
**UNIVERSIDADE FEDERAL DO CEARÁ**

**CAMPUS QUIXADÁ**

**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**RAUL DE ARAÚJO LIMA**

# CLASSIFICAÇÃO DE POLARIDADE DE SENTIMENTOS DE MENSAGENS CURTAS UTILIZANDO WORD2VEC

**QUIXADÁ – CEARÁ**

**2017**

RAUL DE ARAÚJO LIMA

# CLASSIFICAÇÃO DE POLARIDADE DE SENTIMENTOS DE MENSAGENS CURTAS UTILIZANDO WORD2VEC

Monografia apresentada no curso de Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Ciência da Computação.
Área de concentração: Computação.

Orientador: Dr. Paulo de Tarso Guerra Oliveira

QUIXADÁ – CEARÁ

2017

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

---

L71c Lima, Raul de Araújo.
Classificação de Polaridade de Sentimentos de Mensagens Curtas Utilizando Word2Vec / Raul de Araújo Lima. – 2017.
49 f. : il.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2017.
Orientação: Prof. Dr. Paulo de Tarso Guerra Oliveira.

1. Emoções-Análise. 2. Mensagens eletrônicas. 3. Word2Vec. I. Título.

CDD 004

RAUL DE ARAÚJO LIMA

# CLASSIFICAÇÃO DE POLARIDADE DE SENTIMENTOS DE MENSAGENS CURTAS UTILIZANDO WORD2VEC

Monografia apresentada no curso de Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Ciência da Computação.
Área de concentração: Computação.

Aprovada em: ___/___/_____

BANCA EXAMINADORA

---

Dr. Paulo de Tarso Guerra Oliveira (Orientador)
Universidade Federal do Ceará – UFC

---

Me. Regis Pires Magalhães
Universidade Federal do Ceará - UFC

---

Me. Victor Aguiar Evangelista de Farias
Universidade Federal do Ceará - UFC

A Deus.

Aos meus pais, Adailton e Jackeline.

## AGRADECIMENTOS

A Deus pela saúde, o amor, a persistência e a força diante das dificuldades.

À minha família que, com muita confiança e carinho, sempre me auxiliou e apoiou até que eu chegasse à esta etapa.

A todos os professores do curso de Ciência da Computação e, de maneira especial, aos professores Críston, Regis, Ticiana, Arthur Araruna e Wladimir, que, com seus exemplos de dedicação e empenho, foram tão importantes na minha vida acadêmica.

Ao professor Paulo de Tarso, pelo exemplo, a dedicação e a paciência na orientação deste trabalho.

À professora Carla Ilane, tutora do PET - Tecnologia da Informação, por todo o apoio e confiança.

Aos meus amigos e colegas de curso, Daiane, Décio, Dieinison, João Vitor, Ronildo e Salathiel, Ana Paula, Wallinson Deives e Rômulo por todo o aprendizado construído e compartilhado ao longo do curso.

Aos meus amigos e colegas bolsistas do PET - Tecnologia da Informação, Tiago, Micaele, Alexsandro, Jordy, Júlio, Igor e Érika, por todas as parcerias e vivências e, especialmente, por suportarem as minhas piadas sem graça.

“Peçam e vocês receberão, procurem e vocês encontrarão, batam e a porta vos será aberta; porque quem pede, recebe; quem procura, encontra; e a quem bate, a porta se abre.”

(Jesus Cristo)

## RESUMO

A análise de sentimentos e a classificação de polaridade de textos constitui-se como uma das principais ferramentas atualmente utilizadas por empresas e organizações para os mais variados fins. Dentre as muitas técnicas empregadas para essa finalidade, a representação vetorial incorporada de palavras, unida aos classificadores de Aprendizado de Máquina, tem se destacado e atraído a atenção da comunidade científica. Este trabalho faz uma análise da utilização da representação vetorial incorporada de palavras, construída através do Word2Vec, no processo de extração de *features* 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 *features* textuais, contribuindo para a obtenção de bons resultados de classificação.



**Palavras-chave:** Emoções-Análise. Mensagens eletrônicas. Word2Vec.

**ABSTRACT**

The Sentiment Analisys and the polarity classification of texts constitutes one of the main tools currently used by companies and organizations for the most varied purposes. Among the many techniques used for this purpose, the vector representation of words, together with the Machine Learning classifiers, is attracting the attention of the scientific community. This work makes 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, one of the largest social networks currently used. 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.

**Keywords:** Emotions-Analisys. Electronic Messages. Word2Vec.

## LISTA DE FIGURAS

Figura 1 – Exemplo de SVM com duas classes . . . . . 18
Figura 2 – Modelo de um neurônio de McCulloch e Pitts . . . . . 21
Figura 3 – Camadas do Word2Vec . . . . . 22
Figura 4 – Arquitetura CBOW . . . . . 24
Figura 5 – Arquitetura Skip-gram . . . . . 25
Figura 6 – Passos até a clasificação de polaridade dos *tweets*. . . . . 32

## LISTA DE TABELAS


Tabela 1 – Comparação entre os trabalhos relacionados e o presente trabalho. . . . . 28
Tabela 2 – Bases de dados utilizadas no trabalho. . . . . 33
Tabela 3 – Bases de dados adicionais. . . . . 33
Tabela 4 – Modelos Word2Vec. . . . . 34
Tabela 5 – Resultados de *F-measure* da classificação da base Hobbit3. . . . . 36
Tabela 6 – Resultados de *F-measure* da classificação da base Archeage. . . . . 36
Tabela 7 – Resultados de *F-measure* obtidos pelos classificadores para a base de *tweets* iPhone6. . . . . 36
Tabela 8 – Comparação dos resultados de *F-measure* com o trabalho de Lochter, Zanetti e Almeida (2015). . . . . 38
Tabela 9 – Resultados de *F-measure* da base Hobbit3 com vetores de *tweets* utilizando o TF-IDF. . . . . 40
Tabela 10 – Resultados de *F-measure* da base Archeage com vetores de *tweets* utilizando o TF-IDF. . . . . 40
Tabela 11 – Resultados de *F-measure* da base iPhone6 com vetores de *tweets* utilizando o TF-IDF. . . . . 41
Tabela 12 – Resultados de *F-measure* da base Hobbit3 com vetores de *tweets* utilizando *bag-of-clusters*. . . . . 42
Tabela 13 – Resultados de *F-measure* da base Archeage com vetores de *tweets* utilizando *bag-of-clusters*. . . . . 42
Tabela 14 – Resultados de *F-measure* da base iPhone6 com vetores de *tweets* utilizando *bag-of-clusters*. . . . . 42
Tabela 15 – Comparação entre os melhores resultados de *F-measure* obtidos nas três abordagens com todas as bases. . . . . 43
Tabela A.1 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados pela média. . . . . 47
Tabela A.2 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados pela média. . . . . 47
Tabela A.3 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados pela média. . . . . 47

Tabela A.4 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados com o TF-IDF. . . . . 48
Tabela A.5 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados com o TF-IDF. . . . . 48
Tabela A.6 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados com o TF-IDF. . . . . 48
Tabela A.7 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados como *bag-of-clusters*. . . . . 49
Tabela A.8 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados como *bag-of-clusters*. . . . . 49
Tabela A.9 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados como *bag-of-clusters*. . . . . 49

**SUMÁRIO**


**1 INTRODUÇÃO** . . . . . 13
**2 FUNDAMENTAÇÃO TEÓRICA** . . . . . 15
**2.1 Análise de Sentimentos** . . . . . 15
**2.2 Classificadores de Aprendizado de Máquina** . . . . . 16
***2.2.1 Classificador Naive Bayes*** . . . . . 16
***2.2.2 Máquinas de Vetores de Suporte*** . . . . . 17
**2.3 Métricas para Avaliação de Modelos de Classificação** . . . . . 19
**2.4 Representação Vetorial de Palavavras** . . . . . 20
***2.4.1 Representação One-hot*** . . . . . 20
***2.4.2 Word2Vec*** . . . . . 21
**2.5 Clusterização** . . . . . 26
**2.6 TF-IDF: *Term Frequency - Inverse Document Frequency*** . . . . . 26
**3 TRABALHOS RELACIONADOS** . . . . . 28
**3.1 Detecção de Opinião em Mensagens Curtas usando Comitê de Classificadores e Indexação Semântica** . . . . . 28
**3.2 Chinese Comments Sentiment Classification Based on Word2Vec and SVM$^{perf}$** . . . . . 29
**3.3 Análise de Sentimento de Tweets Relacionados aos Protestos que ocorreram no Brasil entre Junho e Agosto de 2013** . . . . . 30
**3.4 Incorporação de representação vetorial distribuída de palavras e parágrafos na classificação de SMS SPAM** . . . . . 30
**4 CLASSIFICAÇÃO DE POLARIDADE DE *TWEETS* UTILIZANDO WORD2VEC** . . . . . 32
**4.1 Obtenção e Pré-processamento dos Dados** . . . . . 32
**4.2 Construção de Modelos Word2Vec** . . . . . 33
**4.3 *Tweet2Vec*: Representação Vetorial dos *Tweets*** . . . . . 34
**4.4 Classificação e Resultados** . . . . . 35
**4.5 Discussão** . . . . . 37
**5 EXPERIMENTOS COMPLEMENTARES** . . . . . 40
**5.1 *Tweet2Vec* usando TF-IDF** . . . . . 40
**5.2 *Tweet2Vec* usando *bag-of-clusters*** . . . . . 41

**5.3** **Discussão** . . . . . 42
**6** **CONSIDERAÇÕES FINAIS** . . . . . 44
**REFERÊNCIAS** . . . . . 45
**APÊNDICE A – RESULTADOS DETALHADOS DE CLASSIFICAÇÃO** 47

# 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. Esse volume de mensagens faz desses ambientes virtuais excelentes veículos de dados a partir dos quais muitas empresas e organizações veem uma grande oportunidade para a criação de conhecimento (GOMES, 2013). No entanto, extrair informações úteis das mensagens difundidas através das redes sociais não é uma tarefa trivial.

