An Analysis of the Sentiment Classification of Short Messages Using Wod2Vec

Raul de Araújo Lima[1], **Paulo T. Guerra**[1]

[1]Universidade Federal do Ceará – Campus Quixadá
Av. José de Freitas Queiroz, 5003 – Cedro – Quixadá – CE 63.902-580

raul.araujo3212@alu.ufc.br, paulodetarso@ufc.br

Abstract. *Sentiment analisys and the polarity classification of texts constitute one of the main tools currently used by companies and organizations for the most varied purposes. This work presents an analysis of the use of word embeddings, built through Word2Vec, in the process of features extraction for polarity classification of short messages written in English. The texts used were extracted from Twitter and the results obtained show that, in spite of the possible need to use larger textual bases to obtain better vectors, Word2Vec is a promising tool for the features extraction of textual data, contributing to obtain good classification results.*

Resumo. *A análise de sentimentos e a classificação de polaridade de textos constituem-se como duas das principais ferramentas atualmente utilizadas por empresas e organizações para os mais variados fins. Este trabalho apresenta uma análise da utilização da representação vetorial incorporada de palavras, construída através do Word2Vec, no processo de extração de características para a classificação de polaridade de mensagens curtas escritas em língua inglesa. Os textos utilizados foram extraídos do Twitter e os resultados obtidos mostram que, apesar da possível necessidade da utilização de bases textuais maiores para que sejam obtidos vetores mais bem incorporados, o Word2Vec constitui-se como uma ferramenta promissora para a extração de características textuais, contribuindo para a obtenção de bons resultados de classificação.*

1. Introdução

A utilização das redes sociais vem crescendo gradativamente e, com isso, cresce também o volume de mensagens que difundem opiniões, ideias e muitas outras informações relevantes para os mais diversos interesses. Uma das maiores e mais utilizadas redes sociais é o Twitter[1], um *microblog* onde os usuários podem compartilhar textos e difundir opiniões e informações sobre os mais variados assuntos [de França and Oliveira 2014]. Uma particularidade do Twitter é que os *tweets*, nome dado às mensagens compartilhadas através da rede social, possuíram, por bastante tempo, um limite máximo de 140 caracteres (hoje são até 280 caracteres), reduzindo consideravelmente o tamanho dos textos publicados pelos usuários.

De modo geral, os textos extraídos do Twitter e das demais redes sociais contêm gírias, abreviações de palavras e expressões próprias do ambiente virtual que dificultam

[1]https://twitter.com

o processo automático de análise e classificação [Lochter et al. 2015]. Nesse contexto, a análise de sentimentos, juntamente com o uso de técnicas de aprendizado de máquina e a representação vetorial de palavras, surgem como as principais e mais eficazes ferramentas para o processo de extração de características e classificação de textos, como demonstram os trabalhos de [Zhang et al. 2015, Xue et al. 2014, Aguiar and Prati 2015].

Este artigo se propõe a avaliar o uso do Word2Vec no processo de análise de polaridade de sentimentos de mensagens curtas extraídas do Twitter, utilizando como base alguns dos conjuntos de dados criados e disponibilizados no trabalho de [Lochter et al. 2015] e comparando os resultados obtidos com a abordagem apresentada pelos autores supracitados, utilizando, para tanto, as mesmas métricas de classificação.

O restante deste artigo está organizado da seguinte maneira. A Seção 2 apresenta um breve resumo sobre alguns trabalhos relacionados. Os classificadores de Aprendizado de Máquina utilizados neste trabalho são apresentados na Seção 3 A representação vetorial de palavras por meio do Word2Vec é apresentado na Seção 4. A Seção 5 descreve os procedimentos metodológicos realizados neste trabalho, bem como os resultados obtidos. Finalmente, as principais conclusões e discussões, bem como os direcionamentos para a realização de trabalhos futuros, são apresentados na Seção 6.

