Evaluation of word embedding techniques for hate-speech detection in Portuguese

Claver Pari Soto[1,2], Gustavo M. S. Nunes[1], José Gabriel R. C. Gomes[1]

[1]Programa de Engenharia Elétrica – COPPE
Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil
[2]Departamento de Computação
Universidade Federal Rural do Rio de Janeiro, Seropédica, RJ, Brasil

claver@pads.ufrj.br, gustavo_mn@poli.ufrj.br, gabriel@pads.ufrj.br

Abstract. *This paper compares the results from the exploration of feature vectors generated by word embedding techniques (word2vec and wang2vec, specifically) from a text database in the order of billion tokens, with the results from using feature vectors generated from small databases in the order of tens of thousands, for the hate speech detection task in Portuguese. This comparison continues the research developed by other authors in Brazil and Portugal, and takes advantage of resources and suggestions made available by them, to explore potential advantages of using smalls datasets for word embeddings.*

Resumo. *Este artigo compara os resultados obtidos da exploração dos vetores de características gerados de técnicas de word embedding (especificamente word2vec e wang2vec) a partir de um banco de textos na ordem do bilhão de tokens com os resultados obtidos utilizando vetores de características gerados a partir de bancos pequenos na ordem de dezenas de milhar, para a tarefa de detecção de discurso de ódio na língua portuguesa. Esta comparação dá continuidade às pesquisas desenvolvidas por outros autores no Brasil e em Portugal, e aproveita os recursos e sugestões por eles disponibilizados, para explorar as vantagens da utilização de bancos de dados menores.*

1. Introdução

Em agosto de 2018 a jornalista Fernanda Pugliero publicou[1] um estudo da situação do conteúdo de ódio na internet brasileira. Desde 2007, quase 4 milhões de denúncias de crimes de ódio foram recebidas pela Central Nacional de Denúncias de Crimes Cibernéticos. Ela apontou também, que a média anual das denúncias está diminuindo pois há menos pessoas dispostas a denunciar e, ao que parece, o sentimento de ódio está se banalizando, naturalizando-se nas redes sociais onde os usuários replicam os discursos, muitas vezes inclusive, sem uma posição crítica.

As plataformas das mídias sociais proíbem comentários de ódio e de assédio. No entanto, aplicar essas regras de proibição, e identificar esses comentários de ódio em grande escala, ainda é um problema não resolvido, como foi constatado em março de 2019, na competição *SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval)*[2], documentada em [Zampieri et al., 2019].

[1]https://www.dw.com/pt-br/como-o-ódio-viralizou-no-brasil/a-45097506, último acesso em: 25/07/2019
[2]https://competitions.codalab.org/competitions/20011, último acesso em: 25/07/2019

A tarefa, no estado da arte, geralmente é modelada como um problema de classificação supervisionada onde os sistemas são treinados usando um conjunto de dados contendo comentários postados, acompanhados dos respectivos rótulos que descrevem a natureza do comentário como sendo ofensivo ou não. O classificador pode ser algum modelo de ML (*Machine Learning*), por exemplo SVM (*Support Vector Machine*), CNN (*Convolutional Neural Network*), RNN (*Recurrent Neural Network*) ou BiLSTM (*Bidirectional Long-Short Term Memory*). Outro lado do problema é a modelagem da linguagem, onde são usados modelos de NLP (*Natural Language Processing*) para extrair dos textos a informação sintática e semântica necessária para construir as representações numéricas que devem ser apresentadas como entrada no classificador anteriormente mencionado.

O objetivo deste artigo é dar continuidade a trabalhos atuais na área de análise de sentimentos em dados reais na língua portuguesa. E especificamente, comparar o desempenho da CNN como classificador em duas situações: I) quando recebe como entrada vetores pré-treinados num domínio geral, II) quando recebe vetores treinados no domínio da aplicação com o modelo de linguagem aplicado no próprio conjunto de dados a ser classificado.

Este trabalho tem seis seções. A Seção 2 contextualiza e descreve brevemente trabalhos relacionados. A Seção 3 apresenta as técnicas utilizadas nos experimentos. A Seção 4 descreve os bancos de dados utilizados, tanto para a classificação como para a representação da língua. A Seção 5 apresenta e discute os resultados obtidos. A Seção 6 apresenta as conclusões e algumas sugestões para trabalhos futuros.

2. Contextualização e Trabalhos Relacionados

Desde a década de 90 existem, na língua inglesa, muitos trabalhos realizados para identificar automaticamente, no mundo digital, mensagens hostis. Por exemplo, [Spertus, 1997] descreve um sistema para reconhecimento de mensagens hostis contidas em e-mails: codifica as mensagens com base na sintaxe e semântica de cada sentença, e as classifica usando árvores de decisão.