Uma das maiores e mais utilizadas redes sociais é o Twitter[1], um *microblog* onde os usuários podem compartilhar textos e fotos, interagindo uns com os outros, difundindo opiniões e informações sobre os mais variados assuntos (FRANÇA; OLIVEIRA, 2014). Muitas empresas têm realizado pesquisas de *marketing* utilizando dados compartilhados através de redes sociais a fim de reduzirem a distância entre os seus produtos e as necessidades dos consumidores (LI; LI, 2011).

Uma particularidade do Twitter é que os *tweets*, nome dado às mensagens compartilhadas através da rede social, possuem no máximo 140 caracteres. Essa limitação, imposta pela própria rede social, contribui para que a linguagem utilizada pelos seus usuários seja, por vezes, bastante diferente da linguagem formalmente escrita. Textos extraídos do Twitter e de muitas outras redes sociais normalmente contêm gírias, abreviações de palavras e expressões próprias que dificultam o processo automático de sua análise e classificação (LOCHTER; ZANETTI; ALMEIDA, 2015). Nesse contexto, a análise de sentimentos, juntamente com o uso de técnicas de mineração de textos para a classificação e extração de conhecimento desses dados podem oferecer diversas vantagens nos mais variados campos (KONTOPOULOS et al., 2013).

Muitos trabalhos se propuseram a classificar textos. O trabalho de Lochter, Zanetti e Almeida (2015) propõe a utilização de técnicas de normalização léxica e indexação semântica unidas a um comitê de classificadores para determinar a polaridade de mensagens curtas extraídas do Twitter para fins de detecção de opinião. França e Oliveira (2014) unem as técnicas de análise de sentimentos e mineração de textos a um classificador Naive Bayes para identificar a opinião da população brasileira em relação às manifestações ocorridas entre junho e agosto de 2013. Aguiar e Prati (2015) propõem a representação vetorial distribuída de palavras para a classificação de SMS Spam. O trabalho de Zhang et al. (2015) utiliza a técnica Word2Vec (MIKOLOV et al.,

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

2013; GOLDBERG; LEVY, 2014) como ferramenta de representação vetorial de palavras e um classificador SVM para identificar a polaridade de sentimentos de comentários escritos em chinês.

A representação vetorial de palavras, e mais precisamente o Word2Vec, tem se destacado como uma ferramenta promissora em problemas de classificação de textos. Através de sua utilização, as palavras de um texto são representadas por vetores construídos através de técnicas de redes neurais, de modo a captar aspectos sintáticos e semânticos de cada palavra em relação ao vocabulário e independentemente do idioma dos textos, dando mais flexibilidade ao pré-processamento dos dados.

Neste trabalho, avaliamos o uso do Word2Vec como ferramenta para extração de *features* num processo de classificação de polaridade de sentimentos de mensagens curtas extraídas do Twitter, realizando um estudo comparativo entre essa abordagem e a abordagem proposta por Lochter, Zanetti e Almeida (2015). Para esta comparação, utilizamos as mesmas bases de *tweets* e alguns dos classificadores explorados pelos autores. Construímos alguns modelos para a representação vetorial de palavras através do Word2Vec e utilizamos os vetores das palavras para construir uma representação vetorial dos *tweets* que posteriormente foram classificados. O público alvo deste trabalho são, portanto, todas as pessoas interessadas nas áreas de análise e classificação de sentimentos de textos, e que buscam conhecer mais sobre o Word2Vec e as vantagens de sua utilização em processamento de linguagem natural.

Os capítulos seguintes estão organizados da seguinte maneira: o Capítulo 2 apresenta os principais conceitos necessários para a realização deste trabalho; o Capítulo 3 apresenta alguns trabalhos relacionados bem como as suas diferenças e semelhanças em relação ao presente trabalho; o Capítulo 4 descreve os procedimentos metodológicos realizados e os resultados obtidos por este trabalho, bem como as discussões sobre os resultados; o Capítulo 5 apresenta alguns experimentos complementares realizados neste trabalho; e, por fim, o Capítulo 6 apresenta as considerações finais sobre este trabalho.

# 2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados os principais conceitos, sua contribuição e importância no que diz respeito ao desenvolvimento deste trabalho.

## 2.1 Análise de Sentimentos

A análise de sentimentos, também chamada de mineração de opinião, é o estudo computacional de opiniões, sentimentos e emoções contidas em textos (LIU, 2010). De modo geral, as informações disponíveis ao redor do mundo podem ser categorizadas em dois tipos principais: fatos e opiniões (LIU, 2010). Os fatos são expressões objetivas sobre entidades, eventos e atributos; enquanto que as opiniões são expressões subjetivas que se referem aos sentimentos e à avaliação das pessoas em relação às entidades, eventos e atributos.

A forma com que as pessoas buscam opiniões e sentimentos mudou bastante com a chegada da Internet. Antes da Internet, as pessoas costumavam consultar a opinião de seus amigos e familiares sobre determinado produto ou assunto e as empresas realizavam pesquisas de opinião entre o público. Hoje, a grande maioria das opiniões pode ser facilmente encontrada através de sites, blogs e redes sociais (LIU, 2010).

Identificar opiniões é uma capacidade importante, pois ajuda na tomada de decisões. Ao conhecer a opinião do público, uma empresa pode, por exemplo, encontrar maneiras mais eficientes de promover seu produto e até identificar os atributos do produto que precisam ser melhorados. Conhecer as opiniões que envolvem assuntos políticos ajudam os próprios políticos a desenvolverem suas estratégias. A opinião dos participantes sobre um determinado evento pode ajudar a organização a repensar as atividades e tornar o evento mais próximo do gosto e da necessidade do público (GOMES, 2013). Nesses e em muitos outros casos, conhecer a opinião do público pode ser um fator primordial para o sucesso de um empreendimento.

Analisar os sentimentos expressos em um texto não é uma tarefa trivial. Não se pode, por exemplo, identificá-los apenas com base na presença ou ausência de palavras que normalmente expressam opiniões ou sentimentos positivos ou negativos (DOSCIATTI; FERREIRA; PARAISO, 2013). Os exemplos a seguir refletem algumas dessas dificuldades (LOCHTER; ZANETTI; ALMEIDA, 2015):

- Em *"Que computador ótimo!... Parou de funcionar no segundo dia de uso."*, o adjetivo positivo "ótimo" é usado para expressar uma opinião negativa

ironicamente.

- Em *"Esse filme acabou com a minha tristeza!"*, o texto expressa um sentimento positivo apesar de conter a palavra "tristeza".

Em relação às máquinas, os humanos conseguem desenvolver a atividade de identificar sentimentos e opiniões a partir de textos de maneira bem mais satisfatória. Em contrapartida, há a grande dificuldade de realizar esse processo para grandes volumes de textos e dados relevantes "escondidos" em sites, blogs e redes sociais, que trazem à tona a necessidade de sistemas e aplicações que, de maneira automatizada, realizem tal atividade (LIU, 2010).

Segundo Maynard e Funk (2011), a utilização de classificadores de aprendizado de máquina constitui-se como uma das principais técnicas utilizadas na detecção de sentimentos.

Com o uso dessas técnicas, a análise de sentimentos torna-se uma das principais ferramentas utilizadas em problemas que se propõem a classificar não somente sentimentos, mas quaisquer polaridades em textos (GOMES, 2013), constituindo-se, assim, como um dos conceitos fundamentais para a realização deste trabalho.

## 2.2 Classificadores de Aprendizado de Máquina

De maneira geral, um processo de classificação consiste em separar coisas em determinadas classes, como positivo ou negativo (DUARTE, 2013). Para isso, utiliza-se de funções que atuam através do reconhecimento de padrões.

Existem muitas formas de classificar sentimentos. A principal delas, e que será abordada aqui, consiste na utilização de algoritmos de aprendizado de máquina (DUARTE, 2013).

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.

### *2.2.1 Classificador Naive Bayes*

O classificador Naive Bayes é 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 por 2.1:

$$P(A \mid B) = \frac{P(B \mid A)\,P(A)}{P(B)} \tag{2.1}$$

Intuitivamente, podemos reescrever o teorema de Bayes de modo a aproximá-lo da ideia de um problema de classificação, essa reescrita se dá por 2.2:

$$P(c \mid f_1 \dots f_n) = \frac{P(f_1 \dots f_n \mid c)\,P(c)}{P(f_1 \dots f_n)} \tag{2.2}$$

onde $c$ é uma determinada classe e $f_1 \dots f_n$ são as características levadas em conta no processo de classificação.

Um classificador Naive Bayes calcula essas probabilidades para cada classe possível e atribui ao elemento a classe que alcançou a maior probabilidade. Esse cálculo é realizado por 2.3:

$$P(c \mid f_1 \dots f_n) = \arg\max_{c \in C} P(c) \prod_{k=1}^{n} P(f_k \mid c) \tag{2.3}$$

onde $C$ é o conjunto de todas as classes possíveis. Em um caso específico da classificação de sentimentos, poderíamos ter $C = \{\text{positivo}, \text{negativo}\}$ e $f_1 \dots f_n$ seriam as $n$ palavras pertencentes ao texto que será classificado.