2. Trabalhos Relacionados

Diversos trabalhos se propõem a classificar emoções em textos. [Lochter et al. 2015] propõem a utilização de normalização léxica e indexação semântica na fase de pré-processamento das mensagens realizando a substituição de termos, como gírias e palavras comumente utilizadas somente em redes sociais, por palavras canônicas da língua inglesa. Propõem, ainda, a utilização de um comitê de classificadores, em que vários classificadores são executados sobre a instância do texto cuja classe é desconhecida e ”opinam”para chegar a um resultado final de classificação. Os autores utilizam sete bases de dados, sendo três delas construídas pelos próprios autores e pré-processadas utilizando a abordagem citada anteriormente, as demais bases foram pré-processadas segundo os procedimentos descritos por [Saif et al. 2013].

[Zhang et al. 2015] buscam classificar comentários chineses em positivos ou negativos e propõem a utilização do Word2Vec combinado a um classificador SVM. A base de dados utilizada continha cerca de 100.000 comentários sobre roupas coletados da página chinesa da Amazon[2]. Os dados passaram por uma fase de pré-processamento onde foram removidos todos os caracteres especiais e numéricos, além das *stopwords* (artigos, pronomes, etc.) dos textos coletados. Os autores utilizam duas implementações do SVM, o SVM^{perf} e o LibSVM, e comparam os resultados obtidos entre as combinações destes classificadores com os modelos Word2Vec e TF-IDF, mostrando que a união entre Word2Vec e SVM^{perf} obtém os melhores resultados na classificação.

Os trabalhos [Aguiar and Prati 2015] e [Silva et al. 2016] tem como foco o problema de detectar e filtrar SMS Spam através de técnicas de aprendizado de máquina. O diferencial da proposta de [Aguiar and Prati 2015] é a utilização da representação vetorial distribuída de palavras com Doc2Vec ao invés das técnicas de normalização textual e indexação semântica propostas por [Silva et al. 2016]. Os autores realizaram uma

[2]https://www.amazon.cn/

comparação entre ambas as técnicas, utilizando a mesma base de dados do trabalho de [Silva et al. 2016] e realizaram a classificação através de vários algoritmos a fim de verificar o desempenho de sua abordagem em relação à proposta por [Silva et al. 2016]. Os autores mostram que a utilização do Doc2Vec promoveu melhorias nos resultados dos principais classificadores, além de agregar em flexibilidade, uma vez que dispensa o uso de dicionários de gírias que são diretamente atrelados ao idioma das mensagens.

3. Classificadores de Aprendizado de Máquina

Os classificadores de aprendizado de máquina funcionam a partir de exemplos. Os algoritmos são treinados com dados cujas classes já são conhecidas e, a partir da análise dessas informações, conseguem inferir com certa precisão a classe de um novo dado.

3.1. Classificador Naive Bayes

Dentre os mais conhecidos classificadores, está o classificador Naive Bayes, um classificador probabilístico baseado no teorema de Bayes, em que se calcula a probabilidade de um determinado evento A ocorrer dada a ocorrência de um evento B. Esse cálculo é definido pela Equação 1:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)} \tag{1}$$

Intuitivamente, podemos reescrever o teorema de Bayes de modo a aproximá-lo da ideia de um problema de classificação, essa reescrita se dá na Equação 2:

$$P(c \mid f_1 \dots f_n) = \frac{P(f_1 \dots f_n \mid c)\, P(c)}{P(f_1 \dots f_n)} \tag{2}$$

onde c é uma determinada classe e $f_1 \dots f_n$ são as características levadas em conta no processo de classificação. No caso da classificação de polaridade de textos, $c \in \{\text{positivo}, \text{negativo}\}$ e $f_1, \dots, f_n$ seriam as palavras contidas no texto.

Um classificador Naive Bayes calcula essas probabilidades para cada classe possível e atribui ao elemento a classe que alcançou a maior probabilidade.

3.2. Máquinas de Vetores de Suporte

Máquinas de Vetores de Suporte (SVM, do inglês: *Support Vector Machines*) é um tipo de classificador linear que representa os elementos que serão classificados como um ponto disposto em um hiperplano. Esse hiperplano é dividido em várias regiões de modo que cada região contém elementos de uma determinada classe. Um ponto qualquer posicionado em um hiperplano é rotulado com a classe da região da qual ele mais se aproxima. Existe, ainda, uma margem de separação entre as classes, de modo que um ponto posicionado dentro de uma margem é classificado como neutro.