A pesquisa em Análise de Sentimento envolve muitas áreas do conhecimento, como Processamento de Linguagem Natural, Inteligência Computacional e Linguística Computacional. O percurso desta área de pesquisa indica que tanto os modelos de linguagem como os modelos de classificação apresentam melhor desempenho quando fazem uso de técnicas da Inteligência Computacional/Aprendizado de Máquina, como descrito brevemente nos seguintes parágrafos.

[Bengio et al., 2003] propõem o treinamento de uma rede neural para o aprendizado da função de probabilidade conjunta da sequência de palavras em dois bancos de dados de textos na língua inglesa. Eles obtêm melhores resultados que o então estado da arte usando modelos de linguagem n-grams em métricas intrínsecas de modelagem da língua.

[Mikolov et al., 2013] propõem usar uma NNLM (*Neural Net Language Model*) simples em comparação com o Modelo de Linguagem Neural proposto por [Bengio et al., 2003]. Essa rede neural não tem camada oculta e ela é especializada na representação de palavras em vetores de valores contínuos de conjuntos de dados muito grandes. Os autores fazem testes com métricas de similaridade sintática e semântica de palavras e disponibilizam o código-fonte encorajando o uso em outras aplicações de NLP. Este

método é conhecido como word2vec, e tem dois modelos: modelo CBOW (*Continuous Bag-Of-Words*) que prevê as palavras alvo, dados os contextos; e modelo Skip-gram, que prevê os contextos, dadas as palavras alvo.

[Kim, 2014] realiza vários experimentos melhorando o estado da arte em algumas tarefas, incluindo análise de sentimentos. O autor usa o word2vec pré-treinado em 100 bilhões de palavras do Google News. Para a classificação propõe uma arquitetura de CNN que é escolhida através de busca em malha. Para análise de sentimentos, usa um conjunto de dados de apreciações positiva/negativa de filmes e outro conjunto com apreciações muito_ positiva/positiva/neutra/negativa/muito_negativa. Dos resultados, o autor deduz que os vetores pré-treinados contêm *features* 'universais' boas o suficiente para serem utilizadas nos conjuntos para classificação.

[Ling et al., 2015] apresentam modificações para o word2vec com o objetivo de gerar *word embeddings*[3] mais adequados para tarefas de sintaxe. Os autores denominaram esse método "word2vec estruturado", mas ele é conhecido como wang2vec (Wang Ling é o nome do primeiro autor). Este método também tem os modelos CBOW e Skip-gram. Os autores relatam que o treinamento de bancos de dados na ordem do bilhão de *tokens* levou ao redor de 24 horas.

[Zhang and Wallace, 2016] apresentam um guia prático, analisando a arquitetura da CNN proposta por [Kim, 2014]. Eles utilizam os mesmos conjuntos de dados para análise de sentimentos, e comparam os resultados obtidos quando a língua é modelada com word2vec e obtidos quando a língua é modelada com GloVe (*Global Vectors for word representation*) [Pennington et al., 2014].

São muitos os trabalhos com outras línguas, por exemplo [Petrolito e Dell'Orletta, 2018] apresentam resultados na análise de sentimentos na língua italiana. Eles usam os *word embeddings* word2vec e fastText [Joulin et al., 2016] treinados com bancos de dados fora do domínio de aplicação na ordem de milhões de tokens, e com bancos de dados no domínio da aplicação de tamanhos menores. Entre os resultados obtidos, observam que o word2vec com os bancos no domínio da aplicação tem melhores resultados que com os bancos fora do domínio de aplicação.

Em Portugal, [Rodrigues et al., 2016] criam o primeiro repositório público de *word embeddings* para a língua portuguesa (do Brasil e de Portugal) treinados com 1,7 bilhões de *tokens*[4], unicamente com word2vec. Os autores apontam que o treinamento de 31 configurações do modelo durou uma semana e meia (uma média de 8 horas cada).

No Brasil, [Hartmann et al., 2017] dão continuidade ao trabalho de [Rodrigues et al., 2016] incluindo o banco de dados deles e de outras fontes. Eles avaliam fastText, GloVe, wang2vec e word2vec treinados com cerca de 1,3 bilhões de tokens, incluindo português brasileiro e europeu. Este número é menor que em [Rodrigues et al., 2016] porque é usado um pré-processamento para reduzir o tamanho do vocabulário. Com os word embeddings obtidos, fazem testes em algumas aplicações de NLP. Os autores encorajam outros pesquisadores a testar em outras aplicações estes *word embeddings* pré treinados, disponibilizando-os num repositório do NILC (Núcleo Interinstitucional de

[3] *word embeddings*: Dado um texto, são as representações das palavras em vetores de números reais, e que contêm algum conhecimento das informações de posicionamento entre as palavras

[4] *token*: unidade atômica de dado de informação na análise de texto. Na prática, no presente artigo, toda string delimitada por espaços ou sinais de pontuação é considerada como token.

