cada célula da matriz.

$$precisão = recall = \frac{\sum_{i=1}^{n} va_{i,i}}{\sum_{i=1}^{n} \sum_{j=1}^{n} va_{i,j}} \tag{12}$$

Ambos os valores serão iguais, portanto, o *F-measure* também. Assim, o *F-measure* para o classificador exemplo da Figura 5 é 72,55%.

Para além de representar matrizes de confusão com o valor absoluto da classificação dos dados, uma outra forma de representá-las é normalizando-as. Isso pode ser feito de 3 formas: por meio das colunas, das linhas e da soma de todos os valores da matriz. No escopo deste trabalho é interessante dar atenção às duas primeiras, pois ao fazê-las, evidencia-se a precisão e o *recall* dos resultados. A normalização pelas colunas pode ser calculada pela

equação (13) e a normalização pelas linhas pode ser calculada pela equação (14). $vnc_{i,j}$ e $vnl_{i,j}$ denotam os valores normalizados pelas colunas e linhas, respectivamente, de cada célula da matriz.

$$vnc_{i,j} = \frac{va_{i,j}}{\sum_{i=1}^{n} va_{i,j}} \tag{13}$$

$$vnl_{i,j} = \frac{va_{i,j}}{\sum_{j=1}^{n} va_{i,j}} \tag{14}$$

A nova matriz normalizada pelas colunas apresenta, na diagonal principal, a precisão obtida pelo classificador para cada classe. As colunas devem ser analisadas de forma independente e os valores fora da diagonal principal mostram em que proporção os dados erroneamente classificados realmente pertencem às outras classes. O Quadro 5 mostra a matriz de confusão do Quadro 4 normalizada pelas colunas.

Quadro 5 – Matriz de confusão do Quadro 4 normalizada pelas colunas

Condição real	Condição predita		
	Classe 1	Classe 2	Classe 3
Classe 1 (▷)	0,79	0,19	0,10
Classe 2 (○)	0,14	0,62	0,14
Classe 3 (◇)	0,07	0,19	0,76

Fonte: produzido pelo próprio autor.

A partir do Quadro 5 tira-se então que as precisões para as classes 1, 2 e 3 são, respectivamente: 79,0%, 62,0% e 76,0%. Já a matriz normalizada pelas linhas apresenta, na diagonal principal, o *recall* obtido pelo classificador para cada classe. As linhas devem ser analisadas de forma independente e os valores fora da diagonal principal mostram qual proporção dos dados foi erroneamente classificada e para qual classe. O Quadro 6 mostra a matriz de confusão do Quadro 4 normalizada pelas linhas.

Quadro 6 – Matriz de confusão do Quadro 4 normalizada pelas linhas

Condição real	Condição predita		
	Classe 1	Classe 2	Classe 3
Classe 1 (▷)	0,69	0,19	0,12
Classe 2 (○)	0,13	0,67	0,20
Classe 3 (◇)	0,05	0,15	0,80

Fonte: produzido pelo próprio autor.

A partir do Quadro 6 tira-se então que o *recall* para as classes 1, 2 e 3 é, respectivamente: 69,0%, 67,0% e 80,0%.

3 METODOLOGIA E ETAPAS DE DESENVOLVIMENTO

A pesquisa proposta neste trabalho é um estudo de comparação entre dois algoritmos de Inteligência Artificial na detecção de discurso de ódio na rede social *Twitter*. Por isso, é de natureza aplicada por se tratar de um problema específico dentro da área da Inteligência Artificial e Processamento de Linguagem Natural.

Pelo sucesso que as CNN têm demonstrado na solução de problemas em diversas áreas da IA, levanta-se a hipótese de que apresentará resultados melhores que o SVM na detecção de discurso de ódio. Para verificação da hipótese, foi delineado um experimento que produz dados quantitativos baseados em métodos estatísticos. Com isso, o objetivo da pesquisa pode ser considerado como explicativo, pois pretende fornecer evidências que corroborem com a confirmação ou refutação da hipótese formulada.

Sendo assim, a pesquisa visa responder às seguintes perguntas: como as CNN aliadas a *Word Embeddings* se comparam ao SVM na identificação de discurso de ódio? As CNN apresentam resultados superiores ao SVM como esperado?

3.1 Base de Dados

A base de dados escolhida para realização do experimento foi elaborada por Davidson e outros (2017) para detecção de discurso de ódio e linguagem ofensiva em tuítes na língua inglesa. A coleta foi feita em 2016 a partir de um léxico de palavras ofensivas e a rotulação foi feita de forma colaborativa na plataforma *CrowFlower*, de forma que cada tuíte tivesse pelo menos 3 avaliações. As classes são: discurso de ódio (DO), linguagem ofensiva (LO) e neutro (NE). A Tabela 3 mostra a quantidade de tuítes em cada classe e a proporção que representa do total.

Tabela 3 – Quantidade e proporção de cada classe no corpus do banco de dados

Classe	Quantidade	Proporção
Discurso de ódio	1430	5,77%
Linguagem ofensiva	19190	77,43%
Neutro	4163	16,80%
Total	24783	100%

Fonte: Davidson e outros (2017).

Para o experimento, foi escrito um programa pelo próprio autor na linguagem *Python* v3.7 com os recursos da biblioteca NLTK (BIRD; KLEIN; LOPER, 2009), *Sci-kit Learn*

(PEDREGOSA *et al.*, 2011) e *Keras* (CHOLLET, 2020) e está disponível em repositório online (QUEIROZ, 2021). Cada técnica exige uma metodologia específica e possuem parâmetros diferentes a serem testados.

3.2 Metodologia para o SVM

O arquivo com o conteúdo dos tuítes possui algumas informações, das quais apenas o texto do tuíte em si e a classe mais votada têm relevância para este projeto. Cada documento do corpus passa por padronização, remoção de *stopwords* e tokenização. Além disso, são removidos nomes de usuários e sinais de retuíte (RT).