A Figura 1 apresenta um exemplo de SVM com duas classes.

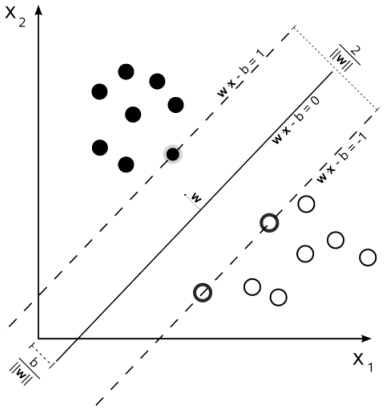

Figura 1. Exemplo de SVM com duas classes.

Fonte: [Duarte 2013]

Em um conjunto de dados linearmente separáveis, como ilustrado na Figura 1, podemos definir o hiperplano como o conjunto de pontos que satisfazem a Equação 3.

$$w \cdot x - b = 0 \tag{3}$$

onde w é o vetor normal para o hiperplano e $w \cdot x$ é o produto vetorial entre os vetores w e x. Na Figura 1, podemos identificar ainda outros dois hiperplanos paralelos que separam as duas classes de dados. Esses hiperplanos são descritos pelas Equações 4 e 5.

$$w \cdot x - b = 1 \tag{4}$$

$$w \cdot x - b = -1 \tag{5}$$

e a distância entre eles é dada por $\frac{2}{\|w\|}$.

O SVM busca maximizar essa distância, que determina a margem de separação entre as classes, a fim de reduzir a probabilidade de erros de classificação. Maximizar $\frac{2}{\|w\|}$ é equivalente a minimizar $\|w\|$ e como é desejável que nenhum vetor se encontre dentro da margem, podemos montar um problema de otimização dado por 6.

$$\text{Minimizar:} \|w\|; \text{sujeito a:} \begin{cases} w \cdot x - b = 1 \\ w \cdot x - b = -1 \end{cases} \tag{6}$$

O SVM é usado nos trabalhos de [Lochter et al. 2015, Zhang et al. 2015, Aguiar and Prati 2015] e, assim como o classificador Naive Bayes, será utilizado como método de classificação neste trabalho.

4. Representação Vetorial de Palavras com Word2Vec

O Word2Vec é um modelo proposto por [Mikolov et al. 2013] que visa a representação vetorial para incorporação de palavras utilizando técnicas de redes neurais artificiais [Aguiar and Prati 2015].

O modelo adota duas arquiteturas bastante eficientes: o *Continuous Bag-of-Words* (CBOW), em que uma palavra é predita com base em seu contexto, e o *Skip-gram*, que prediz o contexto com base em uma palavra. Em ambas as arquiteturas, o contexto de uma palavra é definido pelas palavras circunvizinhas a ela. A quantidade de palavras levadas em consideração no contexto é definida através de um valor inteiro normalmente chamado de janela de contexto. Um exemplo clássico dos aspectos semânticos captados pelo Word2Vec é que, com um conjunto de dados bem treinado, é possível empregar operações algébricas entre os vetores das palavras evidenciando regularidades linguísticas. Por exemplo:

$$vetor(\text{“rei”}) - vetor(\text{“homem”}) + vetor(\text{“mulher”}) = vetor(\text{“rainha”})$$

expressa a relação *“homem está para mulher assim como rei está para rainha”*[3].

A hipótese intuitiva da boa incorporação gerada pelo Word2Vec está no fato de que palavras com significados semelhantes possuem contextos semelhantes e vice-versa. Assim, os vetores das palavras, no espaço vetorial em que se encontram, expressam essas características linguísticas.

5. Classificação de Polaridade de *Tweets* Utilizando Word2Vec

Nesta seção são apresentados os procedimentos metodológicos realizados para a construção da representação vetorial dos *tweets* e de sua classificação. A Figura 2 descreve em linhas gerais a abordagem de classificação desenvolvida.