Linguística Computacional). No presente trabalho, estes *word embeddings* são referenciados como NILC-embeddings. Aparentemente, não existe na internet, outro repositório com vetores pré-treinados com o modelo wang2vec.

No ano seguinte, [de Pelle e Moreira, 2018] criam dois conjuntos de sentenças rotuladas ofensiva/não_ofensiva obtidas de um site de notícias brasileiro. Para a codificação das características das sentenças usam misturas de n-grams. Posteriormente, [Lima e Dal Bianco, 2019] dão continuidade a essa pesquisa acrescentando, na codificação das características, meta-atributos gerados através de informações retiradas da vizinhança de cada sentença. Ambos trabalhos usam SVM para a classificação das sentenças.

Em Portugal, [Fortuna, 2017] cria um banco de sentenças rotuladas para análise de discurso de ódio. Para a codificação das sentenças usa os n-grams, e para classificação testa vários algoritmos, entre eles, MLP e SVM.

No Brasil, [Silva e Serapião, 2018] executam a classificação de sentimentos de ódio usando CNN, os dois bancos de dados de [de Pelle e Moreira, 2018], o banco de dados de [Fortuna, 2017] e os NILC-embeddings wang2vec e GloVe de [Hartmann et al., 2017], conseguindo os melhores resultados com wang2vec.

No presente trabalho, se dá continuidade ao trabalho de [Silva e Serapião, 2018]. Mantém-se o wang2vec, e traz-se o word2vec por ser o mais utilizado, com melhor desempenho, na maioria dos trabalhos sobre análise de sentimentos. A contribuição deste trabalho é a análise do uso de vetores pré-treinados NILC-embeddings, e do uso de vetores treinados no domínio da aplicação, com cada um dos bancos de dados para classificação de discursos de ódio na língua portuguesa.

3. Técnicas Empregadas

A tarefa de análise de sentimentos requer a utilização de técnicas e modelos que podem ser colocados em três grupos: pré-processamento dos textos, modelos de linguagem que representam a língua na aplicação, e modelos para a classificação de sentimentos.

3.1. Pré-Processamento

As sentenças nos bancos de dados foram divididas em *tokens* isolados, e seguindo as sugestões de [Hartmann et al., 2017], normalizados a fim de reduzir o tamanho do vocabulário: os numerais foram substituídos por zeros, as URLs foram mapeadas para o *token* "url" e os e-mails foram mapeados para o *token* "email". Foram removidos os sinais de pontuação e todas as palavras foram reescritas em minúsculo. Além disso, as sentenças foram preenchidas com zeros de modo a alcançar o comprimento da sentença mais longa em cada banco de dados.

3.2. Modelos da Linguagem (Word Embeddings)

Este artigo considera a comparação do modelo Skip-gram do word2vec amplamente usado para a representação da linguagem em aplicações de análise de sentimentos, com o modelo Skip-gram Estruturado do wang2vec que embora concebido para problemas de sintaxe, teve bons resultados na aplicação em semântica, na análise de sentimentos, conforme [Silva e Serapião, 2018]. Por simplicidade, nas próximas seções, o primeiro modelo será referido como word2vec, e o segundo como wang2vec.

Modelo Skip-gram do word2vec

[Mikolov et al., 2013] apresentam os modelos CBOW e Skip-gram. O modelo Skip-gram é a escolha mais utilizada para criar *word embeddings* pois enfatiza as relações entre uma palavra arbitrária e o contexto (palavras vizinhas) dela. O modelo aprende os *embeddings* começando com um conjunto inicial de *word embeddings* e iterativamente ajusta os *embeddings* de cada palavra *w* para ser mais similar aos *embeddings* de palavras que aparecem próximas à palavra *w* num raio definido, e menos similar aos *embeddings* de palavras que não estão dentro desse raio.

Modelo Skip-gram Estruturado do wang2vec

[Ling et al., 2015] apresentam o modelo "Skip-gram estruturado" como uma adaptação do modelo Skip-gram do word2vec. Os autores argumentam que o word2vec é insensível à ordem com que as palavras aparecem nos textos, sendo portanto, útil para aprender representações semânticas, e sub-ótimos para tarefas que envolvem sintaxe. No treinamento do modelo Skip-gram estruturado é considerada a ordenação com que as palavras aparecem nos textos.

3.3. Modelo de classificação - Arquitetura da CNN

No presente trabalho se usa a CNN como modelo de classificação. Embora concebida para aplicações de processamento de imagens, a CNN tem demonstrado bom desempenho em aplicações de NLP, por exemplo, [Kim, 2014] e [Zhang and Wallace, 2016].