Para otimização dos resultados do SVM, alguns parâmetros são variados. No pré-processamento, é testado se o *stemming* produz ou não resultados melhores. O *stemmer* utilizado foi o *Porter Stemmer*, da biblioteca NLTK. Na extração de características, são testadas combinações de unigramas, bigramas e trigramas e é testado também o número mínimo de ocorrências de um termo para mantê-lo na composição da matriz documento-termo. Esse procedimento tem como objetivo verificar se palavras ou sequências que ocorrem poucas vezes por conterem erros ortográficos e expressões pouco usadas não acrescentam significado ao vocabulário. De posse da matriz, é calculado em seguida o TF-IDF sobre ela.

Na aplicação do SVM, os dados foram repartidos em 70% para treinamento com validação cruzada em 5 *folds* e 30% para teste. Houve certificação de que cada grupo continha a mesma proporção de amostras de cada classe. Nessa etapa foi variado o parâmetro C da função que executa o algoritmo. Ele representa quão ajustado aos dados de treinamento o hiperplano encontrado deve estar. Por fim, como as classes têm frequências muito distintas, configurou-se o parâmetro da função que atribui pesos às classes para equipará-las no cálculo dos hiperplanos. O valor dos pesos é inversamente proporcional à frequência de cada classe e é calculado automaticamente pela função descrita na equação (15). São eles: $P_{DO} = 5,78$, $P_{LO} = 0,43$ e $P_{NE} = 1,98$.

$$peso\, classe\, k = \frac{total\ de\ amostras}{núm.\ de\ classes \times núm.\ de\ amostras\ da\ classe\ k} \tag{15}$$

O Quadro 7 resume os valores testados no experimento.

Quadro 7 – Parâmetros e valores testados para otimização do SVM

Parâmetro	Valores testados
Stemming	Sim, não
n-gram	n = (1), (1, 2) e (1, 2, 3)
Frequência mínima do termo	1, 2 e 3
C	10^{-4} até 10^{4}, exponencialmente
Pesos	5,78, 0,43 e 1,98
Kernel	Linear
Proporção treino/teste	70%/30%
Validação cruzada	5 *folds*

Fonte: produzido pelo próprio autor.

3.3 Metodologia para a CNN

Para o caso da CNN, o pré-processamento é feito da mesma forma, com padronização, remoção de *stopwords* e tokenização. O *stemming* também é parâmetro de teste nesse experimento. Essa escolha foi feita baseada no fato de que quando removidas as desinências, as palavras deixam de ser reconhecidas pelo *embedding* pré-treinado, de forma que o vocabulário a ser analisado pela rede tende a reduzir. Também foi levado em consideração a frequência mínima de uma palavra no corpus.

A arquitetura implementada é a explicada na seção 2.3.4 e foi mantida em comparação com a referência com exceção dos comprimentos dos *kernels* utilizados. A camada de *embedding* é seguida de um *dropout* de 25%. Nessa fase do experimento é variada a capacidade de treinamento dos vetores de *embedding*. O modelo utilizado foi o *GloVe* treinado com 2 bilhões de tuítes e vocabulário de 1,2 milhões de palavras (PENNINGTON; SOCHER; MANNING, 2014). São fornecidos 4 modelos em que as dimensões dos vetores são 25, 50, 100 e 200 e é testado apenas o com 200 dimensões.

A camada de convolução é composta por 3 camadas com 100 filtros para cada camada e ativados pela função ReLu. Os filtros eram matrizes originalmente de dimensões 3x200, 4x200 e 5x200. Decidiu-se, todavia, por variar quantos termos são convoluídos conjuntamente para valores menores. Foi aferido que o comprimento médio e mediano dos tuítes é de 11 *tokens* e o tamanho máximo é de 34. Por isso, serão testados *kernels* de comprimento (1, 2, 3), (2, 3, 4) e (3, 4, 5). Cada valor dos ternos indica um comprimento para cada camada de convolução.

Uma operação de *global max pooling* seguida da concatenação do resultado reduz a dimensão da rede, que passa por mais um *dropout*, dessa vez de 50% e ativado com ReLu. Por fim, uma camada densamente conectada com 3 neurônios e ativada com *softmax* fornece os

resultados da classificação para cada *batch*. Como existem 3 classes, a função de perda minimizada foi a *categorical crossentropy*. O otimizador aplicado foi o *RMSprop* com taxa de aprendizado mantida no *default* de 0,001.

A divisão dos dados entre treino e teste foi feita mantendo a proporção usada no SVM e de forma que os mesmos documentos usados como teste em um experimento fosse usada em outro. Para a validação foi respeitada a proporção de 20% da validação cruzada. Entretanto, a rede não faz validação cruzada, ela avalia o resultado para o conjunto de validação ao final de cada época. Foi garantido que os mesmos dados usados para treinamento e teste com o SVM também foram usados para treinamento e teste com a CNN. Os testes foram feitos por 10 épocas, pois notou-se que era suficiente para avaliar a convergência da função de perda. Um resumo dos dados de teste se encontra no Quadro 8.

Quadro 8 – Parâmetros e valores testados para otimização da classificação pela rede

Parâmetro	Valores testados
Stemming	Sim, não
Freqência mínima do termo	1, 2 e 3
Dimensão do vetor de embedding	200
Permitir treinamento da camada de embedding	Sim, não
Kernels	n=(1, 2, 3), (2, 3, 4) e (3, 4, 5)
Pesos	5,78, 0,43 e 1,98
Batch	128
Função de perda	*categorical crossentropy*
Otimizador	rmsprop
Taxa de aprendizado	0,001
Proporção treino/teste	70%/30%
Validação	20%

Fonte: produzido pelo próprio autor.

Os resultados precisão, *recall* e *F-measure*, explicados na seção 2.3.5, da CNN e do SVM, são comparados em conjunto com a matriz de confusão para cada técnica, objetivando-se encontrar valores o mais próximo possível da unidade. Por fim, é feita também uma comparação dos dados obtidos com os dados de literatura semelhante na concepção da conclusão para avaliação da consistência dos dados e da qualidade do próprio método.

4 RESULTADOS E DISCUSSÃO

O primeiro resultado obtido é o do pré-processamento. O Quadro 9 mostra um exemplo de transformação do texto adaptado de um tuíte presente no banco de dados até estar pronto para a etapa de extração de características.