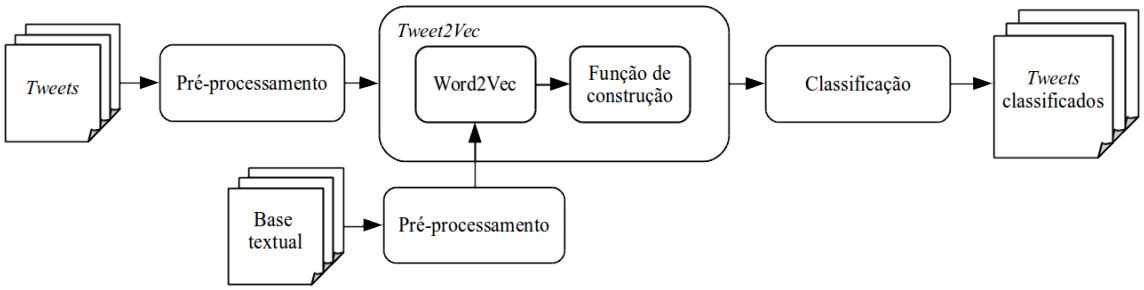

Figura 2. Passos até a classificação de polaridade dos *tweets*.

As bases de *tweets* passaram, inicialmente, por uma etapa de pré-processamento; em seguida, vários modelos Word2Vec foram construídos a partir das bases de *tweets* e de outras bases textuais; os vetores de palavras obtidos através dos modelos, foram utilizados para se obter uma representação vetorial para cada *tweet* por meio do que chamamos de função de construção; por fim, os vetores dos *tweets* foram classificados em positivos ou negativos.

As próximas seções detalham cada um dos passos apresentados. A Seção 5.4 apresenta os resultados obtidos na classificação dos *tweets*.

[3]Mais exemplos podem ser encontrados na página web da ferramenta: `https://code.google.com/archive/p/word2vec/`

5.1. Obtenção e Pré-processamento dos Dados

Em [Lochter et al. 2015], os autores construíram três bases de dados de *tweets* coletados durante o segundo semestre de 2014. Esses *tweets* foram rotulados através de uma ferramenta colaborativa[4] e disponibilizados publicamente através de uma página Web[5].

A Tabela 1 apresenta a quantidade de amostras pertencentes a cada classe, bem como o tema referente a cada uma das bases criadas e utilizadas pelos autores.

Tabela 1. Bases de dados utilizadas no trabalho.

Base de dados	#Positivas	#Negativas	Tema
Hobbit3	354	169	Filme
Archeage	724	994	Jogo
iPhone6	371	161	*Smartphone*

Fez-se necessário realizar uma etapa de pré-processamento dos dados a fim de remover *stopwords*, isto é, palavras consideradas irrelevantes no processo de classificação de polaridade como artigos, preposições, números e URLs. Todas as palavras restantes foram reduzidas ao seu radical e reescritas em caixa baixa.

5.2. Construção de Modelos Word2Vec

Os conjuntos de dados que serão classificados possuem poucas amostras de *tweets*, cada uma contendo até 140 caracteres, não sendo suficiente para gerar modelos Word2Vec adequados. Para que o Word2Vec consiga gerar vetores bem incorporados, faz-se necessária a utilização de um grande volume de textos, de modo que os *tweets*, por serem poucos e possuírem um tamanho reduzido, podem ser insuficientes para possibilitar uma boa representação vetorial das palavras. Além disso, é desejável que os textos utilizados como entrada no Word2Vec possuam temas semelhantes aos dos textos que serão classificados. Esses dois fatores, quantidade de textos e similaridade de temas, são evidenciados no trabalho de [Lai et al. 2016].

Levando esses dois fatores em consideração, utilizamos outras três bases de dados textuais[6], descritas pela Tabela 2, e compostas por várias *reviews* sobre temas semelhantes aos dos conjuntos de *tweets* anteriormente descritos.

Tabela 2. Bases de dados adicionais.