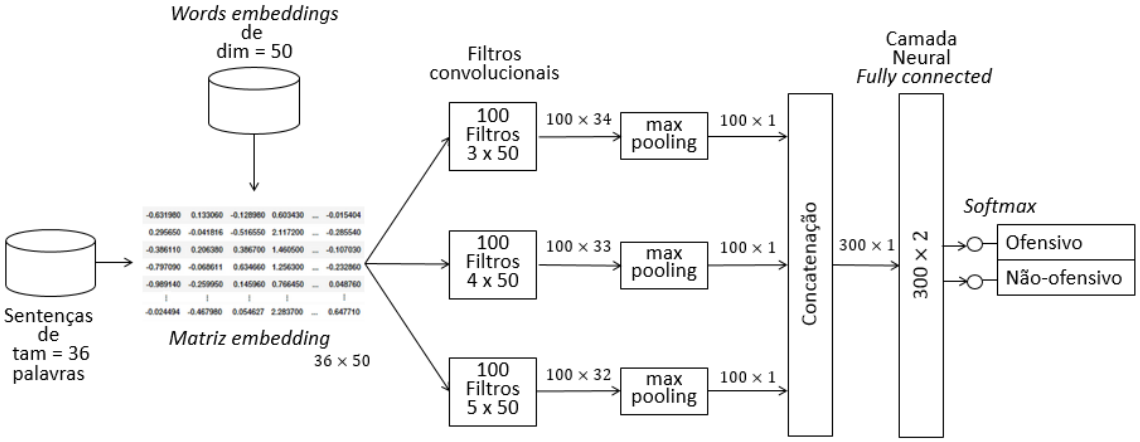

Figura 1. Arquitetura da CNN para classificação de sentenças em língua portuguesa.

Foi adotada a arquitetura proposta por [Kim, 2014] para uso com sentenças na língua portuguesa. A Figura 1 ilustra a arquitetura para o caso da modelagem da língua com vetores de 50 características, e considerando tamanho de 36 palavras para cada sentença a classificar. Com 36 palavras em cada sentença temos uma matriz *embedding* com dimensão 36x50. A matriz *embedding* é processada, através de convolução bidimensional, por 300 filtros. Desses 300 filtros, 100 têm tamanho 3, 100 têm tamanho 4 e 100 têm tamanho 5. Os filtros convolucionais geram mapas de características de comprimento variável, neste caso 34, 33 e 32. Cada convolução é sucedida por uma função de ativação ReLU (*Rectified Linear Unit*). Cada um dos 300 canais é transformado em um único número através de max-pooling. O vetor de tamanho 300 resultante serve

como entrada a uma camada totalmente conectada com duas saídas. Finalmente, a camada softmax identifica a classificação da sentença apresentada.

4. Bancos de Dados

Nesta seção são apresentados os bancos de dados utilizados para a classificação, e os usados para representação da linguagem.

4.1. Bancos de Dados para Classificação

Em língua portuguesa do Brasil, [de Pelle e Moreira, 2017] apresentam comentários ofensivos e não ofensivos coletados de um portal de notícias do Brasil das seções de política e de esporte, organizados em dois bancos de dados OffComBr (Offensive Comments Brazilian web)[5]. Os bancos de dados contém 10.336 comentários postados em 115 notícias no portal. Foram selecionados aleatoriamente 1.250 para efeito de anotação/julgamento humano. Cada comentário é rotulado por três anotadores separadamente, que respondem se consideram que o comentário é ofensivo ou não. Tal conjunto é denominado como OffComBr2, pois pelo menos dois anotadores terão o mesmo parecer. A partir desse, é criado o OffComBr3 onde cada comentário tem a concordância do julgamento dos três anotadores.

Em língua portuguesa de Portugal, [Fortuna, 2017] apresenta um banco de dados[6] extraídos do Twitter. Da quantia original de 42.390 tuítes, 5.668 são rotulados por dois anotadores humanos como ofensivo ou não ofensivo. No presente trabalho esse banco de dados é denominado como HsdPt (Hate speech dataset Portugal).

A Tabela 1 apresenta uma descrição em números dos bancos de dados considerados neste artigo. O HsdPt é o maior, mas suas sentenças são de tamanho menor (a sentença mais longa tem 36 palavras). Nos três bancos de dados é notória a prevalência de comentários não ofensivos: mais de 66% nos três casos.