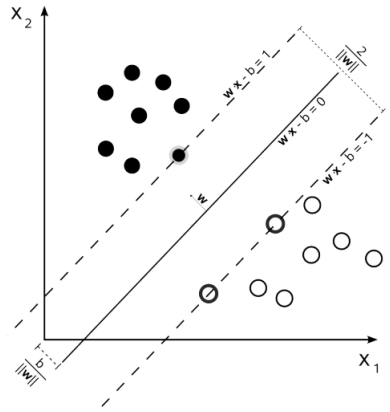
O classificador Naive Bayes tem se apresentado como um método bastante eficaz na análise de dados, não requerendo um grande conjunto de treinamento para conseguir bons resultados (FRANÇA; OLIVEIRA, 2014). Ele é utilizado nos trabalhos de Lochter, Zanetti e Almeida (2015), França e Oliveira (2014), Aguiar e Prati (2015) e será utilizado neste trabalho como método de classificação.

### *2.2.2 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 2.4.

$$w \cdot x - b = 0 \tag{2.4}$$

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 2.5 e 2.6.

$$w \cdot x - b = 1 \tag{2.5}$$

$$w \cdot x - b = -1 \tag{2.6}$$

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 2.7.

$$\text{Minimizar:} \|w\|; \text{sujeito a:} \begin{cases} w \cdot x - b = 1 \\ w \cdot x - b = -1 \end{cases} \tag{2.7}$$

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

## 2.3 Métricas para Avaliação de Modelos de Classificação

Para realizamos uma comparação entre os nossos resultados e os de Lochter, Zanetti e Almeida (2015), utilizaremos como métrica o valor de *F-measure*. Esse valor é dado a partir de duas outras medidas: precisão e cobertura. Estas, por sua vez, necessitam da compreensão dos conceitos de verdadeiro positivo, verdadeiro negativo, falso positivo e falso negativo, definidos a seguir:

- **Verdadeiros Positivos (TP, do inglês: *true positive*)** refere-se à quantidade de elementos que foram **corretamente** classificados como pertencentes à classe 'positivo'.
- **Verdadeiros Negativos (TN, do inglês: *true negative*)** refere-se à quantidade de elementos que foram **corretamente** classificados como pertencentes à classe 'negativo'.
- **Falsos Positivos (FP, do inglês: *false positive*)** refere-se à quantidade de elementos que foram **erroneamente** classificados como pertencentes à classe 'positivo'.
- **Falsos Negativos (FN, do inglês: *false negative*)** refere-se à quantidade de elementos que foram **erroneamente** classificados como pertencentes à classe 'negativo'.

As medidas de acurácia, precisão, cobertura e *F-measure* são dadas pelas seguintes fórmulas:

$$\text{Acurácia} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{Precisão} = \frac{TP}{TP+FP}$$

$$\text{Cobertura} = \frac{TP}{TP+FN}$$

$$F\text{-}measure = 2 \times \frac{\text{Precisão} \times \text{Cobertura}}{\text{Precisão} + \text{Cobertura}}$$

A acurácia mede a exatidão do classificador, a precisão é a razão entre a quantidade de itens classificados corretamente em uma determinada classe $C$ e a quantidade de itens classificados como $C$. A cobertura é a razão entre a quantidade de itens classificados corretamente em uma determinada classe $C$ e a quantidade total de itens que deveriam ter sido classificados em $C$. A *F-measure* é a média harmônica entre precisão e cobertura e expressa bem a ideia da qualidade da classificação, uma vez que é calculada com base na precisão e na cobertura.

## 2.4 Representação Vetorial de Palavavras

### *2.4.1 Representação One-hot*

A representação *One-hot* é uma das maneiras mais simples de representarmos palavras através de vetores. Nela, todas as palavras de um *corpus* são ordenadas através de alguma regra (como a ordem lexicográfica, por exemplo) e cada palavra é identificada por um vetor de mesma dimensão do vocabulário, o conjunto de todas as palavras do *corpus*, que contém o valor 1 na posição em que a palavra se encontra no vocabulário ordenado e o valor 0 nas demais posições.

Considerando, por exemplo, o *corpus* composto pelas seguintes sentenças:

*"o gato viu um rato"*,
*"o gato perseguiu o rato"* e
*"o rato subiu o telhado"*.

O vetor de vocabulário gerado a partir dessas sentenças é composto pelas seguintes palavras ordenadas lexicograficamente:

$$\left[ \text{“gato”} \quad \text{“o”} \quad \text{“perseguiu”} \quad \text{“rato”} \quad \text{“subiu”} \quad \text{“telhado”} \quad \text{“um”} \quad \text{“viu”} \right]$$

O vetor *One-hot* que representa a palavra "perseguiu" seria dado por:

$$X = \left[ \begin{array}{cccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array} \right]$$

Essa representação possui algumas desvantagens que dizem respeito à dimensão dos vetores de cada palavra, que deve ser igual à dimensão do vocabulário, e ao fato de que ela

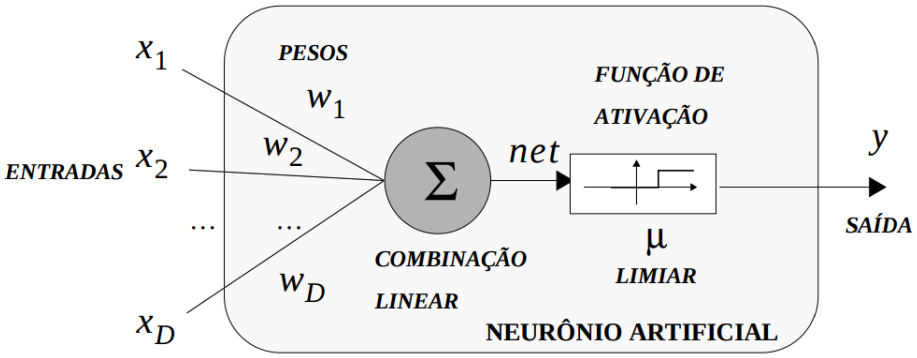
não expressa nenhuma relação sintática ou semântica entre as palavras, limitando-se apenas em identificá-las.

### *2.4.2 Word2Vec*

O Word2Vec é um modelo proposto por Mikolov et al. (2013) que visa a representação vetorial de palavras utilizando técnicas de redes neurais artificiais (AGUIAR; PRATI, 2015).

Uma rede neural artificial é um sistema computacional constituído por unidades processadoras interligadas, denominadas de neurônios, que trabalham juntas para desempenhar uma determinada tarefa (VELLASCO, 2007). Suas principais áreas de aplicação são classificação de padrões e previsão (VELLASCO, 2007). Os neurônios artificiais são a unidade básica de uma rede neural artificial e tentam simular as realidades biológicas que ocorrem no interior de uma célula do sistema nervoso (RAUBER, 2005). A Figura 2 apresenta o modelo de neurônio artificial proposto por McCulloch e Pitts (1943) que é utilizado como modelo básico de um neurônio artificial (RAUBER, 2005).

Figura 2 – Modelo de um neurônio de McCulloch e Pitts




Fonte – Rauber (2005)

Os valores $x_1, x_2, \ldots, x_D$ são as entradas do neurônio e $w_1, w_2, \ldots, w_D$, os pesos das sinapses, que são as arestas que interligam os neurônios, de modo que o peso $w_j$ da entrada $x_j$ determina a importância de $x_j$ para o processamento. A combinação linear gera o valor $net = \sum_{j=1}^{D} w_j x_j$ que, ao ser dado como entrada para um função de ativação $g$, gera a saída $y = g(net - \mu)$.

O modelo Word2Vec utiliza uma rede neural para identificar a similaridade entre as

palavras de um texto. Considerando novamente o conjunto de dados composto pelas seguintes sentenças:

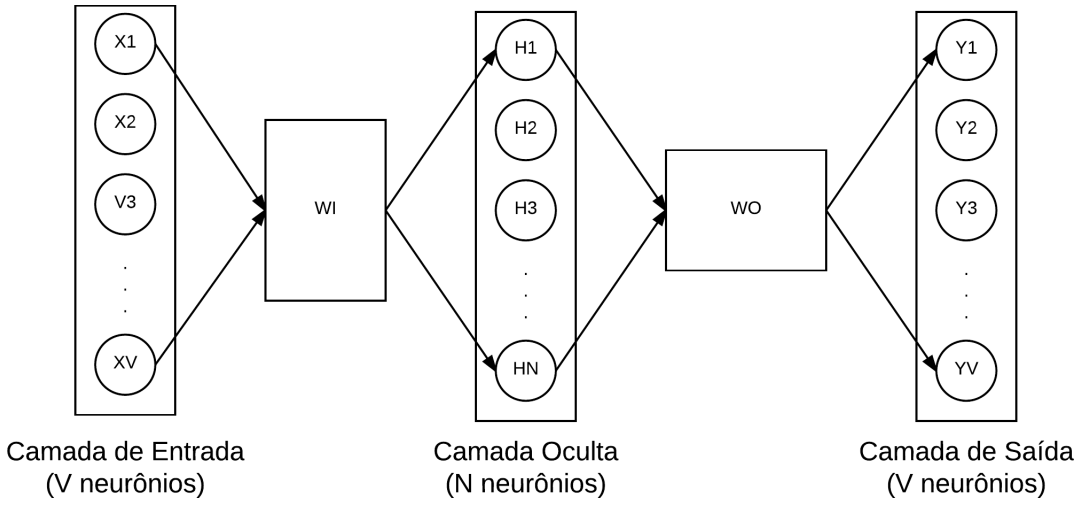
*“o gato viu um rato”*,
*“o gato perseguiu o rato”* e
*“o rato subiu o telhado”*.