Quadro 9 – Resultados obtidos nas etapas de pré-processamento

Etapa do pré-procesamento	Resultado
Texto bruto	@g0rillaz0e: *Sitting on my windowsill, window open, bird tries to fly in. Smacked it idc* 😂 *#ProtectTheAnimals* @peta
Remoção de @'s, #'s, emojis, etc	*Sitting on my windowsill, window open, bird tries to fly in. Smacked it idc*
Padronização	*sitting on my windowsill window open bird tries to fly in smacked it idc*
Remoção de *stopwords*	*sitting windowsill window open bird tries fly smacked idc*
Stemming	*sit windowsill window open bird tri fli smack idc*

Fonte: produzido pelo próprio autor.
Nota: o tuíte pode ser traduzido como "Sentado no peitoril da minha janela, janela aberta, um pássaro tenta entrar. Dei um tapa nele, nem ligo. #ProtejaOsAnimais"

Nota-se a presença do código "😂" e do termo "*idc*". O primeiro é a representação *unicode* decimal do emoji chamado "rosto com lágrimas de alegria" e o segundo é uma expressão em inglês popular no uso online que significa "I don't care" (não me importo, em português). O corpus analisado está repleto desses dois tipos de *tokens*. Enquanto os emojis são facilmente removíveis, as expressões abreviadas acabam passando pelo pré-processamento e podem inserir ruídos na extração de características.

Como resultado do teste com o SVM, o Quadro 10 resume quais valores ocasionaram melhores *F-measures*.

O tamanho do vocabulário obtido foi de 109927 *tokens*, sendo 13787 unigramas e 96140 bigramas. As 4 palavras mais frequentes são: *b*tch* (11373 ocorrências), *ho** (4304), *like* (2866) e *f*ck* (2754). *B*tch* e *ho** são também as que ganham maior peso com o cálculo do TF-IDF. O alto número de aparições dessas palavras provavelmente se deve ao modo como o corpus foi coletado, a partir de um léxico de palavras ofensivas (DAVIDSON *et al.*, 2017). Como esperado, o *stemming* melhorou a classificação porque centraliza as palavras flexionadas em uma única representação. A presença de bigramas indica que palavras tendem a ser usadas de maneira combinada para compor significados ofensivos. O bigrama mais frequente no vocabulário foi [*this b*tch*], expressão usada para direcionar o xingamento a alguém.

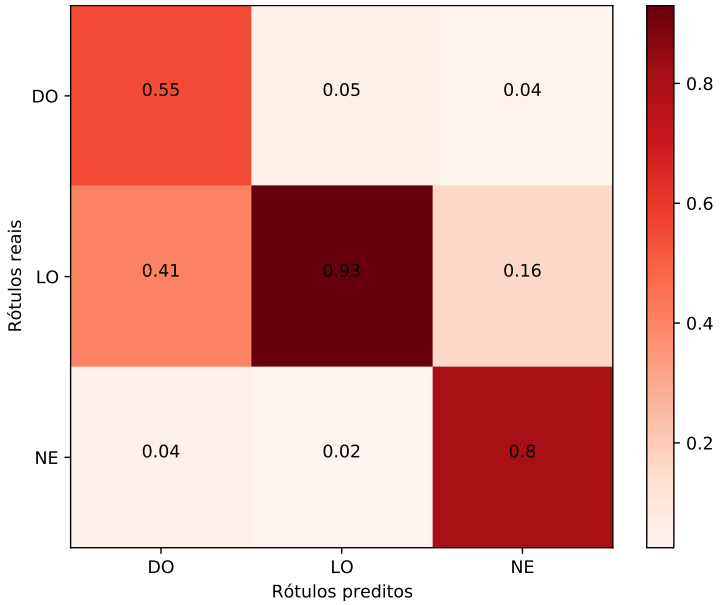
Quadro 10 – Parâmetros e valores testados para otimização do SVM

Parâmetro	Valores testados
Stemming	sim
n-gram	n = (1, 2)
Frequência mínima do termo	1
C	1
Pesos	5,78, 0,43 e 1,98
Kernel	linear
Proporção treino/teste	70%/30%
Validação cruzada	5 *folds*

Fonte: produzido pelo próprio autor.

O resultado da classificação feita pelo SVM pode ser visualizado na Figura 15, que exibe a matriz de confusão normalizada pelas colunas, ou seja, ela exibe a precisão das classificações.

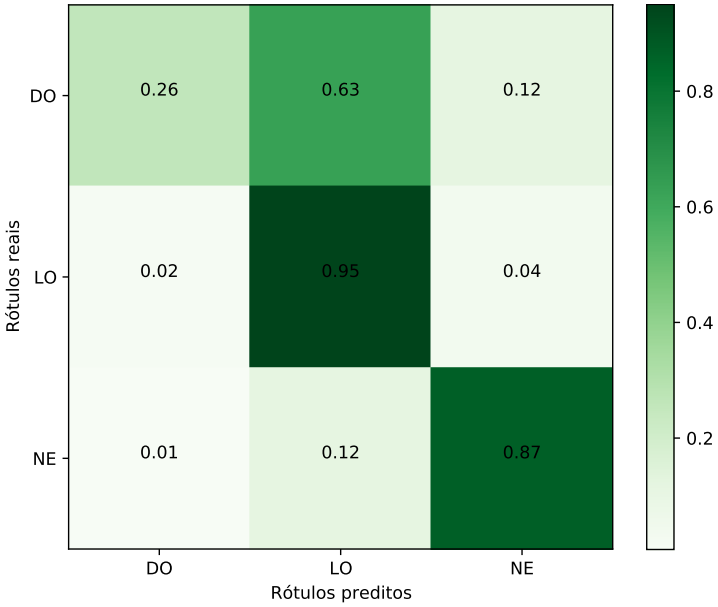
Figura 15 – Matriz de confusão da classificação realizada pelo SVM normalizada pelas colunas



Fonte: produzido pelo próprio autor.