Tabela 1. Descrição dos conjuntos de sentenças utilizados

	Comen-tários	Ofensivos		Não ofensivos		Tam. da maior sentença	Tam. do vocabulário	Nro. de tokens
OffComBr3	1.033	202	(19,5%)	831	(80,5%)	94	412	10.255
OffComBr2	1.250	419	(33,5%)	831	(66,5%)	94	480	12.207
HsdPt	5.668	1.228	(21,5%)	4.440	(78,5%)	36	2051	75.223

4.2. Banco de Dados para Modelamento da Linguagem

NILC-embeddings[7], fruto de [Hartmann et al., 2017] é um repositório de *word embeddings* pré-treinados para a língua portuguesa do Brasil e do Portugal, considerando fontes e gêneros variados, totalizando mais de 1,3 bilhões de *tokens*. Com esse banco de dados os autores treinaram modelos usando word2vec, fastText, wang2vec e GloVe. Os autores apontam que os treinamentos foram realizados com parâmetros de treino padrão dos respectivos modelos. Não é informado o tempo necessário para o treinamento de cada

[5]http://inf.ufrgs.br/~rppelle/hatedetector/, último acesso em: 25/07/2019
[6]https://rdm.inesctec.pt/id/dataset/cs-2017-008, último acesso em: 25/07/2019
[7]http://nilc.icmc.usp.br/embeddings, último acesso em: 25/07/2019

configuração, mas trabalhos similares relatam ser de aproximadamente 24 horas, como relatado em [Ling et al., 2015], e aproximadamente 8 horas, como relatado em [Rodrigues et al., 2016]. No presente trabalho utilizamos os *word embeddings* dos modelos Skip-gram do word2vec e do wang2vec disponibilizados nesse repositório do NILC, indicados como "NILC-embedding" nas Tabelas 4, 6 e 8.

[Ling et.al., 2015] disponibilizam o código[8] para treinar as redes neurais que geram os *word embeddings* nos modelos CBOW e Skip-gram de word2vec e wang2vec utilizando textos arbitrários. No presente trabalho são treinados unicamente os modelos Skip-gram do word2vec e wang2vec sobre os bancos de dados OffComBr3, OffComBr2 e HsdPt. Para cada *word embedding* são utilizados os parâmetros de treino padrão e os comprimentos dos *words embeddings* considerados são 50, 100 ou 300, como se pode apreciar nas Tabelas 4, 6 e 8. O treinamento de cada *word embedding* demora em média 4 segundos, porque o número de *tokens* é bem baixo, sendo o maior de aproximadamente 75 mil tokens do conjunto HsdPt, como pode ser constatado na Tabela 1.

5. Resultados e Discussão

Nesta seção são apresentados os resultados obtidos utilizando as técnicas explicadas na Seção 3, aplicadas aos bancos de dados descritos na Seção 4.

No presente trabalho a CNN foi modelada e treinada utilizando a biblioteca de código aberto TensorFlow[9]. Para o treinamento da rede foram usados os três bancos de dados descritos na Seção 4 para cada caso de *word embeddings* detalhados nas Tabelas 4, 6 e 8. A Tabela 2 descreve os hiperparâmetros da CNN adotados em todos os casos. Para efeitos de comparação, foram adotados os valores de hiperparâmetros que deram os melhores resultados em [Silva e Serapião, 2018].

Tabela 2. Valores de hiperparâmetros no treinamento da CNN

Hiperparâmetro	Valor
Número de épocas	50
Tamanho dos mini-batch	50
Taxa do *dropout* na camada totalmente conectada	0,5

No treinamento com os bancos de dados OffComBr3 e OffComBr2 foi utilizado o método de otimização Adam com os seguintes valores para os parâmetros: learning_rate=0.001, beta1=0.9, beta2=0.999, epsilon=1e-08. Para o banco HsdPt foi utilizado o método de otimização RMSProp com valores de parâmetros: learning_rate=0.001, decay=0.9, momentum=0.0, epsilon=1e-10.

No treinamento da CNN, para cada experimento realizado neste artigo utilizou-se a técnica de validação cruzada de 10 *folds*. Cada conjunto de dados para classificação foi dividido inicialmente em dois conjuntos: treinamento e teste. O grupo teste, com 10% dos dados não participa do treinamento. O conjunto de treinamento, com os 90% dos dados restantes, é dividido em 10 partes; com cada uma dessas partes é realizada a validação do treinamento feito com as outras nove partes. Temos então 10 modelos treinados com a

[8]https://github.com/wlin12/wang2vec, último acesso em: 25/07/2019
[9]https://www.tensorflow.org, último acesso em: 25/07/2019

mesma arquitetura e cada um deles é testado com o conjunto de teste calculando as métricas F-score e Acurácia. Os valores apresentados nas Tabelas 4, 6 e 8 são a média das 10 respectivas medições.

[Kim, 2014] propõe a arquitetura descrita na Seção 3 utilizada para análise de sentimento na língua inglesa, e [Silva e Serapião, 2018] a utilizam para a língua portuguesa. [Silva e Serapião, 2018] também utilizam os três bancos de dados descritos na Seção 4.1. Eles realizam o pré-processamento que consiste em dividir os textos em tokens, reescrever todos em minúsculo e apagar os sinais de pontuação. Comparam o desempenho da CNN com os *words embeddings* wang2vec e GloVe fornecidos pelo NILC (vetores de tamanho 50, 100 e 300) e com vetores criados usando o método BOW (bag-of-words) sobre cada um dos bancos de dados. Esses autores disponibilizam[10] o código-fonte, no qual nos baseamos para conduzir os experimentos feitos no presente trabalho. Nós executamos o código usando o pré-processamento descrito na Seção 3.1 e a validação cruzada descrita no parágrafo anterior.