Temos novamente o seguinte vetor de vocabulário ordenado lexicograficamente:

$$\left[ \text{“gato”} \quad \text{“o”} \quad \text{“perseguiu”} \quad \text{“rato”} \quad \text{“subiu”} \quad \text{“telhado”} \quad \text{“um”} \quad \text{“viu”} \right]$$

Seja $V$ o tamanho do vocabulário, nesse exemplo $V = 8$. O Word2Vec utiliza uma camada de entrada de dimensão $V$, que recebe um vetor na representação *One-hot* como entrada; uma camada oculta de dimensão $N$, em que $N$ diz respeito à dimensão dos vetores que representarão as palavras no modelo; e uma camada de saída de dimensão $V$, que retorna vetores semelhantes aos da representação *One-hot*, mas que, ao invés de 0’s e 1, esses vetores expressam probabilidades para cada palavra do vocabulário. As sinapses que interligam a camada de entrada à camada oculta, e a camada oculta à camada de saída, podem ser representadas, respectivamente, pelas matrizes $WI_{V\times N}$ e $WO_{N\times V}$. A Figura 3 mostra uma representação das camadas do Word2Vec.

Figura 3 – Camadas do Word2Vec




Fonte – Adaptado de Krishan (2015)

Como ilustração, suponha que $WI$ e $WO$ possuem dimensão $N = 3$ com os seguintes valores:

$$WI = \begin{bmatrix} 0.086 & 0.946 & 0.017 \\ 0.908 & 0.764 & 0.230 \\ 0.737 & 0.680 & 0.268 \\ 0.645 & 0.386 & 0.614 \\ 0.226 & 0.099 & 0.161 \\ 0.592 & 0.839 & 0.606 \\ 0.635 & 0.838 & 0.488 \\ 0.981 & 0.358 & 0.487 \end{bmatrix}$$

$$WO = \begin{bmatrix} 0.498 & 0.651 & 0.636 & 0.284 & 0.071 & 0.181 & 0.597 & 0.068 \\ 0.588 & 0.552 & 0.982 & 0.782 & 0.586 & 0.977 & 0.041 & 0.977 \\ 0.519 & 0.284 & 0.647 & 0.274 & 0.781 & 0.027 & 0.239 & 0.579 \end{bmatrix}$$

Suponha que queiramos que a rede aprenda a relação entre as palavras “rato” e “subiu”. Nesse caso, dado como entrada o vetor *One-hot* da palavra “rato”, desejamos verificar a probabilidade da palavra “subiu” a partir do vetor de saída. O vetor de entrada é dado por:

$$X = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Ao realizarmos a operação $R = (X \times WI) \times WO$, que representa a passagem do vetor $X$ da camada de entrada para a camada oculta e, em seguida, da camada oculta para a camada de saída, obtemos, aproximadamente, o seguinte resultado:

$$R = \begin{bmatrix} 0.866 & 0.807 & 1.186 & 0.653 & 0.751 & 0.510 & 0.547 & 0.776 \end{bmatrix}$$

Como o objetivo é calcular probabilidades, o Word2Vec transforma os valores de ativação da camada de saída em probabilidades através da função *softmax*. O resultado é calculado pela Equação 2.8:

$$P(w_k) = \frac{e^{w'_k}}{\sum_{n=1}^{V} e^{w'_n}} \tag{2.8}$$

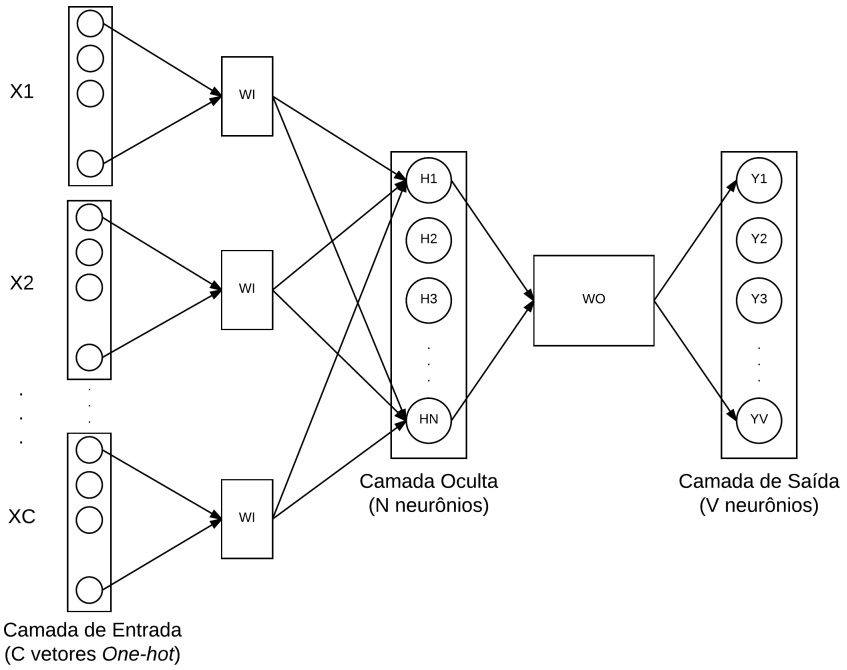
onde $w'_k$ é o resultado da ativação da camada de saída para a palavra $w_k$. A probabilidade condicional da palavra “subiu” dada a ocorrência da palavra “rato”é dada por:

$$P(\text{“subiu”} \mid \text{“rato”}) \simeq \frac{e^{0.751}}{17.5} \simeq 0.121$$

A probabilidade condicional nessa equação ocorre pelo fato do vetor $R$, do qual provém o valor $w'_k = 0.751$, ter sido gerado através do vetor *One-hot* da palavra "rato", dado como entrada.

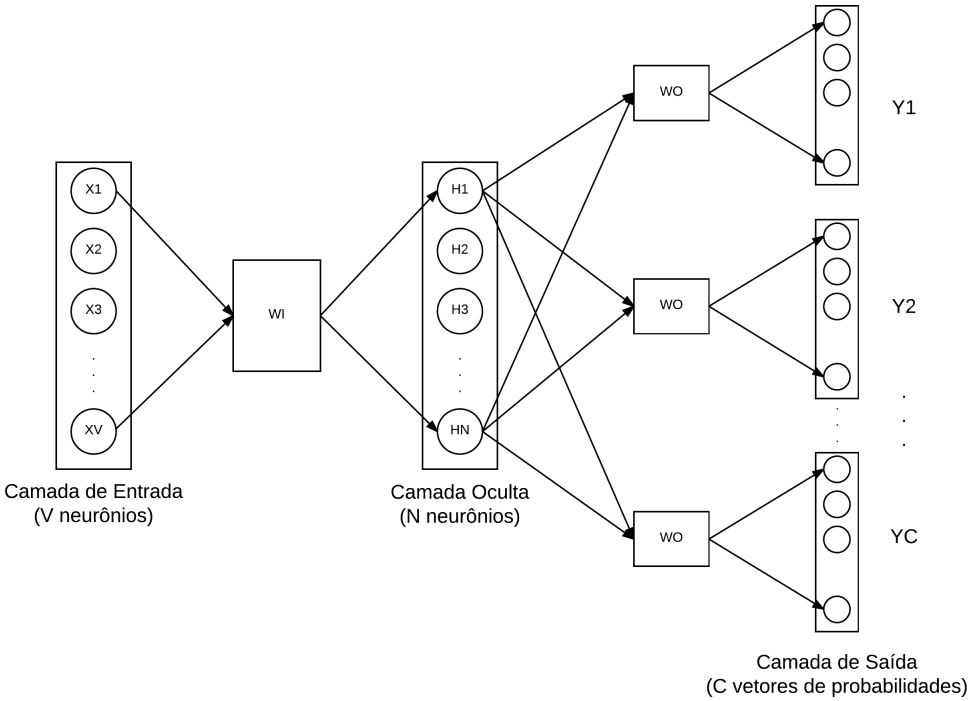
O Word2Vec adota duas arquiteturas bastante eficientes: o *Continuous Bag-of-Words* (CBOW), em que uma palavra $w_i$ é predita com base em seu contexto, que consiste de uma janela de palavras composta pelos termos anteriores e posteriores à $w_i$, $\{\ldots, w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}, \ldots\}$; e o *Skip-gram*, que busca maximizar a predição do contexto $\{\ldots, w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}, \ldots\}$ baseando-se em uma palavra $w_i$ dada como entrada. A mudança que ocorre com a utilização dessas arquiteturas é que, na arquitetura CBOW, a camada de entrada é replicada $C$ vezes, em que $C$ é a quantidade de palavras de contexto (Figura 4); e na arquitetura *Skip-gram* é a camada de saída que passa por essa replicação (Figura 5).

Figura 4 – Arquitetura CBOW




Fonte – Adaptado de Krishan (2015)

Figura 5 – Arquitetura Skip-gram




Fonte – Adaptado de Krishan (2015)

O Word2Vec utiliza o método de *backpropagation* para atualizar os pesos das sinapses, representados pelas matrizes $WI$ e $WO$, inicialmente geradas automaticamente. Esse método consiste em atualizar os pesos das sinapses com base no erro, que é a diferença entre a saída da rede e a saída esperada em cada caso de treino, visando minimizá-lo. Para isso, define uma função de perda e, derivando-a com relação a cada peso, busca o mínimo global.

Através dessas arquiteturas, o Word2Vec consegue captar aspectos sintáticos e semânticos entre as palavras de um vocabulário. Um exemplo clássico desses aspectos captados pelo Word2Vec é que, com um conjunto bem treinado de dados, 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”*[1].

Essa capacidade pode ser compreendida intuitivamente, uma vez que palavras que possuem tais regularidades são utilizadas em contextos similares que também possuem essas regularidades. Um ponto a ser ressaltado é que o Word2Vec funciona muito bem para grandes

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

conjuntos de dados. Pequenos conjuntos de dados não costumam possibilitar a captação desses aspectos.

O Word2Vec constitui-se como um dos principais conceitos necessários para a realização deste trabalho uma vez que objetivamos utilizar o Word2Vec no processo de classificação de sentimentos de textos.