Pela figura é possível ver que as predições para as classes LO e NE foram as mais corretas, com 93% e 80%, respectivamente. Já a classe DO teve apenas 55% de precisão. Entretanto, é importante notar que os erros de predição para essa classe foram em sua maioria para LO e, apesar de pequenos, os erros de predição para a classe LO foram duas vezes mais frequentes para a classe DO que para a classe NE. A Figura 16, por sua vez, exibe a matriz de confusão normalizada pelas linhas, evidenciando o *recall* das classificações.

Figura 16 – Matriz de confusão da classificação realizada pelo SVM normalizada pelas linhas



Fonte: produzido pelo próprio autor.

Nota-se mais uma vez que as predições para as classes LO e NE foram bem sucedidas, com valores para o *recall* de 95% e 87%, respectivamente. Já os dados pertencentes à DO, apenas 26% deles foram preditos corretamente, sendo que 63% foram classificados como LO. Ou seja, em confluência com os dados de precisão, percebe-se que o SVM tem dificuldade de distinguir se discurso de ódio é linguagem ofensiva ou mesmo discurso de ódio, mas diferencia bem quando o tuíte é neutro. Deve-se levar em consideração também que o resultado obtido pelo SVM está diretamente ligado aos métodos de extração de características aplicados. Dessa forma, outras estratégias de modelagem do texto podem ser capazes de fazer essa distinção entre as classes LO e DO que o TF-IDF+SVM não foi capaz de realizar de forma eficaz.

Na execução da rede neural, os parâmetros que forneceram melhores resultados estão resumidos no Quadro 11.

Assim como para o SVM, o *stemming* colabora para a melhoria dos resultados. Entretanto, algumas palavras deixam de ser computadas na camada de *embedding*. Isso ocorre porque algumas delas não têm correspondência no vocabulário treinado pelo GloVe. Nesse caso, houve uma redução de 737 termos, 10,75% do total. Alguns desses termos são grafias pouco frequentes, típicas da linguagem da internet como "*yessssss*" ("simmm", em português) e "*omgggg*" (sigla para " oh meu deus!" em inglês), outros são termos que ao serem simplificados se tornam irreconhecíveis como "*factori*" (fábrica) e "*guilti*" (culpado). Retirar palavras pouco importantes parece ter um efeito positivo, pois a frequência mínima igual a 2 também obteve resultados melhores.

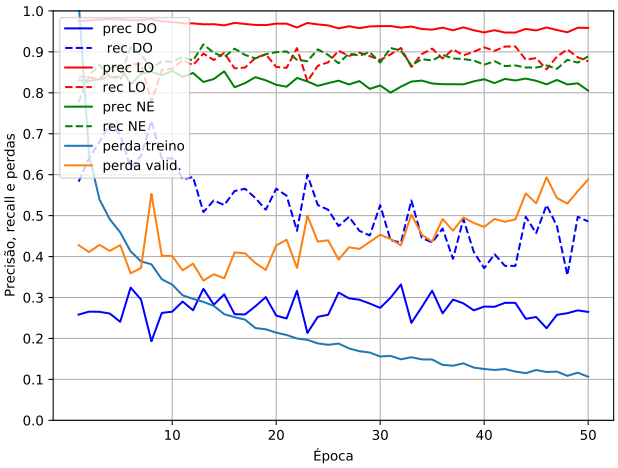
Quadro 11 – Parâmetros e valores testados para otimização da classificação pela CNN

Parâmetro	Valores testados
Stemming	sim
Frequência mínima do termo	2
Dimensão do vetor de *embedding*	200
Permitir treinamento da camada de *embedding*	sim
Kernels	n=(1, 2, 3)
Pesos	1, 1, e 1
Batch	128
Função de perda	*categorical crossentropy*
Otimizador	rmsprop
Taxa de aprendizado	0,001
Proporção treino/teste	70%/30%
Validação	20%

Fonte: produzido pelo próprio autor.

Os tamanhos dos *kernels* da arquitetura original da rede eram 3, 4 e 5. Porém foi observado que *kernels* menores (1, 2 e 3) obtiveram melhor resultado, de forma similar ao SVM que classificou melhor com unigramas e bigramas. Quanto ao treinamento dos pesos da camada de *embedding*, permiti-lo faz com que os vetores treinados pelo GloVe sejam ajustados ao contexto do corpus, melhorando a performance da rede na classificação. Para o treinamento da rede, os pesos não ajudaram a melhorar as taxas de acerto. Mesmo treinando por um número maior de épocas, não houve convergência entre as funções de perda. A Figura 17 mostra o histórico desse treinamento. Nota-se grande oscilação na perda dos dados de validação e da precisão e do *recall* da classe DO. Isso indica que a rede reclassifica grande parte dos dados a cada época, demonstrando indecisão na classificação.

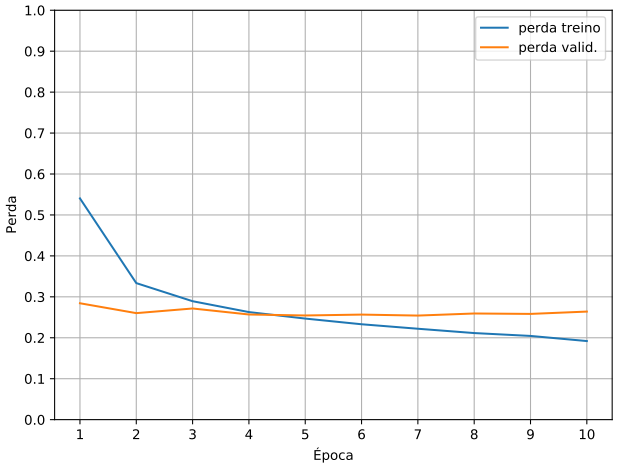
Figura 17 – Evolução das funções de perda e precisão e *recall* com os pesos do Quadro 8 em 50 épocas de treinamento



Fonte: produzido pelo próprio autor.