5.1. Resultados com o Banco de Dados OffComBr3

A Tabela 3 apresenta alguns dos resultados reportados por outros trabalhos na tarefa de detecção de discursos de ódio, utilizando o banco de dados OffComBr3. O sinal "–" em algumas células indica que os respectivos autores não utilizaram essa métrica.

Tabela 3. Resultados de outros autores com o conjunto OffComBr3

Referência	Representação	Classificação	F-score	Acurácia
[de Pelle and Moreira, 2017]	n-grams	SVM 10-folds	0,82	–
[Lima e Bianco, 2019]	n-grams e meta-atributos gerados através de informações das vizinhanças de cada comentário	SVM	0,85	–
[Silva e Serapião, 2018]	NILC-embedding wang2vec, 300 dim	CNN 10-folds	0,96	92,82 %

Na Tabela 4 estão os resultados do presente trabalho, da classificação da CNN usando os *word embeddings* wang2vec e word2vec pré-treinados do NILC-embeddings (lado esquerdo), e usando os respectivos *word embeddings* treinados no próprio banco de dados OffComBr3 (lado direito). O melhor desempenho de classificação da CNN foi observado em dois casos: ao se utilizar o wang2vec do NILC-embeddings, e o word2vec treinado no próprio banco de dados (destacados em negrito e com fundo cinza na Tabela 4). Ele é similar em relação ao segundo melhor desempenho, o qual foi obtido usando o wang2vec treinado no próprio banco de dados (destacado em negrito na Tabela 4). A principal diferença é que o NILC-embedding usa 200 elementos a menos no *word embedding*.

O banco de dados de dados é o de menor tamanho considerado neste artigo (ver Tabela 1). Os resultados indicam desempenhos similares, com uma ligeira vantagem dos word2vec do OffComBr3-embedding se comparados com os word2vec do NILC-embedding. Nas colunas wang2vec ocorre algo diferente: ligeira vantagem do NILC-embedding nas linhas dim=50 e dim=100 se comparadas com o OffComBr3-embedding.

[10]https://github.com/samcaetano/hatespeech_detector, último acesso em: 25/07/2019

Tabela 4. Resultados com o conjunto OffComBr3. Melhor resultado, em negrito e fundo cinza. Segundo melhor resultado, em negrito.

	NILC-embedding				OffComBr3-embedding			
	wang2vec		word2vec		wang2vec		word2vec	
dim	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)
50	0,89	80,97	0,88	79,61	0,89	80,19	0,88	80,10
100	**0,89**	**81,36**	0,88	79,61	0,89	80,87	0,89	80,29
300	0,89	80,49	0,89	80,97	**0,89**	**81,26**	**0,89**	**81,36**

5.2. Resultados com o Banco de Dados OffComBr2

A Tabela 5 mostra alguns dos resultados reportados por outros trabalhos para a tarefa de detecção de discurso de ódio, utilizando o banco de dados OffComBr2. Este banco de dados apresenta tamanho similar ao OffComBr3, discutido na seção anterior. No OffComBr2, a máxima acurácia alcançada pela CNN é de 77,68%, conforme mostrado na Tabela 6. Esse é o menor valor de acurácia dentre os melhores valores de acurácia reportados no presente trabalho (nos três bancos de dados considerados). Uma possível explicação para o desempenho inferior aos demais consiste no fato de que, na construção desse banco de dados, os rótulos de 17,36% das sentenças tiveram concordância de somente dois dos três anotadores, como foi descrito na Seção 4.1 e na Tabela 1.

Tabela 5. Resultados de outros autores com o conjunto OffComBr2

Referência	Representação	Classificação	F-score	Acurácia
[de Pelle and Moreira, 2017]	n-grams	SVM 10-folds	0,77	—
[Lima e Bianco, 2019]	n-grams e meta-atributos gerados através de informações das vizinhanças de cada comentário	SVM	0,72	—
[Silva e Serapião, 2018]	NILC-embedding wang2vec, 100 dim	CNN 10-folds	0,89	82,64 %

O *word embedding* word2vec do NILC-embedding tem o melhor desempenho (77,68%), como mostra a Tabela 6. O segundo melhor resultado é do wang2vec do OffComBr2-embedding, cujo desempenho é inferior em, aproximadamente, 1% (76,56%). Nestes dois resultados, o *word embedding* tem dimensão 300. Pode-se observar resultados similares ao resultado do segundo colocado na coluna wang2vec do OffComBr2-embedding. Em contrapartida, na coluna word2vec do NILC-embedding, observa-se uma maior variação entre os resultados apresentados. Neste caso, o melhor resultado (dim. 300) supera em, aproximadamente, 2% e 4% os demais resultados (dim. 100 e dim. 50, respectivamente). Situação similar ocorre na coluna wang2vec do NILC-embedding.