## 2.5 Clusterização

De modo geral, um problema de clusterização consiste em agrupar elementos de um determinado conjunto de dados de modo que os elementos mais similares pertençam a um mesmo agrupamento (cluster), enquanto que os elementos menos similares são alocados em clusters distintos (OCHI; DIAS; SOARES, 2004).

A clusterização de dados é um dos métodos mais eficazes quando objetivamos agrupar elementos levando em consideração determinadas características, não conhecendo uma classe ou rótulo específico que possa identificar cada cluster.

Os problemas de clusterização são normalmente divididos em duas classes que se distinguem com relação à quantidade de clusters, quando esta é conhecida ou não (OCHI; DIAS; SOARES, 2004).

Dentre os algoritmos de clusterização, o *K-means* (MACQUEEN et al., 1967) constitui-se como um dos principais e mais utilizados. O algoritmo objetiva encontrar a melhor divisão dos dados em $K$ clusters de modo a minimizar a distância entre todos os dados de um determinado cluster e o seu respectivo centroide (PIMENTEL; FRANÇA; OMAR, 2003).

O *K-means* é utilizado em uma das abordagens apresentadas neste trabalho e constitui-se como uma ferramenta adicional para a construção de vetores que representam mensagens e para a análise da incorporação dos vetores advindos do Word2Vec.

## 2.6 TF-IDF: *Term Frequency - Inverse Document Frequency*

O TF-IDF (do inglês: *Term Frequency - Inverse Document Frequency*) é uma medida estatística usada para determinar a importância de uma palavra em um documento pertencente a uma coleção de documentos (corpus).

O TF-IDF calcula, para cada palavra de um documento, um valor baseado na frequência da palavra naquele documento (*Term Frequency*) e no inverso da porcentagem de

documentos em que aquela palavra aparece (*Inverse Document Frequency*), de modo que a palavra com maior valor de TF-IDF em um documento tende a estar mais relacionada a ele (LILLEBERG; ZHU; ZHANG, 2015).

A frequência de um termo $t$ em um documento $d$ (que é um conjunto de palavras), definida por $\text{TF}(t,d)$, é calculada da seguinte maneira:

$$\text{TF}(t,d) = 0.5 + \frac{0.5 \times f(t,d)}{max\{f(w,d) : w \in d\}}$$

onde $f(t,d)$ conta a quantidade de vezes que o termo $t$ aparece no documento $d$.

O inverso da frequência dos documentos que contém o termo $t$, no corpus $D$, denotado por $\text{IDF}(t,D)$, é calculado pela equação abaixo:

$$\text{IDF}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

onde $N$ é a quantidade de documentos em $D$.

Assim, o valor TF-IDF de uma palavra $t$ em um documento $d$ que pertence ao corpus $D$ é dado por:

$$\text{TF-IDF}(t,d,D) = \text{TF}(t,d) \times \text{IDF}(t,D)$$

O TF-IDF é amplamente utilizado em processos de classificação de textos e uma das abordagens apresentadas neste trabalho faz uso dele.

# 3 TRABALHOS RELACIONADOS

O presente trabalho busca utilizar técnicas de aprendizado de máquina, análise de sentimentos e o Word2Vec para a representação vetorial de palavras, com o objetivo de analisar e classificar a polaridade de *tweets* de maneira automática. Os trabalhos de França e Oliveira (2014) e Zhang et al. (2015) são os que mais contribuem para este trabalho no que diz respeito à análise de sentimentos. O trabalho de Aguiar e Prati (2015) também é um forte contribuinte por sua abordagem utilizando representação vetorial de palavras na classificação de SMS Spam.

A análise e mineração de textos e a utilização de classificadores como Naive Bayes ou SVM são elementos comuns presentes em (LOCHTER; ZANETTI; ALMEIDA, 2015; FRANÇA; OLIVEIRA, 2014; AGUIAR; PRATI, 2015). No entanto, o trabalho aqui proposto une essas técnicas à representação vetorial de palavras a fim de melhorar os resultados da classificação de sentimentos de *tweets*.

A Tabela 1 apresenta uma comparação entre os trabalhos relacionados e o presente trabalho.

Tabela 1 – Comparação entre os trabalhos relacionados e o presente trabalho.

| **Trabalho** | **Técnica Utilizada** | **Classificador** | **Tema** |
|---|---|---|---|
| Lochter, Zanetti e Almeida (2015) | Normalização léxica e Indexação semântica | Múltiplos Classificadores e Comitê de Classificadores | Opinião em Mensagens Curtas |
| Zhang et al. (2015) | Word2Vec | SVM | Sentimentos de Comentários Chineses |
| França e Oliveira (2014) | *Bag-of-words* | Naive Bayes | Protestos que ocorreram no Brasil em 2013 |
| Aguiar e Prati (2015) | Doc2Vec | Múltiplos Classificadores | SMS Spam |
| Este trabalho | Word2Vec | SVM e Naive Bayes | Sentimentos em Mensagens Curtas |

Fonte – Elaborado pelo autor

A seguir, temos uma breve descrição de cada trabalho.

## 3.1 Detecção de Opinião em Mensagens Curtas usando Comitê de Classificadores e Indexação Semântica

Lochter, Zanetti e Almeida (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.

Os autores utilizam sete bases de dados, sendo três delas construídas pelos próprios autores e pré-processadas utilizando a abordagem proposta por eles, e as demais pré-processadas segundo os procedimentos descritos por Saif et al. (2013).

Os autores mostraram que o comitê de classificadores proposto obteve as melhores taxas de *F-measure* em relação aos classificadores executados individualmente. Entretanto, é bastante notável a diferença entre os tempos de execução do comitê e dos demais classificadores. Por conta das combinações de técnicas de processamento de linguagem natural com as técnicas de classificação, o comitê de classificadores acaba exigindo um grande custo computacional que, em certos casos, pode inviabilizar a escolha por sua utilização.

Este trabalho propõe-se a solucionar o mesmo problema de Lochter, Zanetti e Almeida (2015), que é a detecção de opinião de mensagens curtas, fazendo uso, inclusive, dos *datasets* e classificadores utilizados pelos autores. Utilizaremos, contudo, a representação vetorial de palavras através do Word2Vec ao invés dos processos de normalização léxica e indexação semântica.

## 3.2 Chinese Comments Sentiment Classification Based on Word2Vec and SVM$^{perf}$

Zhang et al. (2015) buscam classificar comentários chineses em positivos ou negativos e propõem a utilização do Word2Vec combinado a um classificador SVM. O *dataset* utilizado continha cerca de 100.000 comentários sobre roupas coletados da página chinesa da Amazon[1]. Os dados passaram por uma fase de pré-processamento onde foram removidos dos textos todos os caracteres especiais e numéricos, além das *stopwords* (artigos, pronomes, etc.).

O trabalho propõe a utilização de dois métodos para seleção de atributos: um baseado nos dicionários HowNet e IARDict, que contém uma coleção de palavras que expressam opiniões, expandindo-a através do Word2Vec; e outro baseado em partes do discurso, considerando os adjetivos, advérbios, verbos e pronomes encontrados nos textos.

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

[1] <https://www.amazon.cn/>

Word2Vec e TF-IDF, mostrando que a união entre Word2Vec e SVM$^{perf}$ obtém os melhores resultados na classificação.

O trabalho de Zhang et al. (2015) é bastante semelhante a este trabalho sobretudo pela utilização do Word2Vec e do classificador SVM. Difere-se, no entanto, pelo fato de se propor a classificar comentários chineses (não necessariamente curtos) relacionados a roupas e por não fazer uso de outros tipos de classificadores além dos baseados em SVM.

## 3.3 Análise de Sentimento de Tweets Relacionados aos Protestos que ocorreram no Brasil entre Junho e Agosto de 2013

França e Oliveira (2014) defendem a possibilidade de realizar uma análise das opiniões da população brasileira referentes às manifestações ocorridas entre os meses de junho e agosto de 2013. Os autores coletaram *tweets* através de consultas que utilizavam *hashtags* (palavras-chaves que designam um determinado assunto). A base de dados final conta com mais de 300 mil *tweets*.

Após a coleta dos dados, foi realizada a etapa de pré-processamento, onde foram removidas as menções de perfis, URLs e *stopwords*, além da operação de *stemming*, onde realiza-se a extração dos radicais das palavras. O algoritmo Naive Bayes foi então utilizado para classificar a polaridade dos *tweets* e os resultados mostraram que a maioria dos *tweets* se mostrava a favor dos protestos.

O trabalho de França e Oliveira (2014) se assemelha ao presente trabalho pelo fato de se utilizar do classificador Naive Bayes juntamente com técnicas de análise de sentimentos e mineração de textos a fim de identificar o sentimento de determinados *tweets*. Difere-se, sobretudo, pelo fato de não utilizar representação vetorial de palavras e de limitar-se à utilização do classificador Naive Bayes.