Os pesos da ponderação da função de perda, então, foram neutralizados, o que estabilizou a função de perda e as classificações. Ao treinar a rede por 10 épocas, notou-se que o ponto ótimo para que não haja *overfitting* se dá com 5 épocas, como pode ser visto no gráfico da Figura 18. A partir desse ponto o valor retornado pela função de perda calculado sobre os dados de validação passa a aumentar, indicando que a rede está aprendendo a identificar os dados treinamento e não a generalizar a predição.

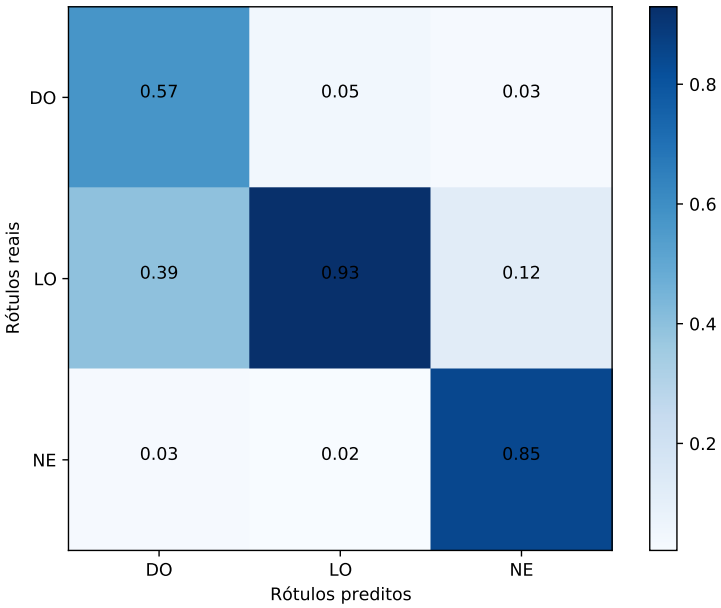
Figura 18 – Evolução da função de perda para os conjuntos de treino e validação em 10 épocas com os novos pesos do Quadro 11



Fonte: produzido pelo próprio autor.

A Figura 19 mostra a matriz de confusão normalizada pela coluna para o resultado de um teste que obteve a menor perda.

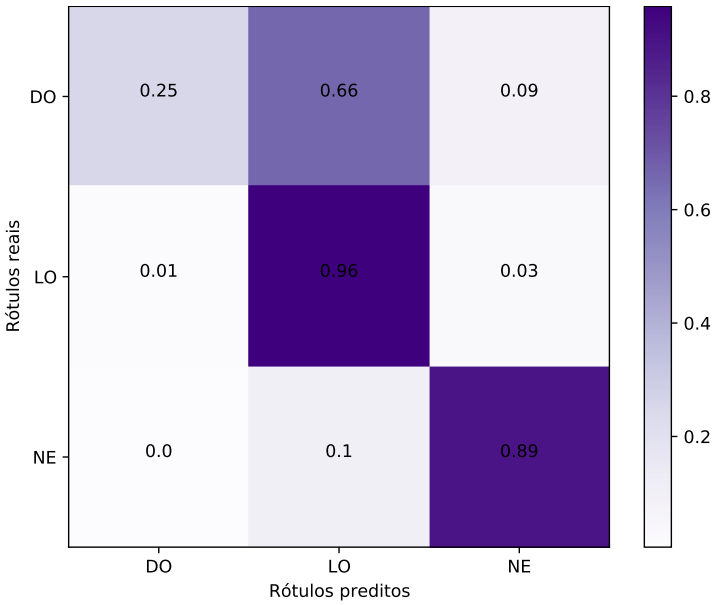
Figura 19 – Matriz de confusão da classificação realizada pela CNN normalizada pelas colunas



Fonte: produzido pelo próprio autor.

Já a Figura 20 mostra a matriz de confusão normalizada pela linha para o resultado do mesmo teste.

Figura 20 – Matriz de confusão da classificação realizada pela CNN normalizada pelas linhas



Fonte: produzido pelo próprio autor.

Percebe-se em ambas as imagens o mesmo padrão de falha de identificação na classe DO observado nas Figuras 15 e 16. Mais da metade dos tuítes da classe DO foram classificados como LO. Enquanto pouco mais da metade dos que foram classificados como DO realmente o são.

Nota-se novamente também que a classe NE consegue ser bem distinguida das outras duas classes, com apenas 10% de falsos negativos e 15% de falsos positivos.

A Tabela 4 resume as taxas de acerto de cada técnica. Apesar de os valores serem muito próximos, nota-se incremento nas taxas em todas as comparações exceto para o *recall* da classe DO, que ainda assim teve um decréscimo de 0,45%. O maior aumento pode ser visto na classe NE, cujo *F-measure* foi de 83,71% para 86,91%. Na classe LO, o *F-measure* obtido pela CNN também foi superior e na classe DO pode-se dizer que foram equivalentes, com pequeno incremento para A CNN. De forma geral, analisando as 3 classes em conjunto, tem-se *F-measure* igual a 89,47% alcançado pelo SVM e para o CNN, tem-se *F-measure* igual a 90,53%.

Ao longo da discussão observou-se a dificuldade de ambos os classificadores em predizerem corretamente tuítes que contêm discurso de ódio. As *F-measures* se aproximam de 35%, o que significa que as predições são piores que o acaso. Mais de 60% desses tuítes foram

Tabela 4 – Resultados de precisão, *recall* e *F-measure* obtidos pelos classificadores SVM e CNN para as classes DO, LO, NE e a junção de DO e LO

Classe	Precisão(%)		*Recall(%)*		*F-measure(%)*	
	SVM	CNN	SVM	CNN	SVM	CNN
DO	55,07	57,44	25,68	25,23	35,02	35,06
LO	92,79	92,94	94,82	95,85	93,79	94,37
NE	80,27	84,61	87,47	89,33	83,71	86,91
Geral	89,47	90,51	89,47	90,51	89,47	90,51

Fonte: produzido pelo próprio autor.

classificados como linguagem ofensiva. Além disso, o uso de pesos no treinamento da rede não incrementou os níveis de acerto.