Base de dados	#Amostras	Tema
Filmes	64565	Filmes
Jogos	79437	Jogos digitais
Celulares	194430	Celulares e acessórios

[4] `http://lasid.sor.ufscar.br/ml-tools/`

[5] `http://dcomp.sor.ufscar.br/talmeida/sentcollection/`

[6] Disponíveis em: `https://www.kaggle.com/c/word2vec-nlp-tutorial/data`, `https://github.com/mulhod/steam_reviews` e `http://jmcauley.ucsd.edu/data/amazon/`. Acesso em: 20 out. 2017

Construímos assim, para cada uma das combinações *tweets* e *reviews*, dois modelos Word2Vec: um utilizando a arquitetura CBOW e outro a arquitetura Skip-gram. Os modelos criados a partir dos *tweets* utilizaram uma janela de contexto de tamanho 2, enquanto que os modelos criados através das *reviews*, por possuírem textos maiores, foram construídos utilizando uma janela de contexto de tamanho 10 com uma frequência mínima de 40 vezes para cada palavra, ou seja, cada palavra deve aparecer ao menos 40 vezes no conjunto de dados para que seja levada em consideração. Esses parâmetros, dados como entrada ao Word2Vec, foram selecionados empiricamente.

Os modelos foram gerados através da implementação do Word2Vec disponibilizada para a linguagem Python[7] por meio da biblioteca Gensim[8] [Řehůřek and Sojka 2010]. O resumo da construção destes modelos é apresentado na Tabela 3. Todos os modelos e os códigos desenvolvidos para sua construção, desde a etapa de pré-processamento, estão disponíveis no GitHub[9].

Tabela 3. Modelos Word2Vec.

Identificação do modelo	Base utilizada	Arquitetura	Janela de contexto	Frequência mínima
Hobbit3-SG	Hobbit3	Skip-gram	2	1
Hobbit3-CBOW	Hobbit3	CBOW	2	1
Archeage-SG	Archeage	Skip-gram	2	1
Archeage-CBOW	Archeage	CBOW	2	1
iPhone6-SG	iPhone6	Skip-gram	2	1
iPhone6-CBOW	iPhone6	CBOW	2	1
Filmes-SG	Filmes	Skip-gram	10	40
Filmes-CBOW	Filmes	CBOW	10	40
Jogos-SG	Jogos	Skip-gram	10	40
Jogos-CBOW	Jogos	CBOW	10	40
Celulares-SG	Celulares	Skip-gram	10	40
Celulares-CBOW	Celulares	CBOW	10	40

5.3. *Tweet2Vec*: Representação Vetorial dos *Tweets*

Tendo a representação vetorial das palavras para os conjuntos de *tweets* que pretendemos classificar, o passo seguinte consiste em representar cada *tweet* através desses vetores através de uma função de construção. Utilizamos a representação vetorial de um *tweet* através do vetor médio, calculado a partir dos vetores das palavras que ele contém.

Nessa abordagem, também utilizada nos trabalhos de [Jiang et al. 2016] e [Liu 2017], o vetor de cada *tweet*, $v(T)$, é calculado através da função de construção dada pela equação abaixo:

$$v(T) = \frac{1}{n} \sum_{w \in T} v(w)$$

onde T é um *tweet*, w uma palavra do *tweet* T e n é a quantidade de palavras que o *tweet* T possui. Supondo, por exemplo, que os vetores das palavras que compõem essa mensagem sejam $v(\text{“esse”}) = [1, 1, 1]$, $v(\text{“filme”}) = [1, 2, 3]$, $v(\text{“é”}) = [2, 1, 3]$,

[7] https://www.python.org/

[8] http://radimrehurek.com/gensim/index.html

[9] https://github.com/Raul3212/Word2Vec-Santiment-Analisys

v(“muito”) $= [3, 3, 3]$ e v(“bom”) $= [2, 2, 1]$, o vetor que representa o *tweet* $T =$ “esse filme é muito bom” seria dado por $v(T) = [1.8, 1.8, 2.2]$.

5.4. Classificação e Resultados

Os vetores dos *tweets* de cada base, gerados através dos modelos Word2Vec, foram submetidos aos classificadores Naive Bayes Bernoulli (NBB) e SVM linear (SVML) utilizando a técnica de validação cruzada com cinco partições. Assim, o conjunto de dados foi dividido em cinco subconjuntos mutuamente exclusivos e foram realizadas cinco execuções de cada classificador, onde, em cada execução, um subconjunto diferente é utilizado como conjunto de teste enquanto que os demais são utilizados para treino. O Scikit-learn[10] de [Pedregosa et al. 2011] foi utilizado na realização desses procedimentos.