## 3.4 Incorporação de representação vetorial distribuída de palavras e parágrafos na classificação de SMS SPAM

Aguiar e Prati (2015) buscam atacar o mesmo problema que Silva et al. (2014), que consiste em detectar e filtrar SMS Spam através de técnicas de aprendizado de máquina. O diferencial da proposta de Aguiar e Prati (2015) é a utilização da representação vetorial distribuída de palavras com Doc2Vec ao invés das técnicas de normalização textual e indexação semântica

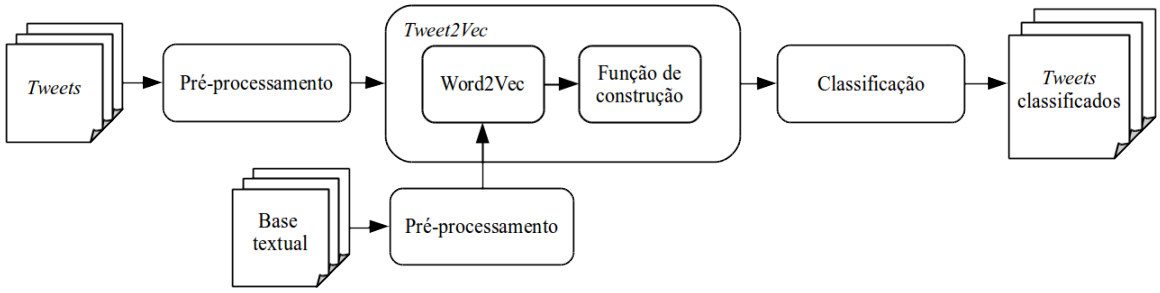
propostas por Silva et al. (2014). Os autores realizaram uma comparação entre ambas as técnicas, utilizando a mesma base de dados do trabalho de Silva et al. (2014) e realizam a classificação através de vários algoritmos a fim de verificar o desempenho de sua abordagem em relação à proposta por Silva et al. (2014). 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.

O trabalho de Aguiar e Prati (2015) é semelhante ao presente trabalho pelo fato de propor uma abordagem que substitui a utilização das técnicas de normalização léxica e indexação semântica pela representação vetorial de palavras, além de realizar testes com vários classificadores diferentes. Difere-se, no entanto, por utilizar o Doc2Vec e por se propor a classificar SMS SPAM.

# 4 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 6 descreve em linhas gerais a abordagem de classificação desenvolvida.

Figura 6 – Passos até a clasificação de polaridade dos *tweets*.




Fonte – Elaborado pelo autor.

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 4.4 apresenta os resultados obtidos na classificação dos *tweets* e a Seção 4.5 traz algumas discussões relacionadas aos resultados.

## 4.1 Obtenção e Pré-processamento dos Dados

Lochter, Zanetti e Almeida (2015) construíram três *datasets* de *tweets* coletados durante o segundo semestre de 2014. Esses *tweets* foram rotulados através de uma ferramenta colaborativa[1] e disponibilizados publicamente através de uma página Web[2].

A Tabela 2 apresenta a quantidade de amostras pertencentes a cada classe, bem como o tema referente a cada uma das bases criadas e utilizadas pelos autores.

[1] <http://lasid.sor.ufscar.br/ml-tools/>
[2] <http://dcomp.sor.ufscar.br/talmeida/sentcollection/>

Tabela 2 – Bases de dados utilizadas no trabalho.

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

Fonte – Adaptado de Lochter, Zanetti e Almeida (2015).

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.

## 4.2 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, os textos utilizados como entrada no Word2Vec devem possuir temas semelhantes aos dos textos que deverão ser 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, foram utilizadas outras três bases de dados textuais[3], descritas pela Tabela 3, e compostas por várias *reviews* sobre temas semelhantes aos dos conjuntos de *tweets* anteriormente descritos.

Tabela 3 – Bases de dados adicionais.

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

Fonte – Elaborado pelo autor.

Construímos, para cada uma das combinações *tweets* e *reviews*, dois modelos

[3] 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

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[4] por meio da biblioteca Gensim[5] (ŘEHŮŘEK; SOJKA, 2010). O resumo da construção destes modelos é apresentado na Tabela 4. Todos os modelos e os códigos desenvolvidos para sua construção, desde a etapa de pré-processamento, estão disponíveis no GitHub[6].

Tabela 4 – 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 |

Fonte – Elaborado pelo autor.

## 4.3 *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.

[4] <https://www.python.org/>
[5] <http://radimrehurek.com/gensim/index.html>
[6] <https://github.com/Raul3212/Word2Vec-Santiment-Analisys>

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. Exemplificando, considere um *tweet* $T =$ "esse filme é muito bom". Supondo que os vetores das palavras que compõem essa mensagem sejam $v(\text{"esse"}) = [1,1,1]$, $v(\text{"filme"}) = [1,2,3]$, $v(\text{"é"}) = [2,1,3]$, $v(\text{"muito"}) = [3,3,3]$ e $v(\text{"bom"}) = [2,2,1]$, o vetor que representa o *tweet* $T$ seria dado por $v(T) = [1.8,1.8,2.2]$.

## 4.4 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[7] de Pedregosa et al. (2011) foi utilizado na realização desses procedimentos.

Os vetores de cada modelo foram utilizados como *features* 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 5 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*.

[7] <http://scikit-learn.org>

Tabela 5 – 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 |

Fonte – Elaborado pelo autor

A Tabela 6 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 6 – 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 |

Fonte – Elaborado pelo autor

A Tabela 7 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 5 e 6, os resultados obtidos pelo classificador SVML foram melhores que os do NBB.

Tabela 7 – 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 |

Fonte – Elaborado pelo autor

## 4.5 Discussão

Os resultados apresentados nas Tabelas 5, 6 e 7 da Seção 4.4, 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.

Apesar de todas as bases terem obtido melhores resultados com a classificação utilizando os modelos Word2Vec construídos a partir das *reviews*, é somente nos resultados de classificação da base Archeage que essa diferença se torna mais clara. 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 3), 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 8. apresenta a comparação entre os resultados obtidos por este trabalho e os de Lochter, Zanetti e Almeida (2015).

Tabela 8 – Comparação dos resultados de *F-measure* com o trabalho de Lochter, Zanetti e Almeida (2015).

| | Este Trabalho | | Lochter, Zanetti e Almeida (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 |

Fonte – Elaborado pelo autor

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, Zanetti e Almeida (2015) com os mesmos classificadores e até com o Comitê de Classificadores. O mesmo não pode ser dito com relação à base Archeage, que obteve resultados inferiores.

Os resultados de classificação para a base Hobbit3 apresentaram um aumento de, aproximadamente, 1.14% e 2.19% em relação aos resultados de Lochter, Zanetti e Almeida (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, que, apesar de obter bons resultados, demanda um custo computacional muito maior, uma vez que 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, Zanetti e Almeida (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.

# 5 EXPERIMENTOS COMPLEMENTARES

Outras abordagens para a representação vetorial dos *tweets*, que consiste na mudança da função de construção, foram experimentadas. Suas descrições e resultados são apresentados nas seções seguintes.

## 5.1 *Tweet2Vec* usando TF-IDF

A primeira abordagem, assim como é proposto no trabalho de Lilleberg, Zhu e Zhang (2015), utiliza o valor de TF-IDF para atribuir pesos às palavras da base de dados, e representa os vetores de *tweets* através da média dos vetores das palavras que o compõem ponderada pelo TF-IDF. Assim, o vetor $v(T)$ passa a ser calculado conforme a função de construção dada pela equação abaixo:

$$v(T) = \frac{\sum_{w \in T} \text{TF-IDF}(w) \times v(w)}{\sum_{w \in T} \text{TF-IDF}(w)}$$

onde TF-IDF$(w)$ é o valor de TF-IDF da palavra $w$.

As Tabelas 9, 10 e 11 apresentam os resultados de *F-measure* obtidos na classificação dos *tweets* das bases Hobbit3, Archeage e iPhone6, respectivamente, representados vetorialmente pela média dos vetores de suas palavras, ponderada pelo TF-IDF de cada palavra.

Tabela 9 – Resultados de *F-measure* da base Hobbit3 com vetores de *tweets* utilizando o TF-IDF.

| Classificador | Hobbit3-SG | Hobbit3-CBOW | Filmes-SG | Filmes-CBOW |
|---|---|---|---|---|
| NBB | 0.849 | 0.901 | **0.940** | **0.940** |
| SVML | 0.808 | 0.808 | **0.940** | **0.940** |

Fonte – Elaborado pelo autor

Tabela 10 – Resultados de *F-measure* da base Archeage com vetores de *tweets* utilizando o TF-IDF.

| Classificador | Archeage-SG | Archeage-CBOW | Jogos-SG | Jogos-CBOW |
|---|---|---|---|---|
| NBB | 0.430 | 0.305 | 0.650 | **0.666** |
| SVML | 0.330 | 0.000 | **0.751** | 0.729 |

Fonte – Elaborado pelo autor

Tabela 11 – Resultados de *F-measure* da base iPhone6 com vetores de *tweets* utilizando o TF-IDF.

| Classificador | iPhone6-SG | iPhone6-CBOW | Celulares-SG | Celulares-CBOW |
|---|---|---|---|---|
| NBB | **0.822** | 0.772 | 0.728 | 0.672 |
| SVML | **0.821** | **0.821** | **0.821** | **0.821** |

Fonte – Elaborado pelo autor

## 5.2 *Tweet2Vec* usando *bag-of-clusters*

A segunda abordagem, utilizada no trabalho de Kim, Kim e Cho (2017), consiste em clusterizar os vetores de todas as palavras para, então, representar os vetores dos *tweets* através de um vetor *"bag-of-clusters"*, ou seja, cada *tweet* é representado por um vetor que conta a quantidade de palavras do *tweet* para cada cluster.

Consideremos o vocabulário abaixo:

$$[\text{“a”}, \text{“cachorro”}, \text{“correu”}, \text{“falou”}, \text{“gato”}, \text{“o”}, \text{“rato”}, \text{“viu”}]$$

Suponha que queiramos representar o texto *"o gato viu o rato"* através de um vetor *"bag-of-clusters"*. Nesse caso, o primeiro passo a ser realizado seria a clusterização das palavras do vocabulário. Suponha, agora, que nós realizamos a clusterização dos vetores dessas palavras e obtivemos os seguintes clusters:

$$\text{Cluster 1} = \{v(\text{“cachorro”}), v(\text{“gato”}), v(\text{“rato”})\},$$
$$\text{Cluster 2} = \{v(\text{“correu”}), v(\text{“falou”}), v(\text{“viu”})\} \text{ e}$$
$$\text{Cluster 3} = \{v(\text{“a”}), v(\text{“o”})\}.$$

Assim, o vetor *bag-of-clusters* que representaria o texto *"o gato viu o rato"* seria dado por $v(\text{“o gato viu o rato”}) = [2,1,2]$, uma vez que o texto possui duas palavras cujos vetores pertencem ao Cluster 1 (“gato” e “rato”), uma palavra cujo vetor pertence ao Cluster 2 (“viu”) e duas palavras cujos vetores pertecem ao Cluster 3 (“o” e “o”).