Investigando os tuítes classificados incorretamente, observa-se que as mesmas palavras ofensivas que são usadas tanto como gírias quanto para de fato ofender, também são usadas para ecoar mensagens de ódio. Por exemplo, o tuíte "*Jay may make an appearence at the party. Love that f*g!*" utiliza a palavra "*f*g*" ("Jay pode aparecer na festa. Amo aquela b*cha!", em português) num contexto não odioso. Já o tuíte "*F*gs are MERELY a scientific indicator that SOMETHING IS WRONG! Do not celebrate as 'unique individuals' for they are cancer.*" ("B*chas são meramente um indicador científico de que algo está errado! Não as celebre como 'indivíduos únicos' pois são um câncer!", em português) a utiliza numa clara manifestação de ódio a um grupo de indivíduos. Portanto, é fácil compreender por que a classe DO tem baixos níveis de acerto, sabendo ainda que ela representa apenas 5,77% dos dados enquanto a classe LO representa 77,43%. Cabe ressaltar também que cada tuíte foi analisado por 3 pessoas de maneira colaborativa, sugerindo inconsistência nas rotulações.

Apesar da baixa taxa de acerto para a identificação de discurso de ódio, nota-se que ambas as técnicas tiveram resultado muito similar. As classes LO e NE tiveram *F-measures* acima de 80% e na classificação geral, as duas ficam em torno de 90%. Entretanto, a CNN se mostrou superior tanto quando os resultados são analisados para cada classe tanto no caso geral.

5 CONCLUSÃO

Neste trabalho foi traçado um experimento que estuda a identificação por CNN de discurso de ódio em texto, mais especificamente, tuítes. O corpus estudado é composto por tuítes extraídos da rede social usando um léxico de palavras ofensivas como critério de coleta e divididos em 3 classes: discurso de ódio, linguagem ofensiva e neutro.

Foi observado a partir do banco de dados que o vocabulário utilizado em textos que contêm ódio é similar ao de textos ofensivos, o que dificulta a diferenciação entre as classes.

No experimento, foi empregado como método *baseline* para comparação dos resultados o algoritmo SVM. Já para a CNN, foi adotada uma arquitetura de rede aplicada por outro autor também na tarefa de identificação de discurso de ódio.

Conclui-se que a arquitetura de CNN escolhida para efetuar o experimento se mostrou capaz de obter resultados de precisão, *recall* e *F-measure* equivalentes ou superiores aos obtidos pelo SVM, mesmo com as complexidades apresentadas pela base de dados explorada.

A contribuição deste trabalho foi aplicar uma rede já testada na tarefa de identificação de discurso de ódio em um novo banco de dados, reforçando a evidência da eficácia tanto dessa arquitetura específica quanto da técnica na tarefa.

Trabalhos futuros podem investigar a separabilidade das classes e buscar pesos que atenuem o desequilíbrio entre elas. Podem também ser avaliadas outras bases de dados para corroborar os achados neste trabalho. Além disso, outras arquiteturas de redes mais profundas podem ser pesquisadas. Outra contribuição pode ser a elaboração de uma base de dados sobre este tópico na língua portuguesa, pois ainda existem poucas. Ademais, pode-se pesquisar a identificação de discurso de ódio em textos que combinam suas mensagens a imagens, linguagem muito utilizada no *Twitter* e em outras redes sociais.

REFERÊNCIAS BIBLIOGRÁFICAS

AGOSTINE, C. Embaixada da China rebate Eduardo Bolsonaro e cita preconceito. **Valor Econômico**, São Paulo, 19 mar. 2020. Disponível em: https://valor.globo.com/politica/noticia/2020/03/19/embaixada-da-china-rebate-eduardo-bolsonaro-e-cita-preconceito.ghtml. Acesso em: 22 nov. 2020.

AIZAWA, A. An information-theoretic perspective of TF-IDF measures. **Information Processing and Management**, [*s. l.*], v. 39, n. 1, Jan. 2003. Disponível em: https://www.sciencedirect.com/science/article/pii/S0306457302000213. Acesso em: 20 maio 2021.

ARCENEAUX, N.; WEISS, A. S. Seems stupid until you try it: press coverage of Twitter, 2006-9. **New Media & Society**, [*s. l.*], v. 12, n. 8, p. 1262–1279, Dec. 2010. Disponível em: https://doi.org/10.1177/1461444809360773. Acesso em: 22 nov. 2020.

BADJATIYA, P.; GUPTA, S.; GUPTA, M.; VARMA, V. Deep learning for hate speech detection in tweets. *In*: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB COMPANION, 26., 2017, Perth. **Proceedings** [...]. Geneva: ACM Press, 2017. p. 759–760. Disponível em: https://doi.org/10.1145/3041021.3054223. Acesso em: 19 nov. 2020.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural Language Processing with Python**. 1st ed. Sebastopol: O'Reilly Media, Inc., 2009.

BRASIL. Lei nº 7.716 de 05 de janeiro de 1989. Define os crimes resultantes de preconceito de raça ou de cor. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, p. 1, 6 jan. 1989. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/L7716compilado.htm. Acesso em: 20 jul. 2020.

BURNAP, P.; WILLIAMS, M. Cyber hate speech on Twitter: An application of machine classification and statistical modeling for policy and decision making: Machine classification of cyber hate speech. **Policy and Internet**, [*s. l.*], v. 7, n. 2, Jun. 2015. Disponível em: https://doi.org/10.1002/poi3.85. Acesso em: 19 nov. 2020.

BURNAP, P.; WILLIAMS, M. Us and them: identifying cyber hate on Twitter across multiple protected characteristics. **EPJ Data Science**, [*s. l.*], v. 5, n. 1, p. 1–15, 23 Mar. 2016. Disponível em: https://doi.org/10.1140/epjds/s13688-016-0072-6. Acesso em: 19 nov. 2020.

CAMELO, F. A. B. **Detecção Automática de Discursos de Ódio em Comentários de Jornais Online**. 2017. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Federal Fluminense, Niterói, 2017.

CHOLLET, F. **Keras**. 2020. The Python deep learning API. Disponível em: https://keras.io. Acesso em: 19 ago. 2020.