Os vetores de cada modelo foram utilizados como características para a classificação das bases de *tweets* cujo assunto se assemelha ao da base textual que foi usada em sua construção. Assim, os modelos construídos a partir da base Filmes foram utilizados na classificação dos *tweets* da base Hobbit3; os modelos criados a partir da base Jogos foram empregados no processo de classificação dos *tweets* da base Archeage; e os modelos criados a partir da base Celulares foram utilizados para classificar os *tweets* da base iPhone6.

A Tabela 4 apresenta os resultados *F-measure* obtidos pelos classificadores para a base de *tweets* Hobbit3. É possível notar, primeiramente, que o resultado da classificação dos *tweets* utilizando os modelos do Word2Vec criados a partir da base de textos maior, nesse caso a base Filmes, obtiveram resultados melhores que os resultados obtidos utilizando os modelos baseados nos próprios *tweets*.

Tabela 4. Resultados de *F-measure* da classificação da base Hobbit3.

Classificador	Hobbit3-SG	Hobbit3-CBOW	Filmes-SG	Filmes-CBOW
NBB	0.737	0.750	**0.889**	0.841
SVML	0.808	0.805	**0.931**	0.930

A Tabela 5 mostra os resultados de *F-measure* da classificação para a base de *tweets* Archeage. É possível notar que os resultados da classificação foram, novamente, melhores com os modelos do Word2Vec construídos através da base maior, nesse caso a base Jogos. Nota-se, ainda, uma melhoria considerável na classificação dos vetores advindos dos modelos Word2Vec construídos através da arquitetura *Skip-gram* em relação aos modelos construídos usando CBOW, além de, novamente, o classificador SVML se sair melhor que o NBB.

Tabela 5. Resultados de *F-measure* da classificação da base Archeage.

Classificador	Archeage-SG	Archeage-CBOW	Jogos-SG	Jogos-CBOW
NBB	0.441	0.310	**0.743**	0.713
SVML	0.348	0.088	**0.793**	0.758

[10] `http://scikit-learn.org`

A Tabela 6 apresenta os resultados de *F-measure* obtidos pelos classificadores para a base de *tweets* iPhone6. Através dela é possível notar, mais uma vez, as disparidades entre os resultados de classificação utilizando modelos Word2Vec com altas e baixas quantidades de textos. Semelhantemente às Tabelas 4 e 5, os resultados obtidos pelo classificador SVML foram melhores que os do NBB.

Tabela 6. Resultados de *F-measure* obtidos pelos classificadores para a base de *tweets* iPhone6.

Classificador	iPhone6-SG	iPhone6-CBOW	Celulares-SG	Celulares-CBOW
NBB	0.654	0.743	**0.800**	0.743
SVML	0.821	0.820	**0.839**	0.836

5.5. Discussão

Os resultados apresentados nas Tabelas 4, 5 e 6 da Seção 5.4, mostram que ambos os classificadores utilizados conseguiram obter resultados melhores quando se utilizaram dos vetores gerados pelo Word2Vec a partir dos conjuntos de *reviews*, ainda que estes não estivessem exatamente relacionados com os textos que deveriam ser classificados. Além disso, os resultados também apontam que os vetores das palavras obtidos através de modelos construídos com a arquitetura *Skip-gram* proporcionaram melhores resultados que os vetores obtidos nos modelos construídos com a arquitetura CBOW.

OS melhores resultados com a classificação utilizando os modelos Word2Vec construídos a partir das *reviews* são obtidos na classificação da base Archeage. Nesse caso, a utilização da base de *reviews* para a construção dos modelos mostrou-se bastante eficaz, promovendo uma melhoria considerável nos resultados de *F-measure*, que, para a base Archeage com o classificador SVML e o modelo Jogos-SG, saltou de 0.348 para 0.793, mais que duplicando sua performance. Comparando os resultados do classificador SVML para as bases Hobbit3 e iPhone6 utilizando modelos *Skip-gram*, observamos um aumento de, aproximadamente, 15.22% e 2.19%, respectivamente.