Tabela 6. Resultados com o conjunto OffComBr2. Melhor resultado, em negrito e fundo cinza. Segundo melhor resultado, em negrito.

	NILC-embedding				OffComBr2-embedding			
	wang2vec		word2vec		wang2vec		word2vec	
dim	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)
50	0,83	74,88	0,83	73,76	0,84	75,36	0,84	75,52
100	0,84	76,00	0,84	75,84	0,85	76,32	0,84	75,36
300	0,85	77,04	**0,86**	**77,68**	**0,85**	**76,56**	0,85	76,24

5.3. Resultados com o Banco de Dados HsdPt

[Fortuna, 2017] que criou e usou o banco de dados HsdPt (vide Tabela 1) extraiu os vetores de características utilizando n-grams. Como o HsdPt apresenta um desbalanceamento entre comentários positivos e negativos, a autora, em uma parte de sua pesquisa, trabalhou com o banco de dados balanceado em 1228 sentenças para cada sentimento. Ela treinou o classificador com 2.056 sentenças balanceadas, usando validação cruzada de 3 *folds*, e realizou os testes com 400 sentenças balanceadas. [Silva e Serapião, 2018] treinaram outro classificador sem compensar o desbalanceamento do banco de dados HsdPt. A Tabela 7 mostra os resultados obtidos por [Fortuna, 2017] e [Silva e Serapião, 2018].

Tabela 7. Resultados de outros autores com o conjunto HsdPt

Referência	Balanceado	Representação	Classificação	F-score	Acurácia
[Fortuna, 2017]	Sim	n-grams	SVM Linear 3-folds	0,76	78,30 %
[Silva e Serapião, 2018]	Não	NILC-embedding wang2vec, 100 dim	CNN 10-folds	0,96	92,74 %

A Tabela 8 apresenta os resultados do presente trabalho utilizando o banco de dados HsdPt completo, desbalanceado, e descrito na Seção 4.1. Dentre os bancos de dados considerados no presente trabalho, o HsdPt é o que apresenta o maior número de sentenças (5.668). Os resultados revelam desempenhos de classificação similares usando o NILC-embedding e usando o HsdPt-embedding, com a diferença que o HsdPt-embedding apresenta seu melhor resultado com dimensão 50 e o NILC-embedding, com dimensão 100.

Tabela 8. Resultados com o conjunto HsdPt. Melhor resultado, em negrito e fundo cinza. Segundo melhor resultado, em negrito.

	NILC-embedding				HsdPt-embedding			
	wang2vec		word2vec		wang2vec		word2vec	
dim	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)	F-score	Acurácia (%)
50	0,92	86,89	0,92	87,19	**0,92**	**87,30**	0,92	87,12
100	0,92	87,10	**0,92**	**87,30**	0,92	87,12	0,92	86,90
300	0,92	87,10	0,92	86,86	**0,92**	**87,21**	0,92	87,14

6. Conclusões e Trabalhos Futuros

O presente artigo dá continuidade aos trabalhos com a língua portuguesa e observa experimentalmente a semelhança no desempenho da classificação com CNN usando wang2vec e word2vec. Ainda, aponta a vantagem de usar os *word embeddings* treinados com textos do domínio da aplicação, mesmo sendo estes de tamanho muito menor em tokens. [Silva e Serapião, 2018] observaram experimentalmente a superioridade do wang2vec do NILC-embedding em comparação com o GloVe do NILC-embedding ao serem usados com a arquitetura CNN.

As Tabelas 4, 6 e 8 apresentam resultados com *word embeddings* de tamanho 50, 100 e 300. Em alguns casos, os melhores resultados, ou resultados próximos aos melhores, são obtidos utilizando word embeddings com dimensões menores. Isso foi observado, por exemplo, na Tabela 4, onde os dois melhores colocados apresentam desempenho igual, mas com uma diferença de 200 entre o tamanho das dimensões dos words embeddings (dimensão 100 no NILC-embedding e dimensão 300 no OffComBr3-embedding). Outro exemplo é na Tabela 8, onde os dois melhores desempenhos são obtidos utilizando word embeddings com dimensões 50 e 100 (no HsdPt-embedding e NILC-embedding, respectivamente). Pode ser possível encontrar melhores resultados testando com outros tamanhos de *word embeddings*. Fica como sugestão considerar conjuntos de treinamento medianos para o modelo da linguagem (na ordem da centena de milhares de tokens) coletando mais sentenças do domínio da aplicação, tendo em vista que o treinamento é rápido, sendo assim viável de usar uma busca em malha para os parâmetros do word2vec e wang2vec.