CLEMENT, J. Distribution of Twitter users worldwide as of october 2020, by age group. **Statista**, [*s. l.*], 29 Oct. 2020a. Disponível em: https://www.statista.com/statistics/283119/age-distribution-of-global-twitter-users/. Acesso em: 22 nov. 2020.

CLEMENT, J. Most popular social networks worldwide as of july 2020, ranked by number of active users. **Statista**, [*s. l.*], 29 Oct. 2020b. Disponível em: https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/. Acesso em: 22 nov. 2020.

CLEMENT, J. Number of social network users worldwide from 2017 to 2025. **Statista**, [*s. l.*], 15 Jul. 2020c. Disponível em: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/. Acesso em: 22 nov. 2020.

CLEMENT, J. Share of users in the united states who have considered leaving selected social media platforms as of 3rd quarter 2020. **Statista**, [*s. l.*], 23 Sep. 2020d. Disponível em: https://www.statista.com/statistics/262145/us-social-networkers-who-want-use-social-networking-less/. Acesso em: 22 nov. 2020.

DAVIDSON, T.; WARMSLEY, D.; MACY, M.; WEBER, I. Automated hate speech detection and the problem of offensive language *In*: INTERNATIONAL AAAI CONFERENCE ON WEB AND SOCIAL MEDIA, 11., 2017, Montreal. **Proceedings** [...]. Palo Alto: The AAAI Press, 2017. p. 512–515. Disponível em: https://arxiv.org/abs/1703.04009. Acesso em: 22 nov. 2020.

DWOSKIN, E.; WHALEN, J.; CABATO, R. Content moderators at YouTube, Facebook and Twitter see the worst of the web – and suffer silently. **The Washington Post**, Manila, 25 Jul. 2019. Disponível em: https://www.washingtonpost.com/technology/2019/07/25/social-media-companies-are-outsourcing-their-dirty-work-philippines-generation-workers-is-paying-price/. Acesso em: 22 nov. 2020.

EUROPEAN COMISSION. **The Code of conduct on countering illegal hate speech online**. Brussels: [*s. n.*], 2020a. Disponível em: https://ec.europa.eu/commission/presscorner/detail/en/qanda_20_1135. Acesso em: 22 nov. 2020.

EUROPEAN COMISSION. **Countering illegal hate speech online:** 5th evaluation of the Code of Conduct. Brussels: [*s. n.*], 2020b. Disponível em: https://ec.europa.eu/info/sites/info/files/codeofconduct_2020_factsheet_12.pdf. Acesso em: 22 nov. 2020.

FUNG, B. Facebook ad boycott organizers met with zuckerberg. it didn't go well. **CNN Business**, Washington, 8 Jul. 2020. Disponível em: https://edition.cnn.com/2020/07/07/tech/facebook-civil-rights-meeting/index.html. Acesso em: 22 nov. 2020.

GOMES, A.; ANTONIALLI, D.; OLIVA, T. D. Drag queens e inteligência artificial: computadores devem decidir o que é 'tóxico' na internet? **InternetLab**, São Paulo, 28 jun. 2019. Disponível em: https://www.internetlab.org.br/pt/liberdade-de-expressao/drag-queens-e-inteligencia-artificial-computadores-devem-decidir-o-que-e-toxico-na-internet/. Acesso em: 10 jul. 2019.

GÓMEZ, R. G. Um exército para rastrear o ódio nas redes. **El País**, Madri, 23 Mar. 2019. Disponível em: https://brasil.elpais.com/brasil/2019/03/22/tecnologia/1553279547_294211.html. Acesso em: 22 nov. 2020.

HICKS, D.; GASCA, D. A healthier Twitter: Progress and more to do. **Twitter:** Blog, [*s. l.*], 16 Apr. 2019. Disponível em: https://blog.twitter.com/en_us/topics/company/2019/health-update.html. Acesso em: 22 nov. 2020.

INDURKHYA, N.; DAMERAU, F. J. **Handbook of Natural Language Processing**. 2nd ed. Boca Raton: Chapman & Hall/CRC, 2010.

JOACHIMS, T. Text Categorization with Support Vector Machines. *In*: EUROPEAN CONF. MACHINE LEARNING, 10., 1998, Chemnitz. **Proceedings** [...]. Berlin: Springer, 1998. p. 137–142. Disponível em: http://dx.doi.org/10.17877/DE290R-5097. Acesso em: 20 maio 2021.

KEMP, S. Digital 2020: Global digital overview. **DataReportal**, [*s. l.*], 30 Jan. 2020. Disponível em: https://datareportal.com/reports/digital-2020-global-digital-overview. Acesso em: 22 nov. 2020.

LEE, E.-J. Deindividuation effects on group polarization in computer-mediated communication: The role of group identification, public-self-awareness, and perceived argument quality. **Journal of Communication**, Oxford, v. 57, n. 2, p. 385–403, Jun. 2007. Disponível em: https://academic.oup.com/joc/article-abstract/57/2/385/4102671?redirectedFrom=fulltext. Acesso em: 21 nov. 2020.

LOZANO, E.; CEDEÑO, J.; CASTILLO, G.; LAYEDRA, F.; LASSO, H.; VACA, C. Requiem for online harassers: Identifying racism from political tweets. *In*: INTERNATIONAL CONFERENCE ON EDEMOCRACY AND EGOVERNMENT, 4., 2017, Quito. **Proceedings** [...]. Piscataway: IEEE, 2017. p. 154–160. Disponível em: https://ieeexplore.ieee.org/document/7962526. Acesso em: 21 nov. 2020.

LYNN, S. Get busy with word embeddings: An introduction. **Shane Lynn:** Data science, Startups, Analytics, and Data visualisation, [*s. l.*], c2020. Disponível em: https://www.shanelynn.ie/get-busy-with-word-embeddings-introduction/. Acesso em: 22 nov. 2020.