Intuitivamente, essa abordagem parece funcionar bem, uma vez que os vetores gerados pelo Word2Vec armazenam similaridades sintáticas e semânticas entre as palavras. Assim, agrupar palavras semelhantes e contabilizar quais desses agrupamentos estão mais relacionados com um texto, parece funcionar bem em um processo de classificação.

A clusterização dos vetores das palavras foi realizada através do algoritmo *K-means* utilizando uma média de cinco palavras por cluster. Os resultados de *F-measure* obtidos

com a utilização dessa abordagem para as bases Hobbit3, Archeage e iPhone6 são descritos, respectivamente, pelas Tabelas 12, 13 e 14.

Tabela 12 – Resultados de *F-measure* da base Hobbit3 com vetores de *tweets* utilizando *bag-of-clusters*.

| Classificador | Hobbit3-SG | Hobbit3-CBOW | Filmes-SG | Filmes-CBOW |
|---|---|---|---|---|
| NBB | 0.803 | 0.791 | **0.810** | 0.808 |
| SVML | 0.801 | 0.788 | **0.940** | 0.852 |

Fonte – Elaborado pelo autor

Tabela 13 – Resultados de *F-measure* da base Archeage com vetores de *tweets* utilizando *bag-of-clusters*.

| Classificador | Archeage-SG | Archeage-CBOW | Jogos-SG | Jogos-CBOW |
|---|---|---|---|---|
| NBB | 0.405 | 0.094 | **0.764** | 0.740 |
| SVML | 0.570 | 0.096 | **0.772** | 0.750 |

Fonte – Elaborado pelo autor

Tabela 14 – Resultados de *F-measure* da base iPhone6 com vetores de *tweets* utilizando *bag-of-clusters*.

| Classificador | iPhone6-SG | iPhone6-CBOW | Celulares-SG | Celulares-CBOW |
|---|---|---|---|---|
| NBB | 0.807 | 0.806 | 0.821 | **0.825** |
| SVML | 0.801 | 0.769 | 0.798 | **0.814** |

Fonte – Elaborado pelo autor

Os resultados de acurácia, precisão, cobertura e *F-measure* da classificação das bases de dados em todas as abordagens são apresentados através das tabelas presentes no Apêndice A.

## 5.3 Discussão

A comparação entre os melhores resultados obtidos na classificação de todas as bases em todas as abordagens experimentadas é apresentada na Tabela 15.

Tabela 15 – Comparação entre os melhores resultados de *F-measure* obtidos nas três abordagens com todas as bases.

| | Média | | TF-IDF | | Bag-of-clusters | |
|---|---|---|---|---|---|---|
| Base de Dados | NBB | SVML | NBB | SVML | NBB | SVM |
| Hobbit3 | 0.889 | 0.931 | **0.940** | **0.940** | 0.810 | 0.840 |
| Archeage | 0.743 | **0.793** | 0.666 | 0.751 | 0.764 | 0.772 |
| iPhone6 | 0.800 | **0.839** | 0.822 | 0.821 | 0.825 | 0.814 |

Fonte – Elaborado pelo autor

É possível notar que a utilização do TF-IDF para atribuir pesos às palavras fez com que a classificação da base Hobbit3 obtivesse resultados ainda melhores. O mesmo, no entanto, não ocorreu com as demais bases. Provavelmente isso seja novamente um reflexo das diferenças entre as qualidades textuais das bases. As repetições de *tweets* existentes na base Archeage, apresentadas anteriormente, influenciam diretamente no cálculo do TF-IDF fazendo com que palavras pouco importantes tenham seus pesos aumentados ou o contrário.

As Tabelas 9, 10 e 11 nos permitem notar ainda que, através dessa abordagem, os modelos criados com a arquitetura CBOW conseguiram promover melhores resultados de classificação que na abordagem com a média aritmética. Isso provavelmente ocorre porque o TF-IDF acaba por diminuir a influência dos vetores advindos dos modelos Word2Vec, prevalecendo sobre os resultados.

A alternativa *bag-of-clusters*, por sua vez, obteve apenas algumas melhorias pontuais com a utilização de determinados modelos e também reduziu as disparidades entre os resultados de classificação que utilizaram modelos CBOW com os que utilizaram modelos *Skip-gram*. Apesar de termos obtido resultados satisfatórios com essa abordagem, nenhum deles conseguiu superar os resultados já obtidos com as outras abordagens. A utilização dos clusters provavelmente ajudou a equilibrar esses resultados de modo que essa representação vetorial de *tweets* torna-os menos dependentes dos vetores advindos dos modelos Word2Vec.

Ambas as abordagens experimentadas, apesar de ajudarem na obtenção de bons resultados de classificação, não superaram a abordagem que se baseia no cálculo da média dos vetores. Devem, no entanto, ser consideradas sobretudo em processos de classificação de textos menos ruidosos que possibilitem a obtenção de melhores valores de TF-IDF ou de clusters mais bem definidos.

# 6 CONSIDERAÇÕES FINAIS

Neste trabalho apresentamos o Word2Vec, um modelo de rede neural utilizado para a representação vetorial incorporada de palavras, e a sua utilização 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. Duas abordagens alternativas também foram experimentadas: a primeira baseou-se na utilização do TF-IDF de cada palavra, enquanto que a segunda baseou-se na representação dos *tweets* como um vetor *bag-of-clusters*. As abordagens alternativas, apesar de serem eficazes como apresentam os trabalhos de Lilleberg, Zhu e Zhang (2015) e Mikolov et al. (2013), não promoveram grandes melhorias para a classificação das bases utilizadas neste trabalho.

A comparação dos resultados obtidos neste trabalho com os de Lochter, Zanetti e Almeida (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 *features* 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 em suas palavras.

# REFERÊNCIAS

AGUIAR, R. F.; PRATI, R. C. Incorporação de representação vetorial distribuída de palavras e parágrafos na classificação de sms spam. **ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Natal, Brasil**, 2015.

DOSCIATTI, M. M.; FERREIRA, L. P. C.; PARAISO, E. C. Identificando emoçoes em textos em português do brasil usando máquina de vetores de suporte em soluçao multiclasse. **ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Fortaleza, Brasil**, 2013.

DUARTE, E. S. **Sentiment analysis on Twitter for the Portuguese language**. Tese (Doutorado) — Universidade Nova de Lisboa, 2013.

FRANÇA, T. C. de; OLIVEIRA, J. 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)**. [S.l.: s.n.], 2014. p. 128–139.

GOLDBERG, Y.; LEVY, O. word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. **arXiv preprint arXiv:1402.3722**, 2014.

GOMES, H. J. C. **Text Mining: análise de sentimentos na classificação de notícias**. Tese (Doutorado), 2013.

JIANG, S.; LEWRIS, J.; VOLTMER, M.; WANG, H. Integrating rich document representations for text classification. In: IEEE. **Systems and Information Engineering Design Symposium (SIEDS), 2016 IEEE**. [S.l.], 2016. p. 303–308.

KIM, H. K.; KIM, H.; CHO, S. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. **Neurocomputing**, Elsevier, 2017.

KONTOPOULOS, E.; BERBERIDIS, C.; DERGIADES, T.; BASSILIADES, N. Ontology-based sentiment analysis of twitter posts. **Expert systems with applications**, Elsevier, v. 40, n. 10, p. 4065–4074, 2013.

KRISHAN, M. **Words as Vectors**. 2015. Disponível em: <https://iksinc.wordpress.com/tag/word2vec/>. Acesso em: 06 jun. 2017.

LAI, S.; LIU, K.; HE, S.; ZHAO, J. How to generate a good word embedding. **IEEE Intelligent Systems**, IEEE, v. 31, n. 6, p. 5–14, 2016.

LI, Y.-M.; LI, T.-Y. Deriving marketing intelligence over microblogs. In: IEEE. **System Sciences (HICSS), 2011 44th Hawaii International Conference on**. [S.l.], 2011. p. 1–10.

LILLEBERG, J.; ZHU, Y.; ZHANG, Y. Support vector machines and word2vec for text classification with semantic features. In: IEEE. **Cognitive Informatics & Cognitive Computing (ICCI* CC), 2015 IEEE 14th International Conference on**. [S.l.], 2015. p. 136–140.

LIU, B. Sentiment analysis and subjectivity. In: **Handbook of Natural Language Processing, Second Edition**. [S.l.]: Chapman and Hall/CRC, 2010. p. 627–666.

LIU, H. Sentiment analysis of citations using word2vec. **arXiv preprint arXiv:1704.00177**, 2017.

LOCHTER, J. V.; ZANETTI, R. F.; ALMEIDA, T. A. Detecçao de opiniao em mensagens curtas usando comitê de classificadores e indexaçao semântica. **ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Natal, Brasil**, 2015.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. **Proceedings of the fifth Berkeley symposium on mathematical statistics and probability**. [S.l.], 1967. v. 1, n. 14, p. 281–297.

MAYNARD, D.; FUNK, A. Automatic detection of political opinions in tweets. In: SPRINGER. **Extended Semantic Web Conference**. [S.l.], 2011. p. 88–99.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

OCHI, L. S.; DIAS, C. R.; SOARES, S. S. F. Clusterização em mineração de dados. **Instituto de Computação-Universidade Federal Fluminense-Niterói**, 2004.