Recentemente [Song et al., 2018] afirmam obter melhores resultados em aplicações de semântica e de sintaxe com a proposta de melhorar o word2vec e o wang2vec ao considerar informação específica da posição de cada palavra contexto: se está no lado "esquerdo" ou se está no lado "direito" de determinada palavra na sequência. Sugere-se experimentar este modelo tendo em vista que os autores afirmam ainda, ter melhor resposta em banco de dados pequenos, se comparados com os modelos existentes.

A fim de entender a influência do pré-processamento adotado nas pesquisas com análise de sentimento, fica a sugestão de fazer um estudo focando nesse assunto. [Angiani et al., 2016] apresentam um estudo dos métodos de pré-processamento aplicados para análise de sentimentos, e a comparação da efetividade desses métodos na acurácia dos resultados na classificação.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) – Código de financiamento 001, e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Projetos 432997/2018-0 e 309602/2016-5.

Referências

Zampieri M., Malmasi S., Nakov P., Rosenthal S., Farra N., and Kumar R. (2019). "SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval)". In: Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), Minneapolis, Minnesota, USA.

Spertus Ellen (1997). "Smokey: Automatic Recognition of Hostile Messages". In: IAAI-97 Proceedings. https://www.aaai.org/Papers/IAAI/1997/IAAI97-209.pdf.

Bengio Y., Ducharme R., Vincent P. and Jauvin C. (2003). “A Neural Probabilistic Language Model”. In: Journal of Machine Learning Research 3.

Mikolov T., Chen K., Corrado G. and Dean J. (2013). “Efficient Estimation of Word Representation in Vector Space”. arXiv:1301.3781v3.

Kim Yoon. (2014). “Convolutional Neural Networks for Sentence Classification”. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar.

Ling W., Dyer C., Black A. and Trancoso I. (2015). “Two/Too Simple Adaptations of Word2Vec for Syntax Problems”, In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies", Denver, Colorado.

Zhang Y., Wallace B. (2016). “A Sensitivity Analysis of (and Practitiones’ Guide to) Convolutional Neural Networks for sentence Classification”. arXiv:1510.03820.

Pennington J., Socher R. and Manning C. (2014). “GloVe: Global Vectors for Word Representation.” In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar .

Petrolito R. and Dell’Orletta F. (2018). “Word Embeddings in Sentiment Analysis“. In Proceedings of 5th Italian Conference on Computational Linguistics. Turin, Italy.

Joulin A., Grave E., Bojanowski P. and Mikolov T. (2016). “Bag of Tricks for Efficient Text Classification”. arXiv:1607.01759v3.

Rodrigues J., Branco A., Neale S. and Silva J. (2016). “LX-DSemVectors: Distributional Semantics Models for Portuguese”. In: Computational Processing of the Portuguese Language: 12th International Conference, PROPOR 2016, Tomar, Portugal.

Hartmann N., Fonseca E., Shulby C., Treviso M., Rodrigues J. and Aluísio S. (2017). “Portuguese Word Embeddings: Evaluatin on Word Analogies and Natural Language Tasks”. In: Proceedings of Symposium in Information and Human Language Technology, Uberlândia, MG, Brazil.

de Pelle R. and Moreira V. (2017). “Offensive Comments in the Brazilian Web: a dataset and baseline results”, In: Proceedings of Brazilian Workshop on Social Network Analysis and Mining, São Paulo, SP, Brazil.

Lima C. e Bianco G. (2019). “Extração de características para identificação de discurso de ódio em documentos”, Em: XV Escola Regional de Informática de Banco de Dados, pages 70-78, Chapecó, SC, Brazil.

Fortuna P. (2017). “Automatic Detection of HateSpeech in Text: An Overview of the Topic and Dataset Annotation with Hierarchical Classes”. Dissertação de mestrado Integrado em Eng. Informática e Computação. Faculdade de Engenharia da Universidade do Porto.

Silva S., e Serapião A. (2018). “Detecção de discurso de ódio em português usando CNN combinada a vetores de palavras”, In: Proceedings of KDMILE 2018, Symposium on Knowledge Discovery, Mining and Learning, São Paulo, SP, Brazil.

Song Y., Shi S., Li J., Zhang H. (2018). “Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings”. In: Proceedings of NAACL-HLT 2018, New Orleans, Louisiana.

Angiani G., Ferrari L., Fontanini T., Fornacciari P., Iotti E., Magliani F., and Manicardi S. (2016). “A Comparison between Preprocessing Techniques for Sentiment Analysis in Twitter”. In: Proceedings of the 2nd International Workshop on Knowledge Discovery on the WEB, Cagliari, Italy.