MALINI, F. O método perspectivista de análise de redes sociais: cartografando topologias e temporalidades em rede. *In*: ENCONTRO ANUAL DA COMPÓS, 25., 2016, Goiânia. **Anais** [...]. Goiânia: UFG, 2016. p. 1–30. Disponível em: http://www.compos.org.br/biblioteca/compos_malini_2016_3269.pdf. Acesso em: 21 nov. 2020.

MULLEN, B. Immigrant suicide rates as a function of ethnophaulisms: Hate speech predicts death. **Psychosomatic Medicine**, [*s. l.*], v. 66, n. 3, p. 343–348, May/Jun. 2004. Disponível em: https://journals.lww.com/psychosomaticmedicine/Abstract/2004/05000/Immigrant_Suicide_Rates_as_a_Function_of.9.aspx. Acesso em: 19 nov. 2020.

ORGANIZAÇÃO DAS NAÇÕES UNIDAS. **Strategy and Plan of Action on Hate Speech**. New York: [*s. n.*], 2019. 2 p. Disponível em: https://www.un.org/en/genocideprevention/hate-speech-strategy.shtml. Acesso em: 22 nov. 2020.

ORIOLA, O.; KOTZÉ, E. Evaluating machine learning techniques for detecting offensive and hate speech in south african tweets. **IEEE Access**, [*s. l.*], v. 8, p. 21496–21509, 20 Jan. 2020. Disponível em: https://ieeexplore.ieee.org/document/8963960. Acesso em: 19 nov. 2020.

PAL, S. Implementing word2vec in tensorflow. **Analytics Vidhya**, [*s. l.*], 22 Jun. 2019. Disponível em: https://medium.com/analytics-vidhya/implementing-word2vec-in-tensorflow-44f93cf2665f. Acesso em: 22 nov. 2020.

PARK, J.; FUNG, P. One-step and two-step classification for abusive language detection on Twitter. *In*: WORKSHOP ON ABUSIVE LANGUAGE ONLINE, 1., 2017, Vancouver. **Proceedings** [...]. Vancouver: Association for Computational Linguistics, 2017. p. 41–45. Disponível em: https://www.aclweb.org/anthology/W17-3006. Acesso em: 19 nov. 2020.

PEDDINTI, S.; ROSS, K. W.; CAPPOS, J. User anonymity on Twitter. **IEEE Security and Privacy**, [*s. l.*], v. 15, n. 3, p. 84–87, May 2017. Disponível em: https://ieeexplore.ieee.org/document/7945182. Acesso em: 21 nov. 2020.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, [*s. l.*], n. 12, p. 2825–2830, 2011.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation *In*: CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING, 2014, Doha. **Proceedings** [...]. Stroudsburg: Association for Computational Linguistics, 2014. p. 1532–1543. Disponível em: http://www.aclweb.org/anthology/D14-1162. Acesso em: 5 maio 2021.

PITSILIS, G. K.; RAMAMPIARO, H.; LANGSETH, H. Effective hate-speech detection in Twitter data using recurrent neural networks. **Applied Intelligence**, [*s. l.*], v. 48, n. 12, p. 4730–4742, Jul. 2018. Disponível em: https://doi.org/10.1007/s10489-018-1242-y. Acesso em: 19 nov. 2020.

QUEIROZ, M. D. **Identificação de discurso de ódio e linguagem ofensiva em tuítes usando SVM e CNN**. 2021. Disponível em: https://github.com/mlldorio/hate-speech-svm-cnn. Acesso em: 13 abr. 2021.

ROSA, H.; MATOS, D. M.; RIBEIRO, R.; COHEUR, L.; CARVALHO, J. A "deeper" look at detecting cyberbullying in social networks. *In*: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, 2018, Rio de Janeiro. **Proceedings** [...]. Piscataway: IEEE, 2018. p. 1–8. Disponível em: https://ieeexplore.ieee.org/document/8489211. Acesso em: 19 nov. 2020.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence:** A Modern Approach. 3rd ed. Upper Saddle River: Prentice Hall, 2010.

SAHA, K.; CHANDRASEKHARAN, E.; CHOUDHURY, M. D. Prevalence and psychological effects of hateful speech in online college communities. *In*: ACM CONFERENCE ON WEB SCIENCE, 10., 2019, Boston. **Proceedings** [...]. New York: Association for Computing Machinery, 2019. p. 255–264. Disponível em: https://doi.org/10.1145/3292522.3326032. Acesso em: 19 nov. 2020.

TWITTER. Hateful conduct policy. **Twitter:** Help Center, [*s. l.*], c2020a. Disponível em: https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy. Acesso em: 22 nov. 2020.

TWITTER. Sobre diferentes tipos de tweets. **Twitter:** Central de Ajuda, [*s. l.*], c2020b. Disponível em: https://help.twitter.com/pt/using-twitter/types-of-tweets. Acesso em: 16 jul. 2020.

TWITTER SAFETY. Updating our rules against hateful conduct. **Twitter:** Blog, [*s. l.*], 5 Mar. 2020. Disponível em: https://blog.twitter.com/en_us/topics/company/2019/hatefulconductupdate.html. Acesso em: 10 Jul. 2020.

UNITED NATIONS CONFERENCE ON TRADE AND DEPELOPMENT. Cybercrime legislation worldwide. **UNCTAD:** Prosperity for all, Geneva, 2 Apr. 2020. Disponível em: https://unctad.org/en/Pages/DTL/STI_and_ICTs/ICT4D-Legislation/eCom-Cybercrime-Laws.aspx. Acesso em: 10 jul. 2020.

UZOCHUKWU, C. E.; OKAFOR, E. G. Social media, hate speech and conflict: Interplay of influences. **International Journal of Social Sciences and Humanities Review**, [*s. l.*], v. 9, n. 1, p. 144–158, Jan. 2019. Disponível em: https://www.ijsshr.com/journal/index.php/IJSSHR/article/view/476. Acesso em: 5 maio 2021.

WATANABE, H.; BOUAZIZI, M.; OHTSUKI, T. Hate speech on Twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. **IEEE Access**, [*s. l.*], v. 6, p. 13825–13835, Feb. 2018. Disponível em: https://ieeexplore.ieee.org/document/8292838. Acesso em: 22 nov. 2020.