Essa grande melhoria no resultado de classificação da base Archeage deve estar relacionada com a qualidade dos *tweets* da base. Enquanto que as bases Hobbit3 e iPhone6 contém *tweets* relativamente limpos e com pouquíssimas repetições, a base Archeage contém vários *tweets* repetidos e ruidosos. Os textos *"@archeage gameplay! todays goals are to get decent at arena pvp; go boating; work on trade skills"* e *"long-awaited mmo archeage is ddosed before it even launches"*, por exemplo, aparecem em 10 vezes e 43 vezes nos *tweets* da base, respectivamente. Isso faz com que o Word2Vec não consiga obter bons vetores, uma vez que as repetições interferem diretamente no contexto das palavras, implicando, assim, em resultados de classificação insatisfatórios. A utilização da base de *reviews* para a criação dos vetores de palavras possibilitou, portanto, a grande melhoria dos resultados da base Archeage.

Olhando para o tamanho das bases Filmes e Celulares (apresentados na Tabela 2), utilizadas na construção dos modelos Filmes-SG e Celulares-SG, respectivamente, percebemos que a base Celulares possui uma quantidade de *reviews* muito maior que a base Filmes. No entanto, o crescimento no resultado de classificação da base Hobbit3 é muito maior que o crescimento no resultado da classificação da base iPhone6. Isso provavelmente ocorre pelo fato de os *tweets* da base Hobbit3 serem naturalmente melhores que os

da base iPhone6, conterem menos amostras ou, ainda, pelo tema da base Celulares não se alinhar exatamente bem com o tema da base iPhone6.

A Tabela 7. apresenta a comparação entre os resultados obtidos por este trabalho e os de [Lochter et al. 2015].

Tabela 7. Comparação dos resultados de *F-measure* com o trabalho de [Lochter et al. 2015].

	Este Trabalho		**[Lochter et al. 2015]**		
Base de Dados	**NBB**	**SVML**	**NBB**	**SVML**	**Comitê**
Hobbit3	0.88	**0.93**	0.87	0.91	0.92
Archeage	0.74	0.79	0.87	0.84	**0.87**
iPhone6	0.80	**0.83**	0.74	0.71	0.74

A tabela mostra que a classificação das bases Hobbit3 e iPhone6, utilizando os modelos Word2Vec construídos a partir de *reviews* com a arquitetura *Skip-gram*, obteve resultados melhores que os obtidos por [Lochter et al. 2015] com os mesmos classificadores e até com o Comitê de Classificadores. Não possuindo, contudo, resultados melhores para a base Archeage.

Os resultados de classificação para a base Hobbit3 apresentaram um aumento de, aproximadamente, 1.14% e 2.19% em relação aos resultados de [Lochter et al. 2015] com os classificadores NBB e SVML, respectivamente. De modo particular, o classificador SVML obteve um aumento de, aproximadamente, 1.08% em relação ao Comitê de Classificadores, possuindo, contudo, um custo computacional muito menor, uma vez que na abordagem de Comitê, vários classificadores são executados sequencialmente.

Os resultados de classificação para a base iPhone6, por sua vez, apresentaram um aumento de, aproximadamente, 8.1% e 16.9% em relação aos resultados de [Lochter et al. 2015] com os classificadores NBB e SVML, respectivamente. O classificador SVML obteve um aumento de, aproximadamente, 12.16% em relação ao Comitê de Classificadores.

A sensibilidade do Word2Vec com relação à qualidade dos textos que lhes são fornecidos como entrada, fator evidenciado a partir das observações dos resultados de classificação da base Archeage, faz com que haja a necessidade da obtenção de boas bases textuais para que melhores resultados sejam alcançados.

A classificação da polaridade de sentimentos de mensagens curtas pode, dependendo da qualidade e da quantidade dos textos que se pretende classificar, requerer a obtenção dessas outras bases textuais para a geração de modelos. A similaridade entre os temas e o tamanho das bases, sendo o primeiro, o mais importante [Lai et al. 2016], são os principais requisitos necessários para a seleção de uma boa base textual que favoreça a construção de bons modelos.