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**, v. 12, p. 2825–2830, 2011.

PIMENTEL, E. P.; FRANÇA, V. F. de; OMAR, N. A identificação de grupos de aprendizes no ensino presencial utilizando técnicas de clusterização. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2003. v. 1, n. 1, p. 495–504.

RAUBER, T. W. Redes neurais artificiais. **Universidade Federal do Espírito Santo**, 2005.

ŘEHŮŘEK, R.; SOJKA, P. Software Framework for Topic Modelling with Large Corpora. In: **Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks**. Valletta, Malta: ELRA, 2010. p. 45–50. Disponível em: <http://is.muni.cz/publication/884893/en>.

SAIF, H.; FERNANDEZ, M.; HE, Y.; ALANI, H. **Evaluation datasets for twitter sentiment analysis: a survey and a new dataset, the sts-gold**. [S.l: S.n]. 2013.

SILVA, T. P.; SANTOS, I.; ALMEIDA, T. A.; HIDALGO, J. M. G. **Normalização Textual e Indexação Semântica Aplicadas na Filtragem de SMS Spam**. [S.l: S.n]. 2014.

VELLASCO, M. Redes neurais artificiais. **Rio de Janeiro, Brasil, PUC, notas de Aula, Brasil**, 2007.

ZHANG, D.; XU, H.; SU, Z.; XU, Y. Chinese comments sentiment classification based on word2vec and svm perf. **Expert Systems with Applications**, Elsevier, v. 42, n. 4, p. 1857–1863, 2015.

# APÊNDICE A – RESULTADOS DETALHADOS DE CLASSIFICAÇÃO

Tabela A.1 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados pela média.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Hobbit3-SG | NBB | 0.605 | 0.671 | 0.819 | 0.737 |
| Hobbit3-CBOW | NBB | 0.637 | 0.704 | 0.802 | 0.750 |
| Filmes-SG | NBB | 0.856 | 0.931 | 0.850 | **0.889** |
| Filmes-CBOW | NBB | 0.795 | 0.887 | 0.799 | 0.841 |
| Hobbit3-SG | SVML | 0.678 | 0.678 | 1.000 | 0.808 |
| Hobbit3-CBOW | SVML | 0.680 | 0.685 | 0.977 | 0.805 |
| Filmes-SG | SVML | 0.904 | 0.908 | 0.954 | **0.931** |
| Filmes-CBOW | SVML | 0.904 | 0.913 | 0.949 | 0.930 |

Fonte – Elaborado pelo autor

Tabela A.2 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados pela média.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Archeage-SG | NBB | 0.619 | 0.578 | 0.356 | 0.441 |
| Archeage-CBOW | NBB | 0.589 | 0.530 | 0.219 | 0.310 |
| Jogos-SG | NBB | 0.776 | 0.720 | 0.767 | **0.743** |
| Jogos-CBOW | NBB | 0.747 | 0.683 | 0.747 | 0.713 |
| Archeage-SG | SVML | 0.610 | 0.590 | 0.247 | 0.348 |
| Archeage-CBOW | SVML | 0.577 | 0.492 | 0.048 | 0.088 |
| Jogos-SG | SVML | 0.828 | 0.807 | 0.780 | **0.793** |
| Jogos-CBOW | SVML | 0.800 | 0.776 | 0.741 | 0.758 |

Fonte – Elaborado pelo autor

Tabela A.3 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados pela média.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| iPhone6-SG | NBB | 0.548 | 0.702 | 0.611 | 0.654 |
| iPhone6-CBOW | NBB | 0.616 | 0.697 | 0.795 | 0.743 |
| Celulares-SG | NBB | 0.731 | 0.829 | 0.773 | **0.800** |
| Celulares-CBOW | NBB | 0.616 | 0.697 | 0.795 | 0.743 |
| iPhone6-SG | SVML | 0.697 | 0.697 | 1.0 | 0.821 |
| iPhone6-CBOW | SVML | 0.703 | 0.710 | 0.970 | 0.820 |
| Celulares-SG | SVML | 0.757 | 0.780 | 0.908 | **0.839** |
| Celulares-CBOW | SVML | 0.755 | 0.784 | 0.894 | 0.836 |

Fonte – Elaborado pelo autor

Tabela A.4 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados com o TF-IDF.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Hobbit-SG | NBB | 0.772 | 0.768 | 0.949 | 0.849 |
| Hobbit-CBOW | NBB | 0.858 | 0.850 | 0.960 | 0.901 |
| Filmes-SG | NBB | 0.921 | 0.961 | 0.920 | **0.940** |
| Filmes-CBOW | NBB | 0.921 | 0.961 | 0.920 | **0.940** |
| Hobbit-SG | SVML | 0.678 | 0.678 | 1.000 | 0.808 |
| Hobbit-CBOW | SVML | 0.678 | 0.678 | 1.000 | 0.808 |
| Filmes-SG | SVML | 0.921 | 0.961 | 0.920 | **0.940** |
| Filmes-CBOW | SVML | 0.921 | 0.961 | 0.920 | **0.940** |

Fonte – Elaborado pelo autor

Tabela A.5 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados com o TF-IDF.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Archeage-SG | NBB | 0.616 | 0.575 | 0.343 | 0.4304 |
| Archeage-CBOW | NBB | 0.575 | 0.492 | 0.220 | 0.305 |
| Jogos-SG | NBB | 0.683 | 0.608 | 0.698 | 0.650 |
| Jogos-CBOW | NBB | 0.703 | 0.633 | 0.703 | **0.666** |
| Archeage-SG | SVML | 0.596 | 0.548 | 0.236 | 0.330 |
| Archeage-CBOW | SVML | 0.578 | 0.000 | 0.000 | 0.000 |
| Jogos-SG | SVML | 0.789 | 0.748 | 0.754 | **0.751** |
| Jogos-CBOW | SVML | 0.762 | 0.701 | 0.761 | 0.729 |

Fonte – Elaborado pelo autor

Tabela A.6 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados com o TF-IDF.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| iPhone6-SG | NBB | 0.699 | 0.698 | 1.000 | **0.822** |
| iPhone6-CBOW | NBB | 0.661 | 0.727 | 0.822 | 0.772 |
| Celulares-SG | NBB | 0.654 | 0.804 | 0.665 | 0.728 |
| Celulares-CBOW | NBB | 0.610 | 0.812 | 0.574 | 0.672 |
| iPhone6-SG | SVML | 0.697 | 0.697 | 1.000 | **0.821** |
| iPhone6-CBOW | SVML | 0.697 | 0.697 | 1.000 | **0.821** |
| Celulares-SG | SVML | 0.699 | 0.701 | 0.989 | **0.821** |
| Celulares-CBOW | SVML | 0.701 | 0.704 | 0.983 | **0.821** |

Fonte – Elaborado pelo autor

Tabela A.7 – Resultados da classificação da base Hobbit3 utilizando vetores de *tweets* calculados como *bag-of-clusters*.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Hobbit-SG | NBB | 0.680 | 0.688 | 0.963 | 0.803 |
| Hobbit-CBOW | NBB | 0.672 | 0.695 | 0.918 | 0.791 |
| Filmes-SG | NBB | 0.683 | 0.682 | 1.000 | **0.810** |
| Filmes-CBOW | NBB | 0.678 | 0.678 | 1.0 | 0.808 |
| Hobbit-SG | SVML | 0.678 | 0.688 | 0.960 | 0.801 |
| Hobbit-CBOW | SVML | 0.666 | 0.692 | 0.915 | 0.788 |
| Filmes-SG | SVML | 0.917 | 0.926 | 0.954 | **0.940** |
| Filmes-CBOW | SVML | 0.793 | 0.825 | 0.881 | 0.852 |

Fonte – Elaborado pelo autor

Tabela A.8 – Resultados da classificação da base Archeage utilizando vetores de *tweets* calculados como *bag-of-clusters*.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| Archeage-SG | NBB | 0.621 | 0.598 | 0.306 | 0.405 |
| Archeage-CBOW | NBB | 0.575 | 0.469 | 0.052 | 0.094 |
| Jogos-SG | NBB | 0.809 | 0.795 | 0.736 | **0.764** |
| Jogos-CBOW | NBB | 0.788 | 0.768 | 0.714 | 0.740 |
| Archeage-SG | SVML | 0.689 | 0.684 | 0.488 | 0.570 |
| Archeage-CBOW | SVML | 0.573 | 0.448 | 0.053 | 0.096 |
| Jogos-SG | SVML | 0.802 | 0.750 | 0.795 | **0.772** |
| Jogos-CBOW | SVML | 0.783 | 0.729 | 0.772 | 0.750 |

Fonte – Elaborado pelo autor

Tabela A.9 – Resultados da classificação da base iPhone6 utilizando vetores de *tweets* calculados como *bag-of-clusters*.

| Modelo Word2Vec | Classificador | Acurácia | Precisão | Cobertura | *F-measure* |
|---|---|---|---|---|---|
| iPhone6-SG | NBB | 0.682 | 0.698 | 0.956 | 0.807 |
| iPhone6-CBOW | NBB | 0.706 | 0.747 | 0.876 | 0.806 |
| Celulares-SG | NBB | 0.712 | 0.723 | 0.951 | 0.821 |
| Celulares-CBOW | NBB | 0.719 | 0.729 | 0.951 | **0.825** |
| iPhone6-SG | SVML | 0.687 | 0.719 | 0.905 | 0.801 |
| iPhone6-CBOW | SVML | 0.669 | 0.748 | 0.792 | 0.769 |
| Celulares-SG | SVML | 0.710 | 0.776 | 0.822 | 0.798 |
| Celulares-CBOW | SVML | 0.729 | 0.780 | 0.851 | **0.814** |

Fonte – Elaborado pelo autor