6. Conclusão

Neste artigo apresentamos a utilização do Word2Vec, um modelo de rede neural utilizado para a representação vetorial incorporada de palavras, em um processo de classificação de

polaridade de textos, abordando principalmente o problema da classificação de polaridade de mensagens curtas, em nosso caso, *tweets*.

Alguns modelos Word2Vec foram construídos e os vetores das palavras fornecidos por esses modelos foram utilizados para a construção de uma representação vetorial para os *tweets*. Esses *tweets*, agora representados vetorialmente, foram classificados por dois classificadores de aprendizado de máquina, o classificador Naive Bayes com a distribuição de Bernoulli e o SVM linear.

Os resultados obtidos foram comparados entre si e nos mostraram que a qualidade dos vetores gerados pelo Word2Vec está diretamente relacionada com a quantidade de textos utilizada como entrada. Assim, a utilização do Word2Vec na classificação de mensagens curtas pode requerer uma quantidade de dados que está além dos dados que se pretende classificar para possibilitar a obtenção de resultados satisfatórios.

A representação vetorial do texto que se pretende classificar, com base nos vetores de suas palavras, também constitui-se como um passo importante para obtenção de bons resultados. Utilizamos, para representar vetorialmente um *tweet*, o cálculo da média entre os vetores de suas palavras como a abordagem principal.

A comparação dos resultados obtidos neste trabalho com os de [Lochter et al. 2015] nos levam a crer que o Word2Vec constitui-se como uma ferramenta promissora para a representação vetorial incorporada de palavras, possibilitando encontrar resultados satisfatórios quando utilizado para extração de características em um processo de classificação de polaridade de textos.

Como trabalhos futuros, serão criados novos modelos Word2Vec através de outros conjuntos de dados textuais, além da pesquisa e experimentação de novas abordagens para a representação vetorial das mensagens com base nos vetores de suas palavras.

Referências

Aguiar, R. F. and Prati, R. C. (2015). Incorporação de representação vetorial distribuída de palavras e parágrafos na classificação de sms spam. *ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Natal, Brasil*.

de França, T. C. and Oliveira, J. (2014). Análise de sentimento de tweets relacionados aos protestos que ocorreram no brasil entre junho e agosto de 2013. In *Proceedings of the III Brazilian Workshop on Social Network Analysis and Mining (BRASNAN)*, pages 128–139.

Duarte, E. S. (2013). *Sentiment analysis on Twitter for the Portuguese language*. PhD thesis, Universidade Nova de Lisboa.

Jiang, S., Lewris, J., Voltmer, M., and Wang, H. (2016). Integrating rich document representations for text classification. In *Systems and Information Engineering Design Symposium (SIEDS), 2016 IEEE*, pages 303–308. IEEE.

Lai, S., Liu, K., He, S., and Zhao, J. (2016). How to generate a good word embedding. *IEEE Intelligent Systems*, 31(6):5–14.

Liu, H. (2017). Sentiment analysis of citations using word2vec. *arXiv preprint arXiv:1704.00177*.

Lochter, J. V., Zanetti, R. F., and Almeida, T. A. (2015). Detecçao de opiniao em mensagens curtas usando comitê de classificadores e indexaçao semântica. *ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Natal, Brasil*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. Disponível em: `http://is.muni.cz/publication/884893/en`.

Saif, H., Fernandez, M., He, Y., and Alani, H. (2013). Evaluation datasets for twitter sentiment analysis: a survey and a new dataset, the sts-gold. [S.l: S.n].

Silva, T. P. d. et al. (2016). Normalização textual e indexação semântica aplicadas da filtragem de sms spam. [S.l: S.n].

Xue, B., Fu, C., and Shaobin, Z. (2014). A study on sentiment computing and classification of sina weibo with word2vec. In *Big Data (BigData Congress), 2014 IEEE International Congress on*, pages 358–363. IEEE.

Zhang, D., Xu, H., Su, Z., and Xu, Y. (2015). Chinese comments sentiment classification based on word2vec and svm perf. *Expert Systems with Applications*, 42(4):1857–1863.